ІНФОРМАТИКА

Профільний рівень

підручник для 10 класу
закладів загальної середньої освіти

Харків
Видавництво «Ранок»
2018

УДК [004:37.016](075.3)
Р83

Рекомендовано Міністерством освіти і науки України
(наказ Міністерства освіти і науки України від 31.05.2018 № 551)

Р83 Інформатика (профільний рівень) : підруч. для 10 кл. закл. загал. серед. освіти / В. Д. Руденко, Н. В. Речич, В. О. Потієнко. - Харків : Вид-во «Ранок», 2018.
ISBN

УДК [004:37.016](075.3)

Інтернет-підтримка
Електронні матеріали
до підручника розміщено на сайті
interactive.ranok.com.ua

ISBN

Шановні десятикласники та десятикласниці!

За п'ять років ви опанували базові знання та сформували компетенції, маєте первинні навички практичної роботи з багатьма програмними засобами. Ви досягли певного рівня інформаційної культури і здатні самостійно оволодівати сучасними інформаційними технологіями. Та інформатика — дуже динамічна наука. Її подальші напрямки й темпи розвитку значною мірою визначатимуться рівнем підготовки людей, які мають ґрунтовні знання в цій галузі. Імовірно, багатьох із вас, адже ви вивчатимете інформатику на профільному рівні і, сподіваємося, в майбутньому пов'яжете з нею свою професійну діяльність.

У 10 класі ви будете працювати з новими програмними засобами. Ви зануритеся у світ статистики, фінансових розрахунків, інфографіки; навчитеся автоматизувати процес оформлення текстового документа, створювати та опрацьовувати графічні зображення у векторному та растровому графічних редакторах.

Під час опанування курсу багато навчального часу приділяється мовам програмування та структуруванню даних. Пропонований підручник — серед перших, де вивчаються основи об'єктно-орієнтованого програмування. Оволодіння основами алгоритмізації і програмування здійснюється на основі мови Python і середовища програмування IDLE.

Бажаємо вам успіхів,
авторський колектив

Підручник, який ви тримаєте в руках, — ваш надійний помічник. У ньому ви знайдете завдання для самостійного виконання — виконуйте їх на комп'ютері з натхненням, повторюйте теоретичний матеріал і викладайте основні положення на папері.

Описи практичних робіт, запропонованих до курсу інформатики, ви знайдете на сайті «Інтерактивне навчання» (interactive.ranok.com.ua).

Скориставшись цим посиланням, ви також зможете пройти комп'ютерне тестування з автоматичною перевіркою результату.

Різнорівневі питання для перевірки знань і завдання для самостійного виконання відповідають рівням навчальних досягнень:

— початковий і середній рівні

— достатній рівень

— високий рівень

У тексті використано також позначки:

 — означення, висновок

 — питання на повторення

 — зверніть увагу

 — цікаво знати

 — завдання для виконання та обговорення в парах або групах

 — вправи для домашнього виконання

Розділ 1. МОВА ПРОГРАМУВАННЯ ТА СТРУКТУРИ ДАНИХ

1. Структура і способи виконання проектів мовою Python

1.1. Класифікація і складові мов програмування

Із якими мовами програмування ви ознайомилися в попередніх класах? Які задачі розв'язували за їх допомогою?

Історію комп'ютерних наук до певної міри можна подати як історію мов програмування, початок розвитку яких припадає на XIX ст., коли англійський учений Чарльз Беббідж розробив механічну обчислювальну машину. Програму для неї, як вам відомо, написала леді Ада Лавлейс. Мови програмування в сучасному розумінні фактично почали розвиватися з появою електронних обчислювальних машин.

Ада Августа Лавлейс, дочка лорда Байрона, розробила перші програми для аналітичної машини Беббіджа, заклавши тим самим теоретичні основи програмування. На її честь названо мову програмування ADA.

Мова програмування (англ. *Programming language*) — це штучна мова, створена для розробки програм, які призначено для виконання на комп'ютері.

Комп'ютерна програма (англ. *Computer Program*) — це послідовність команд (інструкцій), що забезпечує реалізацію на комп'ютері конкретного алгоритму.

Команда (інструкція) — це вказівка, що визначає, яку дію (операцію) слід виконувати.

Сьогодні можна нарахувати понад 2 тис. різних мов програмування та їх модифікацій, проте лише окремі набули широкого визнання. Усі мови програмування можна умовно класифікувати за деякими *основними ознаками* (рис. 1).

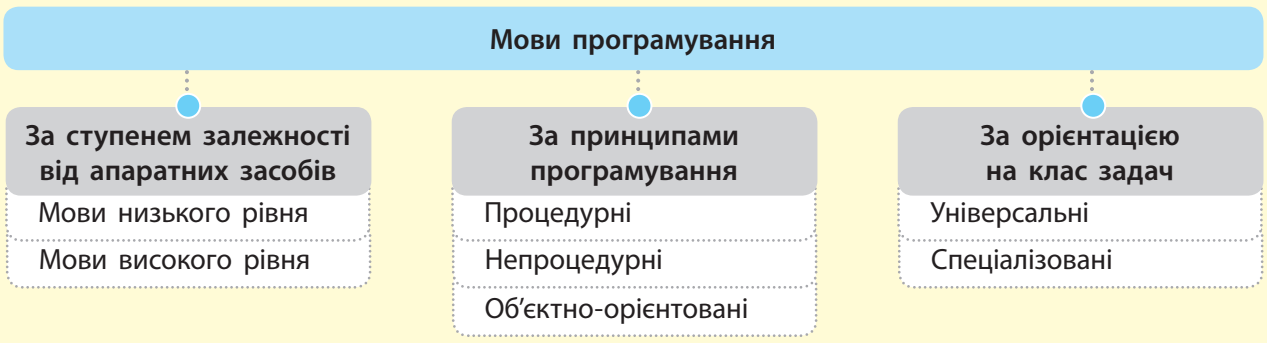


Рис. 1. Класифікація мов програмування

- За ступенем залежності від апаратних засобів розрізняють *мови програмування низького і високого рівнів*.

Мови програмування низького рівня (машинно-орієнтовані) — мови, у яких команди та дані враховують архітектуру комп'ютера. Такі мови орієнтовані на конкретний тип комп'ютера і враховують його особливості.

Практично кожний тип комп'ютера мав власну мову програмування низького рівня. Одна й та сама програма не могла виконуватися на комп'ютері іншого типу, що суттєво обмежувало можливість обміну програмами.

Програми для перших ЕОМ розробляли саме «машинними» мовами. Це був складний процес, тому невдовзі з'явилися мови символьного кодування. Команди подавалися вже не двійковим кодом, а символами. Перетворення символьного коду в машинні команди здійснюється автоматично.

Зазвичай команди сучасних мов програмування записують англійськими літерами з використанням символів, які містяться на клавіатурі. Але в комп'ютері зберігаються й виконуються команди, які подано фізичними сигналами (наприклад, двома рівнями остаточної магнітної індукції, двома значеннями електричної напруги, наявністю та відсутністю світлового променя тощо). Значення фізичних сигналів ототожнюються з математичними значеннями 0 і 1, тобто двійковими символами.

Програми, що подано сукупністю 0 і 1, називають машинними, або машинним кодом. Він указує, яку саме дію слід виконати процесору.

Використовуються різні структури команд. Найчастіше команди складаються з операційної та адресної частин. В операційній частині зазначається яку дію (операцію) слід виконати, а в адресній — виконати над якими даними (приклад).

У нашому випадку код A3 може бути операційною частиною й означати, наприклад, операцію Додати, а B7 і C5 — адресною частиною, яка визначає місце збереження даних, над якими слід виконати операцію.

Уже на перших етапах розвитку обчислювальної техніки почалося розроблення мов, доступних для широкого кола користувачів і не пов'язаних із конкретним комп'ютером. Першою мовою високого рівня, яка набула визнання програмістів, була Fortran.

Процес розроблення програм дещо полегшився, коли до мов символьного кодування почали включати макрокоманди, що реалізуються послідовністю з кількох машинних команд. До різновидів мов символьного кодування належать мови асемблера й автокода.

Мови програмування високого рівня (машинно-незалежні) — мови, на яких програми можуть використовуватися на комп'ютерах різних типів і які більш доступні людині, ніж мови низького рівня.

На різних етапах розвитку комп'ютерної техніки популярністю користувалися різні мови програмування.

Приклад.
Спрощено команду двійковим або шістнадцятковим кодом можна записати так:

10100011 10110111 11000101
або
A3 B7 C5

Вагомий внесок у розвиток теорії мов програмування зробила Катерина Логвинівна Ющенко. Вона написала перші програми для першої ЕОМ, створеної у НАН України під керівництвом С. О. Лебедєва.

- За принципами програмування розрізняють *процедурні*, *непроцедурні* мови та мови *об'єктно-орієнтованого програмування*.

Процедурні мови ґрунтуються на описі послідовної зміни стану комп'ютера, тобто значення комірок пам'яті, стану процесора й інших пристроїв. Вони маніпулюють даними в покроковому режимі, використовуючи послідовні інструкції. У процедурних мовах витримано чітку структуризацію програм, тому їх ще називають *мовами структурного програмування*. До таких мов належать Fortran, Algol, Pascal, BASIC тощо.

Процедурні мови повністю задовольняють потреби розроблення невеликих програм і програм середньої складності. Але на початку 80-х років XX ст. обсяг і складність програм досягли рівня, який вимагав нових концептуальних підходів до програмування.

Непроцедурні мови є ефективними для програмування пошуку даних у великих обсягах, а також для програмування задач, процес розв'язування яких неможливо описати точно (переклад, розпізнавання образів). У цих мовах саму процедуру пошуку розв'язку вбудовано в інтерпретатор мови. До таких мов належать мови *функціонального* і *логічного програмування*.

Наприкінці XX ст. було презентовано нову методику програмування, що отримала назву об'єктно-орієнтованого програмування (ООП). Тобто почали розвиватися мови, що містять конструкції, які дають змогу визначати об'єкти, що належать класам і мають властивості роботи з абстрактними типами даних. До таких мов належать C++, Java, C#, Python та ін. Нині мови ООП практично витіснили з ринку професійного програмування процедурні мови.

- За орієнтацією на клас задач мови програмування поділяють на *універсальні* та *спеціалізовані*.

Універсальні мови призначені для розв'язування широкого класу задач. До таких мов належать PL/1, Algol, Pascal, C тощо. Особливим класом універсальних мов є візуальні середовища програмування: VisualBasic, Delphi й ін.

Спеціалізовані мови враховують специфіку предметної галузі. На цей час існують десятки спеціалізованих мов програмування, наприклад, мови веб-програмування, мови скриптів тощо. *Мова скриптів* використовується для створення невеликих допоміжних програм, мова Javascript — для створення динамічних об'єктів на веб-сторінках. *Мови розмітки* містять шаблони та засоби опису вмісту, структури й формату електронних документів, наприклад мова HTML забезпечує розмітку гіпертекстового документа. *Мови для роботи з базами даних* забезпечують створення й супровід баз даних.

Зазначимо, що не всі з перелічених мов у класичному розумінні є мовами програмування. Так, мова HTML є мовою розмітки гіпертексту, але її також часто помилково називають мовою програмування.

Першою мовою високого рівня, яка набула широкого визнання серед програмістів світу, була **Fortran**. Її було розроблено корпорацією IBM (США) у 1954 році. Мова **Fortran** наближена до мови алгебри та орієнтована на розв'язування обчислювальних задач.
У 1960 році групою вчених різних країн створено мову **Algol-60**, теж орієнтовану на розв'язування обчислювальних задач.

День програміста святкують у 256-й день року (у високосний рік це 12 вересня, а в невисокосний — 13 вересня). Як ви думаєте, чому обрано саме цей день?

Вибір пояснюється тим, що це число символічне, воно тісно пов'язане з комп'ютерами, але не асоціюється з конкретними особами чи кодами спеціальностей. Число 256 відповідає кількості символів, які можна подати за допомогою одного байта.

Починаючи з 60-х років XX ст. розвиток мов програмування відбувається як шляхом спеціалізації, так і шляхом універсалізації.

Однією з перших спеціалізованих мов була мова COBOL, розроблена в США 1961 року й орієнтована на розв'язування економічних задач. Згодом з'явилися десятки різних спеціалізованих мов, наприклад, Simula — мова моделювання, LISP — мова для інформаційно-логічних задач, RPG — мова для розв'язування навчальних задач тощо.

Будь-яка мова програмування високого рівня, як і будь-яка інша мова, має основні складові (рис. 2): *алфавіт*, *синтаксис*, *семантику*.

> Найкращий спосіб у чомусь розібратися до кінця — це спробувати навчити цього комп'ютер.
>
> *Дональд Ервін Кнут*

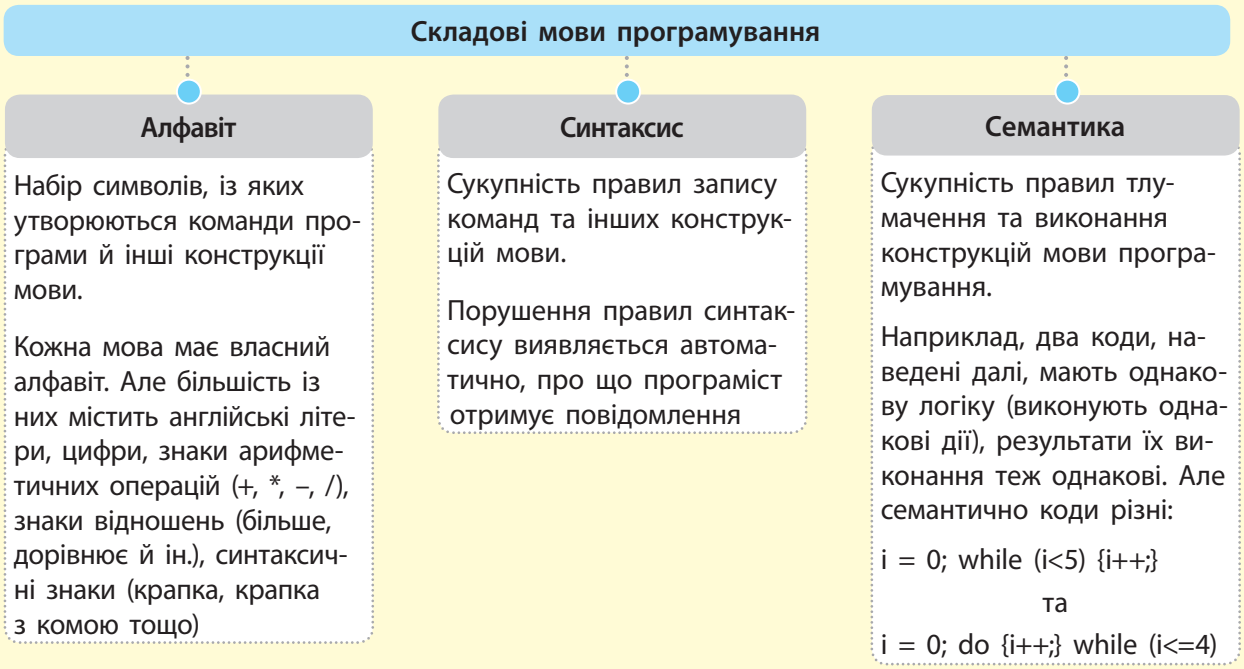


Рис. 2. Основні складові мови програмування

Мова програмування має словник — певну кількість слів, правила вживання яких визначено певною мовою і які мають строго визначене призначення. Такі слова називають *зарезервованими (ключовими)*, наприклад, for, input, if, print.

Запитання для перевірки знань

1. Що таке мова програмування?
2. Які мови називають машинно-орієнтованими?
3. Які мови називають мовами програмування високого рівня?
4. За якими ознаками класифікують мови програмування?
5. Як класифікуються мови програмування за орієнтацією на клас задач?
6. Як класифікуються мови програмування за принципами програмування?
7. Чому виникла потреба в об'єктно-орієнтованому програмуванні?
8. Назвіть основні складові мов програмування.
9. Поясніть на прикладах сутність синтаксису мови програмування.

1.2. Призначення і склад середовища програмування

Які середовища програмування ви використовували в попередніх класах? Назвіть їх переваги й недоліки.

Для зручної розробки програм існують спеціальні засоби їх створення, — середовища (системи) програмування, які забезпечують весь цикл роботи з програмою — від її розроблення до виконання й отримання необхідних результатів.

Вважається, що першим пристроєм із програмним керуванням був ткацький верстат, побудований Жозефом Марі Жаккаром у 1804 році. Верстат здійснив революцію в ткацькій промисловості: Жаккар віднайшов можливість за допомогою перфокарт програмувати візерунки на тканинах.

Середовище програмування — це комплекс програмних засобів, які призначено для автоматизації процесу підготовки та виконання програм користувача.

Розглянемо основні складові середовища програмування (рис. 1).

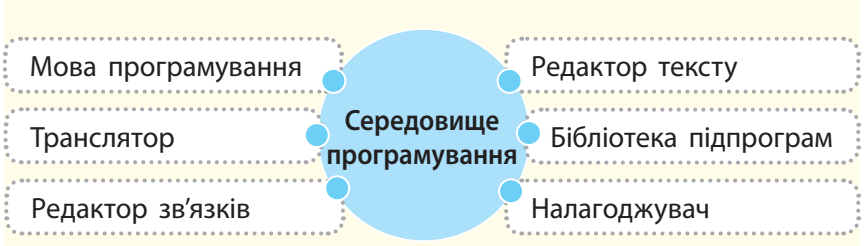


Рис. 1. Складові середовища програмування

Для свідомого розуміння призначення складових середовища програмування опишемо етапи процесу розробки програми, пов'язані з використанням комп'ютера.

Етап 1

Уводимо текст розробленої програми, яку називають *початковим кодом*, у комп'ютер і зберігаємо в пам'яті. Для цього середовище програмування має редактор тексту, який забезпечує введення й редагування початкового коду.

Етап 2

Після введення програми та виправлення помилок, які могли статися під час уведення, здійснюється перетворення програми з мови програмування високого рівня у двійковий код.

Таке перетворення здійснюється за допомогою транслятора програм.

Розрізняють два типи трансляторів: *компілятори* й *інтерпретатори*.

У процесі *інтерпретації* з початкового програмного коду послідовно кожна команда (інструкція) перетворюється у двійковий код і відразу виконується, — на екрані висвітлюється результат її виконання. Після завершення виконання однієї команди виконується наступна і так до останньої команди. Але результат перетворення не зберігається, і кожного запуску програма починається спочатку.

У процесі *компіляції* здійснюється перетворення всього тексту програмного коду у двійковий код. Отриману після компіляції програму називають *об'єктним модулем*. Така програма ще не готова до виконання.

Початковий код зазвичай містить посилання на інші модулі (підпрограми), які містяться в бібліотеці підпрограм (наприклад, модуль обчислення квадратного кореня). Таким чином, до програмного модуля потрібно додати коди необхідних підпрограм, щоб підготувати програму для виконання.
Компільовані програми виконуються швидше за інтерпретовані. Режим інтерпретації потребує додаткової основної пам'яті, оскільки інтерпретатор повинен увесь час зберігатися разом із кодом. Але інтерпретація в роботі зручніша. Особливо для програмістів, які лише починають працювати з середовищем програмування, оскільки контролюється результат кожної команди.

Етап 3

Після компіляції редактор зв'язків «склеює» окремі двійкові модулі в єдину програму, яка називається програмою, що виконується, і яка вже призначена для виконання.
Цей процес (*етапи 1–3*) подано схемою (рис. 2).

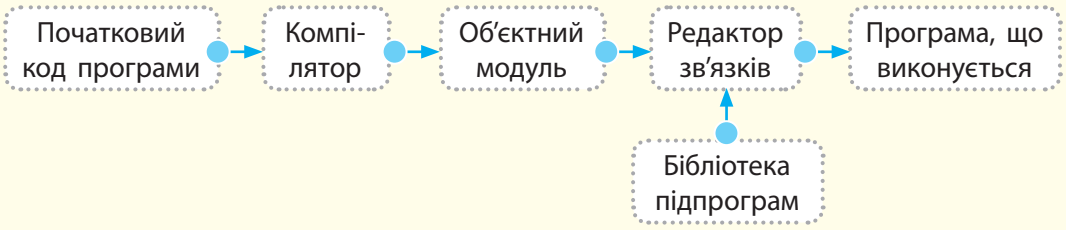


Рис. 2. Процес опрацювання початкового коду програми

Для подальшого виконання програмного коду, що виконується, компілятор не потрібен. Отже, після компіляції програма подана двійковими символами 1 і 0.

Етап 4

Отримана програма, що виконується, не гарантує, що немає логічних помилок. Вона може виконуватися, але результат виконання може бути неправильним. Тож потрібно здійснити тестування (випробування) програми на предмет виявлення й усунення в ній логічних помилок.
Тестування — досить відповідальний етап. У великих ІТ-компаніях над розробленням програм, які називають проектами, працюють десятки й навіть сотні програмістів різних напрямків. Одні з них розробляють проекти, інші займаються тестуванням програм, економічним обґрунтуванням тощо.
На цьому етапі застосовується налагоджувач програм, який дає змогу покроково аналізувати програму. Налагоджувач дозволяє виконувати трасирування програми, встановлювати й видаляти контрольні точки в програмах, умову призупинення виконання програми тощо.

Описаний процес розробки програм є класичним для процедурних мов програмування. Для програм, розроблених мовою ООП, є відмінності. Їх сутність полягає в тому, що після компіляції отримується не машинний, а проміжний код, так званий байт-код. За допомогою спеціального програмного забезпечення він потім перетворюється на машинний.

Байт-код — це проміжний код між початковим кодом і кодом, що виконується.

Такий підхід зумовлений тим, що в Інтернеті вільно переміщуються дані та програми (*аплети* — невеликі програми, призначені для передавання через Інтернет і виконання

Аплет переносить окремі функції із сервера до клієнта. У разі клацання посилання, що містить аплет, він автоматично завантажується на комп'ютер і запускається в браузері.

Потребу в розробці мов ООП із використанням байт-коду на початку 1990-х років зумовлено розвитком виробництва побутових приладів зі вбудованими контролерами на різних типах процесорів.

Спочатку виникла потреба в програмах і компіляторах для кожного типу процесора. Та розробка компілятора виявилася справою доволі дорогою. Так з'явилася ідея створити мову для розробки коду, придатного для виконання на будь-якому типі процесора.

За рік виникла потреба в мові, яку можна було б використовувати на будь-якому типі процесора. Це було зумовлене розвитком Всесвітньої павутини й Інтернету, які об'єднали різні типи комп'ютерів із різними процесорами й ОС.

в браузері, сумісному з мовою програмування). Їх потрібно захистити від вірусів та інших шкідливих програм, а також реалізувати переносимість програм.

Під переносимістю розуміють можливість завантаження й виконання аплета на комп'ютерах із будь-яким типом процесора, будь-якою операційною системою та браузером, що під'єднані до Інтернету. Саме ці проблеми й дозволяє розв'язати байт-код.

Зрозуміло, що використання будь-якого проміжного коду, у тому числі й байт-коду, знижує швидкість виконання програм і потребує додаткових апаратних засобів. Утім, ці втрати незначні порівняно з отриманим виграшем. Якби ООП-програма одразу компілювалася в машинний код, то для кожного комп'ютера зі своїм типом процесора необхідно було б мати окрему версію тієї самої програми, що економічно вкрай невигідно.

Інколи використовуються так звані динамічні компілятори. Їх сутність полягає в тому, що байт-код компілюється в машинний код не весь одразу, а окремими фрагментами, у міру необхідності. Інші частини коду можуть виконуватися в режимі інтерпретації. Цим самим досягається висока ефективність роботи з кодом.

Середовища (системи) програмування часто іменуються за назвою мови, яка в них реалізується, наприклад середовище Pascal, середовище Delphi. Інколи назва середовища містить префікс, який вказує на розробника середовища: назва середовища Turbo-C означає, що її розробником є фірма Borland.

Нині все частіше використовуються інтегровані середовища програмування, які забезпечують роботу з кількома мовами. Такими середовищами є, наприклад, IntelliJ IDEA, Eclipse. Варіант Ultimate Edition середовища IDEA забезпечує роботу з мовами програмування Java, PHP, Python.

Деякі середовища підтримують як режим інтерпретації, так і режим компіляції програм.

Далі в процесі опису мови програмування Python ми будемо застосовувати середовище IDLE.

Запитання для перевірки знань

1. Для чого призначено середовища програмування?
2. Назвіть складові середовища програмування.
3. Які функції виконує редактор тексту?
4. Поясніть сутність інтерпретації програм.
5. Які переваги та дефекти мають компілятори й інтерпретатори програм?
6. Для чого призначено редактор зв'язку?
7. Які основні дії виконуються в процесі налагодження програм?
8. Назвіть сучасні інтегровані середовища програмування.
9. Що називають об'єктним кодом?
10. Які особливості мають середовища об'єктно-орієнтованого програмування?
11. Для чого застосовують байт-код?
12. Які переваги мають інтегровані середовища програмування?

1.3. Основні можливості мови Python і структура проекту

Яку методику програмування підтримує мова, яку ви вивчали? Чи користувалися ви об'єктно-орієнтованими мовами програмування?

Матеріал нашого підручника зорієнтовано на роботу з мовою програмування Python*, яка підтримує об'єктно-орієнтований і процедурний методи програмування з інтерпретацією команд (інструкцій).

Мова Python підтримується всіма операційними системами і дозволяє розв'язувати складні математичні задачі, створювати графічні зображення, розробляти веб-сайти, працювати з реляційними базами даних.

Мова Python має потужну стандартну бібліотеку, яку користувач може розширювати власними бібліотеками й бібліотеками інших користувачів. Наприклад, розширення .NumPy містить реалізацію різноманітних математичних обчислень, модуль tkinter дає змогу реалізувати графічний інтерфейс користувача.

Програми можуть розроблятися в консольному режимі (такі програми мають розширення .py) і з графічним інтерфейсом (програми мають розширення .pyw).

Програма мовою Python — це звичайний текстовий файл, інструкції (команди) якого виконуються інтерпретатором для кожного рядка. Під час першого запуску програми створюється байт-код, який зберігається у файлі з розширенням .pyw. Якщо після цього програма не змінювалася, то в процесі наступних її запусків буде виконуватися байт-код.

Усі дані мови, у тому числі прості типи даних (числа, рядки) є об'єктами. У змінній зберігається не сам об'єкт, а посилання на нього, тобто адреса пам'яті, у якій зберігається об'єкт.

Структура проекту мовою Python складається з окремих модулів. Модуль — це будь-який файл із програмним кодом. Кількість таких модулів не обмежена. Один модуль може бути вкладений в інший модуль, тобто застосовується багатоієрархічна структура модулів. Модулі можуть групуватися в пакети.

Модулі можуть розроблятися самим програмістом, а можуть використовуватися вже наявні в стандартній бібліотеці мови. Один із модулів є *головним*, з нього запускається проект на виконання.

Щоб запустити один модуль з іншого, перший необхідно під'єднати до останнього (імпортувати).

* Мова програмування Python розповсюджується вільно за посиланням: http://python.org Наразі існують дві версії Python — 2.x (застаріла) та 3.x (перспективна), які не сумісні між собою. У цьому підручнику наведені приклади, розглянуті у версії 3.4.

Нідерландський програміст Гвідо ван Россум — щирий прихильник скетч-серіалу «Літаючий цирк Монті Пайтона» (англ. Monty Python's Flying Circus). На честь цієї програми він назвав створену ним мову програмування — **Python**.
Існують версії для Linux, Windows, MacOS.

Програму мовою **Python** можна створювати й редагувати за допомогою будь-якого редактора, наприклад, Notepad++, Eclipse, PythonWin та ін.

Мова **Python** підтримує динамічну типізацію даних. Це означає, що оголошувати типи даних не потрібно, мова самостійно слідкує та визначає їх тип на основі їх зовнішнього вигляду. Автоматично звільняється пам'ять для тих даних, які стають непотрібними.

На рис. 1 подано варіант структури проекту мовою Python.

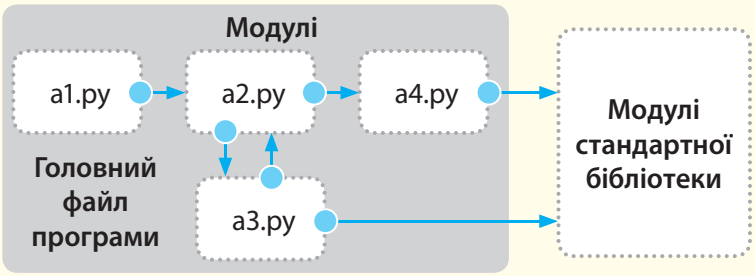


Рис. 1. Варіант структури проекту мовою Python

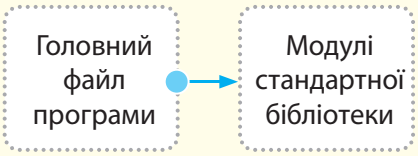


Рис. 2. Найпростіша структура програми мовою Python

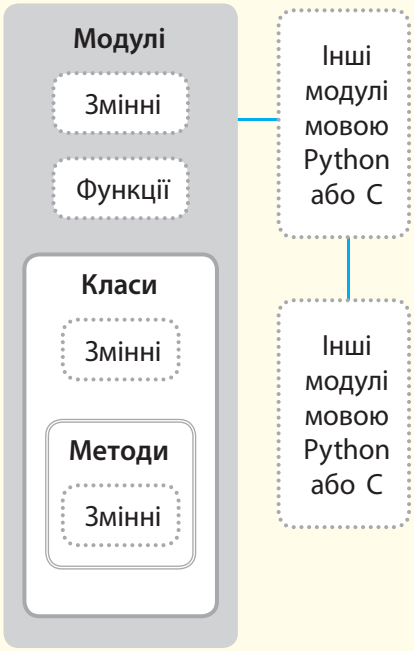


Рис. 3. Структура проекту модуля мовою Python

Оскільки ми розглядатимемо лише найпростіші навчальні проекти, то далі їх можемо називати просто програмами.

Наведена структура програми складається з чотирьох файлів: a1.py, a2.py, a3.py, a4.py, які розробляє програміст, а також модулів стандартної бібліотеки мови Python. Файл a1.py є головним, із нього запускається програма. Цей файл, так само як і інші три модулі, складається з інструкцій мови Python. Фактично це звичайні текстові файли, які нами будуть розроблятися в середовищі IDLE.

Файли a2.py, a3.py і a4.py самостійно не запускаються. Файл a2.py запускається з головного файла. Але для запуску його слід спочатку імпортувати у файл a1.py. Файли a3.py і a4.py запускаються з файла a2.py, але для цього їх також слід імпортувати у файл a2.py. Із файлів a3.py і a4.py запускаються модулі стандартної бібліотеки, які також потрібно імпортувати у ці файли.

Із рис. 1 видно, що мова Python реалізує багатоієрархічну структуру вкладеності модулів. Але в підручнику буде застосовуватися найпростіша архітектура програми, яка містить лише головний файл, та програму, яка містить головний файл і модулі стандартної бібліотеки (рис. 2).

У такій архітектурі модулі стандартної бібліотеки імпортуються безпосередньо в головний файл програми. Реально програмний модуль може складатися не тільки з інструкцій мови Python, а й зі змінних, функцій і класів (рис. 3).

В ООП мовою Python використовуються класи і методи (основи описано нижче). Із рис. 3 видно, що модуль може імпортувати інші модулі, які написано не тільки мовою Python, але й мовою C. Він також може бути імпортований в інші модулі.

Запитання для перевірки знань

1. Якими операційними системами підтримується мова Python?
2. Яке розширення мають файли програм, які створено в консольному режимі?
3. Назвіть основні переваги мови **Python**.
4. Який тип трансляції застосовується у **Python**?
5. Із якими мовами може інтегруватися мова Python?
6. Що називають динамічною типізацією даних?
7. Поясніть структуру проекту мовою Python.
8. Яку структуру може мати модуль мовою Python?

1.4. Режими виконання програмного коду в середовищі IDLE

У середовищі IDLE програмний код мовою Python можна виконувати в *інтерактивному режимі* та *режимі виконання файлів програм*.

1.4.1. Виконання програмного коду в інтерактивному режимі

Пригадайте, який вид трансляції програми використовувався вами у 8 і 9 класах. Які, на вашу думку, його переваги й недоліки?

В *інтерактивному режимі* результати виконання інструкцій користувача виводяться одразу після їх уведення. Тобто вводиться перша інструкція, яка одразу виконується, потім уводиться друга інструкція і т. д.

Розглянемо способи запуску інтерактивного режиму роботи інтерпретатора IDLE.

- За допомогою командного рядка ОС Windows (щоб його відкрити, слід клацнути правою кнопкою миші кнопку Пуск і виконати команду Командний рядок). До командного рядка слід увести команду python.

  Відкриється вікно, яке зображене на рис. 1.

Інтерактивний режим доцільно використовувати на етапі вивчення синтаксису мови, коли необхідно переконатися в правильності виконання окремих інструкцій після їх уведення. А також для тестування коду, що зберігається у файлах.

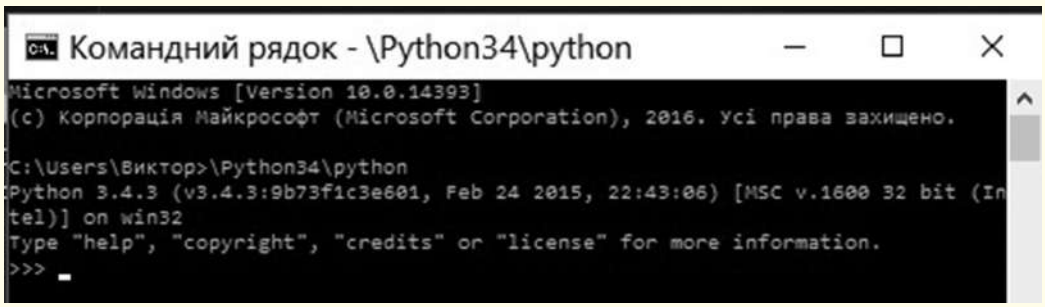


Рис. 1. Вікно інтерпретатора, яке відкрите за допомогою командного рядка, версія Python3.4

- Із робочого стола. Якщо ярлик мови Python розміщено на робочому столі, то слід клацнути цей ярлик.
- Із головного вікна середовища програмування IDLE. Для цього слід виконати команди:

  Пуск → Усі програми → Python3.x* → IDLE (Python3.xGUI — 32 біт). Відкриється вікно, яке зображене на рис. 2.

  Саме цей спосіб запуску застосовуватиметься нами далі.

Розглянемо порядок виконання найпростішої програми мовою Python в інтерактивному режимі (рис. 2).

Мова Python розповсюджується вільно на підставі ліцензії GNU General Public License.

* Замість «x» укажіть номер версії Python, що встановлено на вашому комп'ютері — 3.4 або 3.6.

PEP8 — стиль написання програм мовою Python. Цей документ описує угоду про те, як писати код для мови Python. Це список рекомендацій, яких добровільно дотримуються Python програмісти.

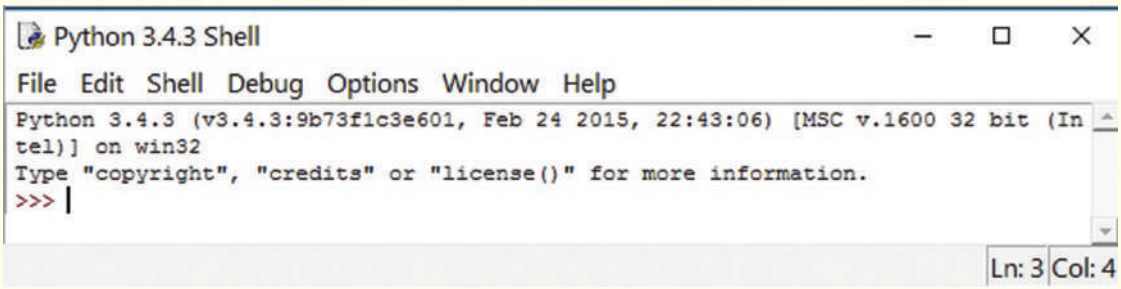


Рис. 2. Вікно інтерпретатора, яке відкрито в середовищі IDLE

Приклад

Припустимо, що програма повинна містити повідомлення про стилі програмування та побажання успіхів. Цю програму можна подати в такому вигляді:

```
#Починаємо вивчати мову Python
print ("Python підтримує ООП
і процедурний стиль")                    #повідомлення
print ("Найбільших успіхів")             #побажання
```

Тут знак **#** — це коментар, який жодним чином не впливає на роботу інтерпретатора (все, що пишеться після нього, не виконується). Він може бути розташований як на початку рядка програми, так і наприкінці кожної команди.

Команда **print** забезпечує виведення на екран повідомлення, що міститься в круглих дужках у подвійних лапках.

Хід виконання

1. Для введення програми запускаємо інтерпретатор мови Python середовища IDLE.
2. Одразу після знака запрошення (>>>) уводимо перший рядок програми (#Починаємо вивчати мову Python) і натискаємо клавішу Enter.
3. Знак запрошення з'явиться на наступному рядку, після якого уводимо команду другого рядка програми (print ("Python підтримує ООП і процедурний стиль") #повідомлення і натискаємо клавішу Enter.
4. Система виведе повідомлення "Python підтримує ООП і процедурний стиль", а в наступному рядку з'явиться знак запрошення, після якого уводимо print ("Найбільших успіхів") #побажання.

Динаміку описаного процесу введення та виконання програми в інтерактивному режимі зображено на рис. 3.

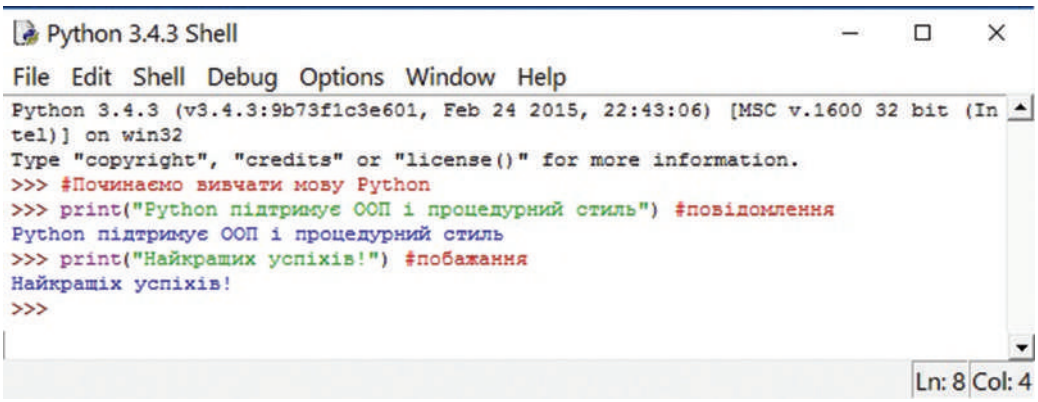


Рис. 3. Перша програма мовою Python (результат роботи)

Після знаку запрошення (>>>) не повинно бути пробілів (але є невеличкий відступ для кращого сприйняття), інакше видаватиметься повідомлення про синтаксичну помилку.

Команди відокремлюються крапкою з комою (;), якщо в одному рядку уводяться дві або більше команд.

Кожну команду рекомендується вводити в окремому рядку, але команди можна розміщувати й у кількох рядках. Наприклад, команду x=130+82+45 можна розмістити у два рядки за такими варіантами:

```
x = 130 + 82 \
+ 45
```

або

```
x = (130 + 82
+ 45)
```

У мові Python відсутні спеціальні засоби для виділення блоків, наприклад фігурні дужки або конструкція begin…end. Якщо команда містить блок, то кожна команда блока мовою Python відокремлюється від початку рядка спеціальним відступом чотирма пробілами. Якщо кількість пробілів різна, інтерпретатор видає повідомлення про фатальну помилку.

Щоб завершити роботу інтерактивного режиму, необхідно натиснути сполучення клавіш Ctrl+Z.

Після того як Гвідо ван Россум розробив мову Python (приблизно у 1991 році), він виклав її в Інтернет. Мова Python сподобалася програмістам і почала вільно поширюватися.

Таким чином, до розробки приєдналося вже співтовариство програмістів.

У середньому кожні два чи 2,5 роки з'являється нова версія мови.

Запитання для перевірки знань

1. Для чого застосовується коментар у програмному коді?
2. Яке позначення має знак запрошення в інтерактивному режимі?
3. Як завершується робота інтерпретатора в інтерактивному режимі?
4. Які переваги має інтерактивний режим виконання програмного коду?
5. Як можна запустити інтерактивний режим інтерпретатора IDLE?
6. Поясніть сутність інтерактивного режиму.
7. Як правильно увести кілька команд в одному рядку?
8. Як оформлюється блок команд у мові Python?

Завдання для самостійного виконання

1. Запустіть інтерактивний режим роботи IDLE за допомогою командного рядка ОС Windows.
2. Запустіть інтерактивний режим інтерпретатора IDLE з головного вікна середовища програмування IDLE.
3. Розробіть і виконайте програмний код в інтерактивному режимі, який виводить на екран два речення:

   У мові Python застосовується байт-код.

   Python забезпечує високу компактність і наочність програмного коду.
4. Розробіть і виконайте програмний код, який в інтерактивному режимі виводить таке повідомлення: "Python підтримує процедурні і об'єктно-орієнтовні методи програмування".
5. Розробіть і виконайте програмний код в інтерактивному режимі для виведення повідомлення: "Python має високий рейтинг" із використанням коментаря.
6. Обчисліть значення виразу p = 123 / 4 + 45 * 3 – 66 * (3 + 87) в інтерактивному режимі.

1.4.2. Виконання файлів програмного коду

Інтерактивний режим роботи інтерпретатора забезпечує простоту і наочність у роботі програміста. Чому не можна обмежитися лише цим режимом?

Способи запуску файла програмного коду на виконання:
- із командного рядка середовища програмування;
- клацанням ярлика файла;
- за допомогою користувацького інтерфейсу середовища IDLE.

Інтерактивний режим роботи є досить зручним. Але програмний код, який уводиться в інтерактивному режимі, не зберігається: він зникає одразу після того, як інтерпретатор мови Python його виконає. Для повторного виконання програмного коду потрібно увести код заново, що є суттєвим недоліком інтерактивного режиму.

Програмні коди необхідно зберігати у файлах, які, як раніше зазначалося, називають модулями. Файли модулів — це програми, які називають також сценаріями, хоча між ними є невелика різниця. Ім'я файла може не містити розширення. Але якщо передбачається імпортувати програмний код з інших файлів, то ім'я файла має містити розширення .py.

Файл програмного коду інтерпретатор може виконувати необмежену кількість разів.

Після запуску коду на виконання інтерпретатор виконує послідовно одну інструкцію за іншою в порядку їх розташування й видає результат на екран монітора. Якщо в програмному коді буде виявлено синтаксичну помилку, відповідне повідомлення виводиться на екран.

Розглянемо порядок створення файла найпростішого програмного коду на прикладі.

Приклад

Припустимо, що потрібно створити файл із кодом, який виконує складання чисел 65 і 279 та множення числа 21 на число 15.

1. Запускаємо середовище програмування IDLE (Python3.x 32-bit). У відкритому вікні Python3.x Shell виконуємо команду File → New File.

2. У вікно Untitled, яке відкрилося, уводимо програму так, як зображено на рис. 1.

Останньою в програмі є команда input(). Наразі ця команда відіграє допоміжну роль. За її допомогою результат виконання програми буде відображатися на екрані доти, доки не буде натиснуто клавішу Enter.

Можливо, що без цієї команди результат відображатиметься досить короткий час, і ми не зможемо його побачити.

```
#Складання і множення чисел
print("65+279=", 65 + 279)
print("21*15=", 21 * 15)
print("Програма завершена")
input()
```

Рис. 1. Програма додавання і множення двох чисел

3. Зберігаємо програму за допомогою команди File → Save As... У результаті відкриється вікно Збереження файлу (рис. 2).

Звертаємо увагу на те, що за замовчуванням пропонується зберегти файл у папці Python3 диску C*. Далі всі файли програм зберігатимемо саме в ній. Інакше необхідно змінити ім'я папки та місце її розташування.

Уводимо ім'я prog_02 (без розширення .py, оскільки на цьому етапі вивчення мови Python не передбачає імпортування файлів), і натискаємо кнопку Зберегти.

Файл буде збережено.

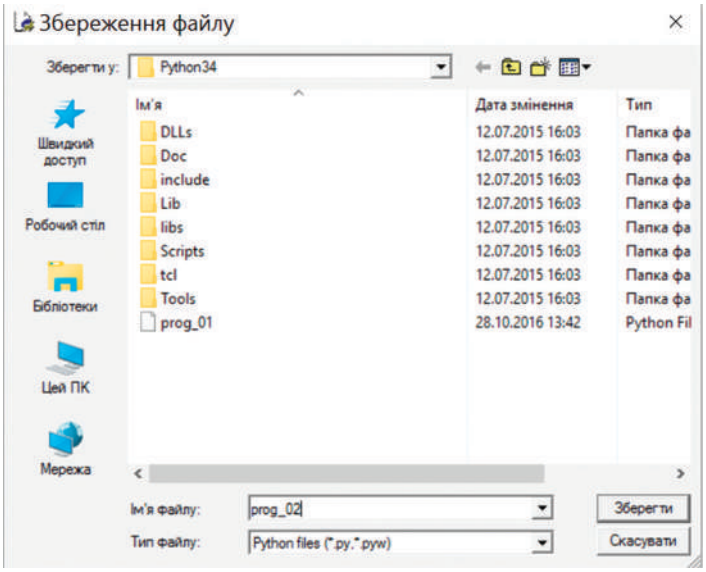


Рис. 2. Вікно **Збереження файлу**

Виконувати програму можна різними способами.

- Якщо програму відкрито в додатковому вікні Python3.4.3 Shell, то достатньо виконати команди Run → Run Module (або натиснути клавішу F5). Результат виконання програми зображено на рис. 3.

Далі будемо користуватися командою Run → Run Module.

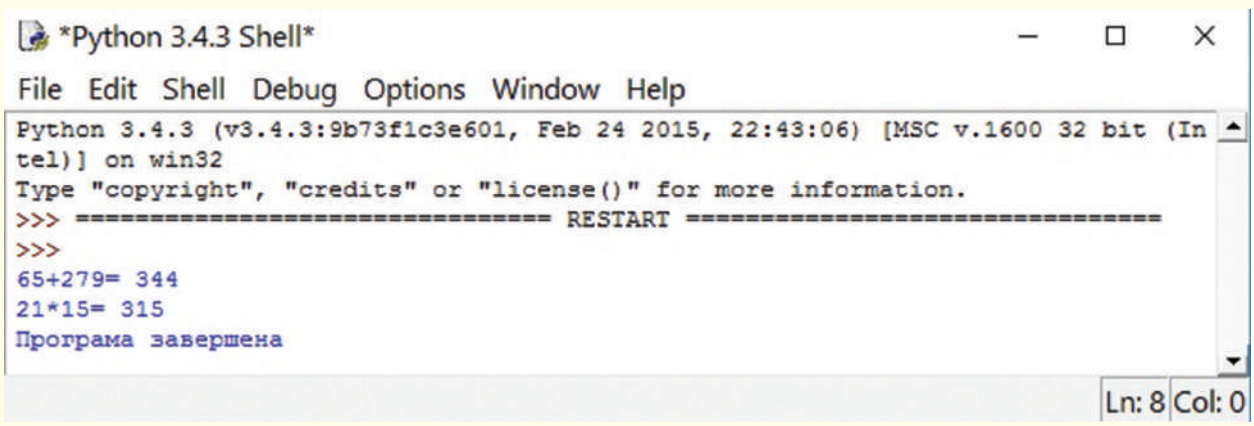


Рис. 3. Результат виконання програми додавання і множення чисел

- Якщо програму створено раніше, її можна** викликати у вікно Python3.4.3 Shell, виконавши команду File → → Open...

Виконати програму, створену раніше, можна також подвійним клацанням імені відповідного файла у папці

Нині **Python** — одна з найпопулярніших мов програмування, яка охоплює нові сфери застосування. Останні 5 років входить до п'ятірки найзатребуваніших технологій.

* Цей варіант збереження пропонується за замовчуванням.

** Відкрити в середовищі Python3.x IDLE у вікні Python3.x Shell.

Мову **Python** відрізняє швидкість і простота скриптів. Разом із набором доступних бібліотек для роботи з мережами і файлами це робить її незамінним помічником системного адміністратора.

C:\Python34. Але в цьому випадку останньою командою програми обов'язково повинна бути input(). Інакше результат її виконання можна не побачити.

Зазначимо, що за замовчуванням у Python 3 для кодування команд здійснюється код UTF-8, який буде використовуватися далі.

Для застосування іншого коду необхідно в першому або другому рядку програми вказати назву коду за допомогою інструкції: # –*– coding: <код> –*–.

Наприклад, для коду Windows 1251 інструкція матиме такий зміст: # –*– coding: cp1251 –*–.

Запитання для перевірки знань

1. Як можна запустити на виконання файл програмного коду?
2. Як можна викликати на екран файл програмного коду?
3. Наведіть приклад найпростішого програмного коду мовою Python.
4. Як зберегти файл програмного коду?
5. Які вади має інтерактивний режим інтерпретатора?
6. Чому доцільно зберігати програмний код у файлі?
7. Для чого в програмному коді останньою зазначається команда input()?

Завдання для самостійного виконання

1. Складіть програму для обчислення значення виразу 2.3 + 106 * 4 і збережіть її у файлі з іменем f_01. Виконайте програму і переконайтеся, що отриманий результат є правильним.
2. Завантажте середовище IDLE, викличте створену в пункті 1 програму з файла f_01, замініть у програмі операцію множення числа 106 на 4 на операцію ділення. Збережіть програму в тому самому файлі. Виконайте програму та переконайтеся, що отриманий результат є правильним.
3. Складіть програму для обчислення площі трикутника за значеннями його сторін. Збережіть програму у файлі f_02. Виконайте програму та доведіть, що отриманий результат є правильним.
4. Викличте програму, що зберігається у файлі f_02. Внесіть до неї такі зміни, щоб обчислювалася площа прямокутного трикутника за значеннями його катетів.
5. У коло радіусом r вписано правильний трикутник. Розробіть програму визначення площі кола, яку не зайнято трикутником. Збережіть програму у файлі f_03. Виконайте програму та доведіть, що вона функціонує правильно.
6. У правильний трикутник зі стороною a вписано коло. Розробіть програму визначення площі трикутника, яку не зайнято колом. Збережіть програму у файлі f_04. Виконайте програму та доведіть, що вона функціонує правильно.
7. Дізнайтеся в Інтернеті про відстань по шосе між містами Хмельницький і Вінниця. Розробіть проект визначення орієнтовного часу прибуття автобуса до Вінниці, який починає рух із Хмельницького о 8.30.
8. Знайдіть в Інтернеті відомості про найнижчу та найвищу температуру в Києві у лютому за останні 5 років. Розробіть проект визначення різниці між цими показниками.

2. Оператори, вирази і засоби опрацювання чисел

Мова Python має багато вбудованих типів даних, які буде описано в наступних розділах. У цьому розділі наводиться лише їх перелік, для того щоб свідомо розуміти необхідність перетворення одного типу даних на інший у процесі програмування арифметичних та інших типів виразів.

2.1. Основні елементи мови Python

Пригадайте елементи мови програмування, яку ви вивчали. Чи всі символи клавіатури комп'ютера ви використовували в процесі введення програм?

Будь-яка конструкція мови програмування створюється з алфавіту. Алфавіт мови Python містить символи, що розташовані на клавіатурі комп'ютера (рис. 1).

Із символів алфавіту створюються лексеми (token).

Лексема — це мінімальна одиниця мови, яка має певне самостійне значення і яку розуміє транслятор. Якщо транслятор її не розуміє, то видається повідомлення про помилку в програмі. За замовчуванням символи кодуються в системі UTF-8.

Розрізняють такі види лексем:

- ключові (зарезервовані) слова (keywords);
- ідентифікатори (identifiers);
- літерали (константи);
- операції;
- знаки пунктуації.

У мові Python використовується кілька десятків ключових (зарезервованих) слів (наприклад, int, list, input, print, float тощо). З їх призначенням будемо знайомитися поступово, у міру потреби в їх застосуванні.

Ідентифікатори (імена) використовуються для позначення змінних, функцій, які створює програміст, та інших об'єктів. Ідентифікатор змінної може складатися з латинських літер, цифр і знака підкреслення. Першим символом імені не може бути цифра або знак підкреслення.

У редакторі IDLE правильні імена змінних висвічуються чорним кольором. Якщо ім'я змінної відображається іншим кольором, його необхідно замінити. Однакові імена з літерами на різних регістрах сприймаються як різні імена. Наприклад, ідентифікатори ster і Ster є різними іменами.

Імена змінних використовуються для доступу до даних. Дані в мові Python подано у формі об'єктів. Тобто *об'єкт* — це ділянка пам'яті з певним значенням і можливими операціями його опрацювання. Кожний об'єкт має свій тип, наприклад int(ціле число), str(рядок) та ін.

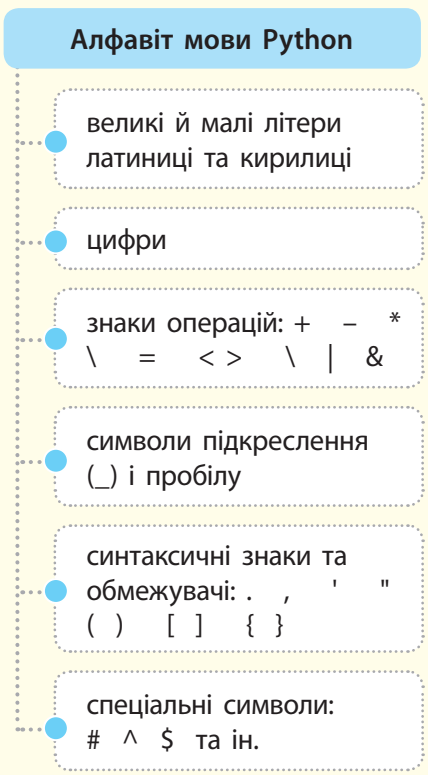


Рис. 1. Алфавіт мови Python

Ключові слова мови, назви функцій та вбудованих у мову об'єктів забороняється використовувати як імена змінних. Наприклад, ідентифікатори змінних dush, cosm_1 і Ftr_2 є правильними, а ідентифікатори 2beta, rnt-sd, _fok, int — неправильними.

Python використовують багато відомих кампаній у всьому світі. Серед них Google, Facebook, Yahoo, NASA, Dropbox, IBM, Red Hat, Pinterest.

У мові Python існують базові об'єктні типи (вбудовані в мову) і ті, що розробляються користувачем засобами самої мови або іншими засобами.

Зазначимо, що змінні зберігають не сам об'єкт, а посилання на об'єкт, тобто адресу об'єкта в пам'яті комп'ютера.

Об'єкт може бути літералом (константою). У табл. 1 наведено найчастіше вживані базові типи об'єктів та приклади їх літералів. Детальніше вони описані в розділі 4.

Таблиця 1. Найчастіше вживані вбудовані типи об'єктів

Тип об'єкта	Приклад літерала
Цілі числа (int)	45, 365
Дійсні числа (float)	36.5, 213.75
Рядки (str)	'Python', "процесор"
Логічні дані (bool)	true, false
Списки (list)	[40, [3, 'монітор'], 32]
Словники (dict)	{'айфон': 'сон', 'doc': 'nm'}
Кортежі (tuple)	(5, 'sp', 31, 'M')
Множини (set)	set('abc'), {'a', 'b', 'c'}

Приклад 1.

```
>>> x_1 = 87          #змінна x_1 має тип int і значення 87
>>> x_2 = 21.45       #змінна x_2 має тип float і значення 21.45
>>> s_1 = 'проект'    #змінна s_1 має тип str і значення «проект»
>>> s_2 = "байт"      #змінна s_2 має тип str і значення «байт»
```

Як уже зазначалося, у мові Python застосовується динамічна типізація змінних. Це означає, що не потрібно оголошувати типи змінних, як це робиться в багатьох мовах програмування, оскільки їхній тип визначається автоматично в процесі присвоювання їм значень. Цей тип визначається значенням, розташованим праворуч від оператора присвоювання, який позначається знаком (=) (приклад 1).

В одному рядку можна присвоїти однакові значення кількох змінним, наприклад:

```
>>> x = y = z = 441   #змінні x, y, z мають тип int і значення 441
```

Ще раз зазначимо, що після виконання оператора присвоювання в змінній зберігається не сам об'єкт, а лише посилання на нього, тобто адреса ділянки пам'яті, у якій зберігається об'єкт. Тому слід бути досить уважним під час запису групових операцій. Розглянемо такий фрагмент програми (приклад 2).

Приклад 2.

```
>>> x = y = [7, 15]  #створюється ніби два об'єкти
>>> x, y             #виведення значень x і y
([7,15], [7,15])     #двічі виводяться значення одного об'єкта
```

Із прикладу 2 видно, що створюється ніби два об'єкти (x і y) типу list, але вони мають одну й ту саму адресу пам'яті, тобто реально створюється один об'єкт, значення якого виводиться двічі. Для підтвердження цього змінимо значення нульового елемента об'єкта y (нумерація елементів у списку починається з нуля) і перевіримо значення об'єктів (приклад 3).

Приклад 3.

```
>>> y[0] = 50        #зміна значення нульового елемента об'єкта y
>>> x, y             #виведення значень x і y
([50, 15], [50, 15])  #значення x і y однакові, хоча x не змінювали
```

Із прикладу 3 видно, що ми змінили тільки значення y[0], а фактично об'єкти мають однакові значення, оскільки реально і x і y мають однакову адресу, тобто є одним об'єктом.

Щоб отримати два об'єкти x і y, необхідно виконати присвоювання окремо для кожного з них, наприклад:

```
>>> x = [21, 807]
>>> y = [21, 807]
```

Для перевірки, чи посилаються дві змінні на один і той самий об'єкт, використовується оператор is. Якщо змінні посилаються на один об'єкт, то оператор повертає значення True, інакше — значення False (приклад 4).

Одним оператором присвоювання можна присвоїти різні значення кільком змінним.

У такому разі змінні та значення відокремлюються комою, наприклад:

```
>>> x_1, x_2, x_3 = 13, 105, 27
>>> x_1, x_2, x_3
(13, 105, 27)
```

Кількість елементів ліворуч і праворуч від оператора присвоювання має бути однакова, інакше буде видано повідомлення про синтаксичну помилку.

Наприклад, помилку буде видано для такої інструкції:

```
>>> x, y, z = (2, 44, 50, 65)
```

Позбавитися цього явища можна за допомогою символу «зірочка» (*), який розміщується перед однією зі змінних. У такому разі ця змінна містить список з усіх зайвих значень (приклад 5).

Приклад 4.

```
>>> x = y = [60, 90]   #змінні x і y посилаються на один об'єкт
>>> x is y
True
>>> x = [23, 20]       #змінні x і y посилаються на різні об'єкти
>>> y = [23, 20]
>>> x is y
False
```

Приклад 5.

```
>>> x_1, x_2, *x_3 = (46, 7, 21, 14)
#змінна x_3 набуде значень 21 і 14
>>> x_1, x_2, x_3
(46, 7, [21, 14])
>>> x_1, *x_2, x_3 = (4, 7, 33, 17)
#змінна x_2 набуде значень 7 і 33
>>> x_1, x_2, x_3
(4, [7, 33], 17)
```

Запитання для перевірки знань

1. Яким ідентифікатором позначаються цілі числа?
2. Яким ідентифікатором позначаються дійсні числа?
3. Які базові типи даних використовуються найчастіше?
4. Сформулюйте правило запису ідентифікаторів змінних.
5. Що зберігається в змінних?
6. Поясніть сутність динамічної типізації змінних.
7. Для чого призначений оператор is?
8. Поясніть сутність поняття «посилання на об'єкт».
9. Як можна одним оператором присвоїти різні значення кільком об'єктам?
10. Що називають лексемою?
11. Які існують види лексем?

Завдання для самостійного виконання

1. Виявіть помилки в записі ідентифікаторів змінних: ac_1, 2ba, x–1, –z_1.
2. Змінним x, y, z присвойте одним оператором присвоювання, відповідно, такі значення: 3, 37, 45.
3. Доведіть, що після виконання операторів:
   ```
   >>> a1 = [5, 20]
   >>> a2 = [5, 20]
   ```
   змінні будуть посилатися на різні об'єкти.
4. Визначте значення змінних a_1, a_2 і a_3 після виконання оператора:
   ```
   >>> a_1, *a_2, a_3 = (1, 6, 2, 3, 4, 5)
   ```
   і доведіть, що отриманий результат є правильним.
5. Визначте, як реагуватиме середовище програмування на спробу виконати оператор:
   ```
   >>> a, b, c, d = (100, 61, 55).
   ```
6. Одним оператором присвойте змінним такі значення: x = 33, 21, 2; y = 7; z = 66.

2.2. Поняття про перетворення типів даних

Чому, на вашу думку, виникає потреба в перетворенні типів даних? Чи доводилося вам стикатися з цією проблемою під час розв'язування математичних задач?

Одна й та сама змінна в процесі виконання коду може посилатися на об'єкти з різними типами даних. Наприклад, після виконання двох інструкцій:

```
>>> x_1 = "принтер"        #змінна x_1 посилається на тип str
>>> x_1 = 3.8              #тепер змінна x_1 посилається на тип float
```

Змінна x_1 спочатку посилається на тип str, потім — на тип float.

Для визначення останнього типу даних, на який посилається змінна, призначена функція type (ім'я змінної) (приклад 1).

Отже, тип даних — це характеристика об'єкта, а не змінної. Змінна містить тільки посилання на об'єкт.

Для кожного конкретного типу даних існує строго визначений набір операцій, які можуть виконуватися над ним.

Наприклад, для даних типу int і float можна виконувати арифметичні операції. Спроба виконати, наприклад, операцію додавання цілого числа та рядка:

```
>>> 43 + "25"
```

приведе до виведення повідомлення про синтаксичну помилку.

Для перетворення одного типу даних на інший у мові Python застосовуються спеціальні функції.

Розглянемо основні функції перетворення одного типу даних на інший.

bool([об'єкт]) — перетворення об'єкта на логічний тип (приклад 2).

Із прикладу видно, що функція bool повертає значення False у випадку, якщо об'єкт дорівнює нулю або порожній, інакше — значення True.

int([об'єкт [,<система числення>]]) — перетворення об'єкта на ціле число. Система числення, у якій подається об'єкт, може бути десятковою, вісімковою, шістнадцятковою. За замовчуванням — десяткова система (приклад 3).

float(ціле число або рядок) — перетворення цілого числа або рядка на число дійсного типу (приклад 4).

str(об'єкт) — перетворення об'єкта на рядок (приклад 5).

list(послідовність) — перетворення елементів послідовності на список (приклад 6).

Приклад 1.

```
>>> x_2 = 300
>>> type(x_2)      #визначення типу змінної x_2
<class ‹int›>      #змінна x_2 посилається на тип int
```

Приклад 2.

```
>>> bool(0), bool(1), bool([4,6]), bool(""), bool("файл")
(False, True, True, False, True)
```

Приклад 3.

```
>>> int(13.5), int("71"), int("A", 16), int("71", 8)
(13, 71, 10, 57)
```

(пояснення: значення функції int("71", 8) має двійкове значення 111001, що еквівалентно 57)

Приклад 4.

```
>>> float(25), float("13.75")
(25.0, 13.75)
```

Приклад 5.

```
>>> str(149), str([21, 16, 35]), str({"x" :10})
('149', '[21, 16, 35]', "{'x'10}")
```

Приклад 6.

```
>>> list("2468")          #перетворення рядка
['2', '4', '6', '8']      #список
>>> list((2,4,6,8))       #перетворення кортежу
[2, 4, 6, 8]              #список
```

На рис. 1 зображено найпростішу програму додавання двох чисел із перетворенням типу str на тип int, яку збережено з іменем prog_03.

```
# Програма складання чисел з перетворенням типу str в тип int
x = int(input("x= ")) # увести, наприклад, 26
y = int(input("y= ")) # увести, наприклад, 9
print(x + y)          # виведення суми чисел
input()               # очикування натиснення Enter
```

Рис. 1. Програма додавання чисел із перетворенням типу str на тип int

За замовчуванням ці змінні мають тип str, тому їм потрібно встановити відповідний тип.

Після запуску програми на екрані з'явиться запрошення на введення значення змінної *x*. Уведемо, наприклад, число 26 і натиснемо клавішу Enter. З'явиться таке саме запрошення для введення значення змінної *y*. Уведемо, наприклад, число 9 і натиснемо Enter. У результаті отримаємо результат 35.

Після того як змінні в програмі вже не потрібні, їх можна *видалити за допомогою функції* del. Спроба використати змінні *x* або *y* після виконання такого фрагмента (приклад 7) призведе до видачі повідомлення про помилку, оскільки змінну x видалено (автоматичним збирачем сміття в Python).

Приклад 7.

```
>>> x = 10; x; y = 20; y
10
20
>>> del x, y
>>> x
```

Запитання для перевірки знань

1. Яка функція призначена для визначення типу даних?
2. Які операції можуть виконуватися над даними типу int?
3. Для чого призначено функцію bool?
4. На скільки типів даних може посилатися змінна в програмі?
5. Як можна видалити з програми непотрібні імена змінних?
6. Який результат буде отримано після виконання інструкції >>> int("D", 16)?
7. Змініть інструкцію >>> 77 * '89' так, щоб її було правильно виконано.

Завдання для самостійного виконання

1. Визначте тип змінної sk_1 після виконання двох інструкцій:
   >>> s = "Python"
   >>> s = 21
2. Визначте, який буде отримано результат після виконання інструкцій:
   >>> int(41.5), int("60"), int("F", 16)
3. Визначте результат виконання інструкції:
   >>> list("40, 22, 5, 66")
   Доведіть, що в інструкції немає помилок.
4. Запишіть і виконайте інструкцію перетворення вісімкового числа 47 на десяткове. Доведіть, що інструкцію виконано правильно.
5. Запишіть і виконайте інструкцію перетворення слова «процесор» на логічний тип. Доведіть, що інструкцію виконано правильно.
6. Складіть і виконайте програму додавання чисел 37 і 29.7. Доведіть, що програму виконано правильно.

2.3. Оператори і вирази

Пригадайте, які операції над числами використовували ви в процесі програмування у 8 і 9 класах. Які для цього застосовувалися оператори?

Операції над об'єктами виконуються за допомогою відповідних операторів. Об'єкти, над якими виконуються операції, називають операндами. Кожний оператор може виконувати операції над строго визначеними для нього типами операндів.

Залежно від типу об'єктів, над якими виконуються операції, оператори групуються в *арифметичні, логічні, порівняння, присвоювання* й ін.

До арифметичних операторів належать такі загальновідомі оператори:

- додавання (+)
- віднімання (–)
- множення (*)
- ділення (/)

Арифметичні оператори призначені для виконання операцій над числами. Якщо операція виконується над цілим і дійсним числами, то ціле число буде спочатку перетворене на дійсний тип, а потім виконуватиметься операція над дійсними числами. Результатом операції в цьому випадку буде число дійсного типу. Результатом операції ділення завжди буде число дійсного типу.

Окрім звичайних арифметичних операцій, у мові Python також застосовуються такі операції (табл. 1).

Таблиця 1. Деякі операції у мові Python

Операція	Позначка	Приклад
Цілочислове ділення (без остачі)	//	У результаті виконання операції 10.0 // 3.0 отримаємо результат 3.0
Ділення за модулем (остача від ділення)	%	У результаті виконання операції 10 % 3.0 отримаємо результат 1.0
Піднесення до степеня	**	У результаті виконання операції 10 ** 2 отримаємо результат 100

У мові Python використовуються арифметичні оператори з присвоюванням (табл. 2).

Таблиця 2. Арифметичні оператори з присвоюванням

Оператор	Позначка	Приклад
Збільшення значення змінної на вказану величину	+=	x += 8 (еквівалентно x = x + 8)
Зменшення значення змінної на вказану величину	–=	x –= 8 (еквівалентно x = x – 8)
Множення значення змінної на вказану величину	*=	x *= 8 (еквівалентно x = x * 8)
Ділення значення змінної на вказану величину	/=	x /= 8 (еквівалентно x = x / 8)

Один із варіантів імпортування можна реалізувати за допомогою інструкції: **from decimal import Decimal**. Наприклад, можна виконати такі інструкції:

```
>>> from decimal import Decimal          #імпортування
>>> Decimal("1.123") – Decimal("0.5")    #виконання операції
Decimal('0.623')                         #результат
```

Для точного виконання операцій над числами в мові **Python** призначений модуль **decimal**, у якому є функція **Decimal**.

Цей модуль слід імпортувати до коду програми. Можна імпортувати модуль або його частину, наприклад, окрему функцію.

Оператори порівняння порівнюють значення об'єкта, який розташовано ліворуч від оператора, зі значенням об'єкта, який розташовано праворуч від цього оператора. Якщо умова виконується, повертається значення True, інакше — False.

Склад, позначення та приклади використання операторів наведено в табл. 3.

Таблиця 3. Оператори порівняння

Позначення	Назва	Пояснення	Приклад
==	Дорівнює	Якщо значення операндів однакові, повертається значення True, інакше — False	>>> 23 == 40 False
!=	Не дорівнює	Якщо значення операндів не однакові, повертається значення True, інакше — False	>>> != 13 True
>	Більше	Якщо значення операнда зліва більше за значення справа, повертається значення True, інакше — False	>>> 5 > 204 False
<	Менше	Якщо значення операнда зліва менше за значення справа, повертається значення True, інакше — False	>>> 5 < 204 True
>=	Більше або дорівнює	Якщо значення операнда зліва більше або дорівнює правому, повертається значення True, інакше — False	>>> 17 >= 17 True
<=	Менше або дорівнює	Якщо значення операнда зліва менше або дорівнює правому, повертається значення True, інакше — False	>>> 43 <= 17 False

У мові Python використовуються такі логічні оператори: not (ні), or (або), and (і). Вони виконуються над даними логічного типу, які мають два значення: True (істинне) і False (хибне). Результати виконання цих операторів наведені в табл. 4.

Таблиця 4. Результати виконання логічних операторів

X	Y	not x	x or y	x and y
False	False	True	False	False
True	False	False	True	False
False	True	True	True	False
True	True	False	True	True

Із табл. 4 видно, що оператор not є унарним, тобто діє лише для одного операнда. Результатом його виконання є значення, протилежне значенню операнда x. Результатом виконання оператора or є значення False лише в тому випадку, коли обидва

операнди мають значення False. У всіх інших випадках отримується значення True.

Результатом виконання оператора and є значення True лише у випадку, коли обидва операнди мають значення True. У всіх інших випадках отримується значення False.

Оператори над послідовностями виконують операції над списками, кортежами та рядками.

Далі роз'яснено сутність та наведено приклади виконання операцій над рядками (табл. 5) (операції над іншими типами послідовностей описано в розділі 4).

Python входить до трійки найбільш значущих мов у сфері машинного навчання й аналізу великих обсягів даних. Як універсальна мова **Python** має застосування практично скрізь, навіть в ігровій індустрії.

Таблиця 5. Операції над рядками

Операція	Позначення	Приклад
Об'єднання (призначене для з'єднання послідовностей)	+	У результаті виконання операції над рядками "послі" + "довність" отримаємо результат "послідовність"
Повторення (призначене для повторення послідовності задану кількість разів)	*	>>> "ав" * 5 'авававававав'

Перевірка на входження послідовностей одна до іншої	in	Якщо одна послідовність входить до іншої, видається значення True, інакше — False. Приклад:
		>>> "виконання" in "Результатом виконання операції"
		True
		>>> "оператор" in "Результатом виконання операції"
		False

Перевірка на невходження однієї послідовності до іншої	not in	Якщо одна послідовність входить до іншої, видається повідомлення False, інакше — True. Приклад:
		>>> "байт" not in "біт"
		True
		>>> "байт" not in "байт і біт"
		False

Над рядками можуть виконуватися також дві операції з присвоюванням (табл. 6).

Таблиця 6. Операції з присвоюванням

Операція	Позначення	Приклад
Об'єднання з присвоюванням	+=	>>> s = "арифме"; s += "тичні"; print(s) арифметичні
Повторення задану кількість разів	*=	>>> s = "пр"; s *= 6; s 'прпрпрпрпрпр'

Python дуже популярна для написання серверної частини веб-сайтів для мобільних і веб-застосунків та побудови різноманітних сервісів.

Оператори використовуються у виразах. Поняття виразу в програмуванні аналогічне поняттю виразу в математиці.

Вираз мовою програмування складається з операндів, операторів і круглих дужок та визначає порядок виконання операцій над даними. Операнди виразу — це змінні, константи, функції, методи. Найпростіший вираз складається з одного операнда, наприклад, константи або змінної.

Залежно від типу операндів і операцій, що використовуються у виразі, розрізняють вирази:
- *арифметичні* (результат арифметичного типу);
- *логічні* (результат логічного типу);
- *рядкові* (результат рядкового типу).

Для кожного типу операцій існують чіткі правила їх запису та виконання, з якими ми будемо знайомитися поступово в процесі їх використання.

Наведемо основні правила для арифметичних виразів.

Для мови **Python** існує велика кількість бібліотек для наукових досліджень, серед яких NumPy, ScPy, Matplotlib.

- Не можна застосовувати підрядкові й надрядкові символи. Наприклад, вираз $2.5*(a-b^2)$ є неправильним.
  Його слід записати так: 2.5*(a–b*b).
- Не можна записувати дві або більше операцій безпосередньо одна за одною. Наприклад, вираз a*–b є некоректним.
  Його слід записати так: a*(–b).
- Кожній дужці, що відкривається, у виразі має відповідати дужка, що закривається.
- Типи операндів виразу мають бути узгоджені. Якщо автоматично вони не узгоджуються, слід використати засоби перетворення типів, що вже були описані раніше.

Операції в арифметичному виразі виконуються з урахуванням їх пріоритету, а операції, що мають однаковий пріоритет, — у порядку їх запису, тобто зліва направо.

Запитання для перевірки знань

1. Який тип даних отримується після виконання операції ділення?
2. Який тип результату отримується після цілочислового ділення?
3. Поясніть сутність арифметичних операторів із присвоюванням.
4. Які існують типи операторів?
5. У чому полягає сутність ділення за модулем?
6. Поясніть сутність оператора !=.
7. Як виконується логічний оператор and?
8. Поясніть сутність оператора or.
9. Які операції виконуються над рядками?
10. Поясніть сутність операції повторення рядків.
11. Наведіть приклад перевірки входження одного рядка до іншого.
12. Наведіть приклад використання оператора повторення з присвоюванням.

Завдання для самостійного виконання

1. Дано два числа. Виконайте їх цілочислове ділення та ділення за модулем.
2. Дано два числа. Виконайте їх додавання та множення з використанням відповідних арифметичних операторів із присвоюванням.
3. Дано два числа. Виконайте їх порівняння на рівність і перевірте, чи більше перше число за друге.
4. Виконайте об'єднання рядків слів "опрацю" і "вання".
5. Складіть найпростіший код перевірки входження рядка "Чемпіонат України" в рядок "Чемпіонат України з футболу".
6. Складіть найпростіший код з'єднання рядків "зменшення значення" і "змінної" з використанням оператора об'єднання з присвоюванням.

2.4. Модулі, функції і методи для опрацювання числових даних

Пригадайте, які функції опрацювання числових даних ви використовували у 8 і 9 класах. Чи доводилося вам самостійно розробляти функції опрацювання чисел?

У мові Python використовуються цілі числа (тип int), дійсні (тип float) і комплексні (тут не розглядаються). Якщо в арифметичній операції використовуються різні типи чисел, то числа типу int автоматично перетворюються на тип float, і результат отримується типу float.

Далі будуть розглядатися операції лише в десятковій системі числення.

Для роботи з числами можуть застосовуватися вбудовані у мову Python функції, основні з яких наведено в табл. 1.

Числа можуть подаватися в десятковій, двійковій, вісімковій і шістнадцятковій системах числення. У процесі виконання арифметичних операцій над числами в різних системах числення вони автоматично перетворюються на десяткову систему числення.

Таблиця 1. Основні функції для роботи з числами

Функція	Призначення	Приклад
int([<об'єкт>])	Перетворює об'єкт на ціле число	>>> int(8.7), int("57") (8, 57)
float([<число або рядок>])	Перетворює ціле число або рядок на дійсне число	>>> float(23), float("11.7") (23.00, 11.7)
round(<число>[,<кількість знаків після коми>]))	Повертає найближче ціле, якщо кількість знаків не вказана, а також кількість знаків після коми, якщо вона вказана	>>> round(0.47), round(45.347, 1) (0, 45.3)
abs(число)	Повертає абсолютне значення	>>> abs(22), abs (–66) (22, 66)
pow(<число>,<степінь>)	Повертає число в степені	>>> pow(10, 2), pow (2, 4) (100, 16)
max(<числа через кому>)	Повертає максимальне значення	>>> max(9, 4, 7), max(5, 9.7, 3) (9, 9.7)
min(<числа через кому>)	Повертає мінімальне значення	>>> min(21, 5, 19), min(7, 2.5, 6) (5, 2.5)
sum([<числа>],<початкове значення>)	Повертає суму чисел. Якщо вказане початкове значення, то воно додається до суми	>>> sum([4, 5, 7]), sum([12, 3], 33) (16, 48)

Модуль **math** містить константи:

число pi:

```
>>> import math
>>> math.pi
3.14159265389793
```

число e:

```
>>> math.e
2.718281828459045
```

Усі типи даних мови Python є класами. Класи містять методи. Метод — це програма, яка виконує ту чи іншу функцію. Метод викликається для конкретного об'єкта. Для його виклику спочатку вказується об'єкт, потім крапка, за якою слідує ім'я метода: <об'єкт>.<ім'я методу>.

Кожний клас підтримує свої методи.

Модуль math, який містить стандартні константи та функції, використовують для роботи з числами. Для роботи з константами та функціями необхідно імпортувати його в програму за допомогою інструкції import math.

Найуживаніші функції модуля math наведено в табл. 2.

Таблиця 2. Найчастіше вживані функції модуля math

Функція	Призначення	Приклади
sqrt()	Корінь квадратний	>>> math.sqrt(85) 9.219544457292887
log10()	Логарифм десятковий	>>> math.log10(15) 1.1760912590556813
ceil()	Найближче більше ціле	>>> math.ceil(3.213) 4
floor()	Найближче менше ціле	>>> math.floor(6.79) 6
pow(число, степінь)	Підносить число до степеня	>>> math.pow(8, 3) 512.0
fmod()	Остача від ділення	>>> math.fmod(35, 3) 2.0
factorial()	Факторіал числа	>>> math.factorial(4) 24

Модуль random містить функції, які генерують випадкові числа. На початку використання функцій необхідно цей модуль імпортувати в програму за допомогою команди import random.

У табл. 3 наведено найуживаніші з них.

Модуль **math** містить:

- стандартні тригонометричні функції: (sin(), cos(), tan());
- обернені тригонометричні функції: asin(), acos(), atan();
- функцію перетворення радіанів на градуси degrees();
- функцію перетворення градусів на радіани radians();
- функцію експоненти exp().

Пам'ятайте, що для виконання наведених у таблиці прикладів спочатку необхідно імпортувати модуль **math** за допомогою команди >>> import math.

Таблиця 3. Деякі функції модуля random

Функція	Призначення	Приклад

random()	Генерує випадкове число від 0.0 до 1.0 [0.0, 1.0) 1.0 не включено до діапазону	>>> import random
		>>> random.random()
		0.4632200164843052

uniform(початок,кінець)	Генерує дійсне випадкове число в діапазоні від «початок» до «кінець». 1.0 включено до діапазону	>>> random.uniform(3, 9) 4.951275580428769
randint(початок,кінець)	Генерує ціле випадкове число в діапазоні від «початок» до «кінець». 1.0 не включено до діапазону	>>> random.randint(3, 8) 7
choice(послідовність)	Вибирає з послідовності (рядка, списку або кортежу) випадковий елемент	>>> random.choice(«Python») 'h'

Запитання для перевірки знань

1. У яких системах числення можуть подаватися числа?
2. Для чого призначено функцію float?
3. Для чого призначено модуль math?
4. Для чого призначено функцію pow?
5. Для чого призначено модуль random?
6. Наведіть приклад використання функції round.
7. Яку структуру має функція sum?

Завдання для самостійного виконання

1. Визначте максимальне та мінімальне число в послідовності: 38, 20, 5, 40, 13.
2. Початкове значення суми дорівнює 14. Додайте до неї такі числа: 23, 50, 37, 13.
3. Визначте випадкове число в послідовності: 23, 40, 29, 33, 17.
4. Обчисліть суму факторіалів чисел 5 і 7.

3. Реалізація базових алгоритмічних конструкцій
3.1. Реалізація алгоритмів із розгалуженням

Відомо, що існує три базові структури алгоритмів: слідування, розгалуження та повторення. Пригадайте, якими операторами мови ви, можливо, реалізовували розгалуження й повторення.

В алгоритмах із розгалуженням залежно від умови виконуються ті чи інші інструкції.

Існують три типи розгалуження: *одноальтернативне* (неповне галуження), *двоальтернативне* (повне галуження) *і багатоальтернативне* (вибір, варіант).

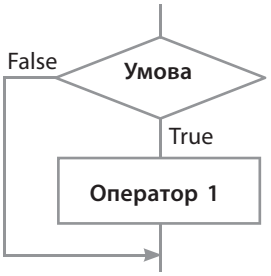


Рис. 1. Графічна схема одноальтернативного розгалуження

- Одноальтернативне розгалуження

Графічну схему одноальтернативного розгалуження зображено на рис. 1.

Одноальтернативне розгалуження виконується так. Перевіряється Умова, і якщо вона істинна (True), то виконується Оператор 1. Інакше буде виконуватися оператор, який розташовано безпосередньо за оператором розгалуження.

У мові Python цей тип розгалуження реалізується оператором умовного переходу в неповній формі:

```
if <логічний вираз>:
    <блок S>
```

Приклад 1.

I варіант запису:

```
if x > 4:
    y = 2 * x + 5;
    z = y + x * x
```

II варіант запису:

```
if x > 4:
    y = 2 * x + 5; z = y + x * x
```

В один рядок 2 оператори записують через «;».

У наведеному фрагменті (приклад 1) умовою є логічний вираз x>4. Якщо він має значення True, то обчислюється значення виразу y = 2 * x + 5, потім — виразу z = y + x * x. Після завершення їх обчислення буде виконуватися оператор, який розташовано безпосередньо за оператором умовного переходу.

Якщо ж логічний вираз x>4 має значення False, вказані вирази не обчислюються, а управління буде одразу передано оператору, що слідує за оператором умовного переходу.

Звернемо увагу на те, що в наведеному прикладі в операторі умовного переходу міститься блок із двох операторів, який починається після чотирьох пробілів.

Приклад 2.

У банк покладено 5000 грн під 20 % річних. Якщо гроші знімаються раніше зазначеного терміну, відсотки не нараховуються. Програму обчислення реальної суми, яку отримано через *k* днів, зображено на рис. 2.

Примітка. У змінній *s* спочатку зберігається початкова сума вкладу, а потім сума, яку отримано з банку.

```
s = 5000                # початкова сума вкладу
k = int(input("Через скільки днів?"))
if k > 366:             # після одного року?
    s = s + s * 0.2     # обчислення суми вкладу
print("s= ", s)         # виведення суми вкладу
input()                 # очикування натиснення Enter
```

Рис. 2. Код обчислення суми вкладу

• Двоальтернативне розгалуження

Графічну схему двоальтернативного розгалуження зображено на рис. 3.

Двоальтернативне розгалуження виконується так: перевіряється Умова і, якщо вона має значення True, то виконується Оператор 1. Інакше виконується Оператор 2.

У мові Python цей тип розгалуження реалізується оператором такої структури:

```
if <логічний вираз>:
    <блок s1>
else:
    <блок s2>
```

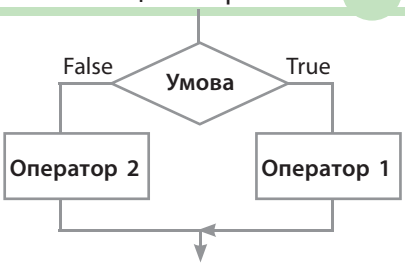


Рис. 3. Графічна схема двоальтернативного розгалуження

Приклад 3.

```
if x==10:
    y = "Python"
    print ("y=", y)
else:
    y = "Pascal"
    print ("y=", y)
```

У цьому прикладі, якщо значення змінної *x* дорівнює 10, виводиться повідомлення Python, інакше — Pascal.

Приклад 4.

Довжина сторони квадрата дорівнює a, радіус кола — r. Чи можна у квадрат уписати коло?

Програму розв'язування задачі наведено на рис. 4.

```
r = int(input("Увести радіус: "))
a = int(input("Увести сторону квадрата: "))
if r == int(a / 2):    # перевірка умови
    print("Можна вписати")
else:                  # інакше
    print("Вписати не можна")
input()
```

Рис. 4. Код визначення можливості вписати коло у квадрат

Звернемо увагу на те, що безпосередньо за ключовими словами if і else може слідувати один оператор або блок операторів. Нагадаємо, оператори блоків починаються від початку рядка через чотири пробіли.

Приклад 5.

Із м. Києва до м. Чернігова на велосипеді о 8.00 виїхав Василь й рухався 5,3 год. Потім о 9.00 виїхав Микола і рухався за тим самим маршрутом 4,2 год. Хто з велосипедистів прибув до Чернігова першим?

Програму розв'язування задачі наведено на рис. 5.

```
if 8.0 + 5.3 < 9.0 + 4.2:
    print("Першим буде Василь")
else:
    print("Першим буде Микола")
input()
```

Рис. 5. Код визначення першого велосипедиста

• Багатоальтернативне розгалуження

У цьому типі розгалуження реалізується розгалуження з багатьма варіантами вибору. Графічну схему розгалуження з трьома можливими варіантами зображено на рис. 6.

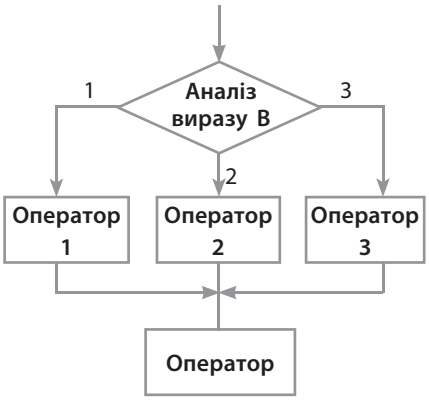


Рис. 6. Графічна схема розгалуження з трьома варіантами вибору

Вираз B повинен мати ціле значення. Якщо його значення дорівнює одиниці, виконується Оператор 1, якщо вираз має значення два, виконується Оператор 2, а якщо три — Оператор 3. Після виконання одного з цих операторів виконується Оператор, що міститься після оператора багатоальтернативного розгалуження.

У випадку якщо вираз B не дорівнює жодному з перелічених значень, одразу виконується Оператор (рис. 6).

У мові Python розгалуження за багатьма варіантами вибору реалізується оператором такої структури:

```
if <вираз> == <значення_1>:
    <блок_1 >
elif <вираз> == <значення_2>:
    <блок_2>
      …
elif <вираз> == <значення_N>:
    <блок_N>
else:
    <блок_N+1>
```

Отже, кожне значення виразу має бути унікальним константним виразом. Значення виразу порівнюється з кожним значенням в операторах elif у порядку їх розташування. Якщо певне значення вираз збігається зі значенням оператора elif, то виконується той блок операторів, що розташований безпосередньо за цим оператором. Якщо значення вираз не збігається з жодним значенням в операторах elif, виконується блок N+1.

Приклад 6.

Призерами чемпіонату України з футболу є команди: 1 — «Динамо», 2 — «Шахтар», 3 — «Зоря». За номером призера необхідно визначити назву команди.

Програму розв'язування задачі зображено на рис. 7.

```
nom = int(input("Увести номер призера: "))
if nom == 1:
    print("Команда Динамо")
elif nom == 2:
    print("Команда Шахтар")
elif nom == 3:
    print("Команда Зоря")
else:
    print("Це не номер призера")
input()
```

Рис. 7. Код визначення призера

Запитання для перевірки знань

1. Які існують типи розгалужень алгоритмів?
2. Яким оператором мови Python реалізується одноальтернативне розгалуження?
3. Сформулюйте сутність двоальтернативного розгалуження.
4. Які помилки є в записі оператора:

```
if x = 3
    y = 2,  p = y = y * 4.1
```

5. Наведіть задачу, для розв'язання якої використовується двоальтернативне розгалуження.
6. Запишіть структуру оператора двоальтернативного розгалуження.
7. Поясніть на прикладі сутність багатоальтернативного розгалуження.
8. Чи можна будь-який алгоритм розгалуження реалізувати тільки оператором одноальтернативного розгалуження? Наведіть приклад.

Завдання для самостійного виконання

1. Оклад працівника або працівниці дорівнює *s* грн. За якісне й дострокове виконання завдання нараховується премія в розмірі 50 % окладу. Розробіть програму визначення реальної заробітної платні.
2. Дано два дійсних числа. Розробіть програму, за допомогою якої менше число замінюється нулем, а більше — одиницею.
3. Дано ціле число *n*. Розробіть програму, за допомогою якої це число збільшується на 7, якщо воно більше 15, і зменшується на 5, якщо воно менше або дорівнює 15.
4. Тренерка формує команду для гри в баскетбол з учениць, які на зріст не нижчі ніж 180 см. Розробіть програму визначення, чи потрапить у цю команду дівчина зростом *h* см?
5. Дано рівносторонній трикутник зі стороною *b*. Розробіть програму визначення, чи можна в трикутник уписати коло з радіусом *r*.
6. Відомі три найкращі результати учнівства школи в стрибках у довжину. Розробіть програму, за допомогою якої за результатами стрибків визначаються прізвища учнів.
7. У п'ятницю в 9 класі такі уроки: 1 — історія, 2 — математика, 3 — географія, 4 — інформатика, 5 — фізкультура. Розробіть програму, за допомогою якої визначається назва предмета за його номером у розкладі.
8. Знайдіть в Інтернеті та складіть список із п'яти найбільших міст України. Розробіть програму, за допомогою якої визначається назва міста за його номером у цьому списку.

3.2. Вкладені оператори умовного переходу

На практиці трапляються випадки, коли виконання або невиконання події залежить не від однієї умови, а від кількох. Пригадайте такі випадки. Як, на вашу думку, можна їх реалізувати операторами умовного переходу?

Вкладені оператори умовного переходу — це оператори умовного переходу, які входять до складу інших операторів умовного переходу.

Існують різні конструкції вкладених операторів. Один із варіантів конструкції вкладених операторів умовного переходу подано на графічній схемі (рис. 1).

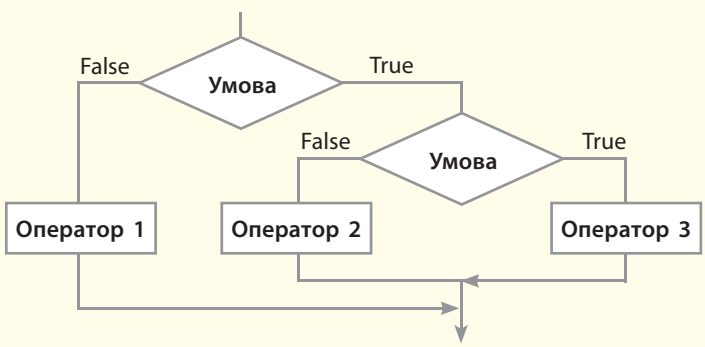


Рис. 1. Графічна схема алгоритму зі вкладеним умовним переходом

Розробники мови Python є прихильниками певної філософії програмування, яку називають «The Zen of Python» («Дзен Пайтона»). Її текст можна отримати в інтерпретаторі Python за допомогою команди import this (лише один раз за сесію).
Автором цієї філософії вважається Тім Пейтерс.

Текст філософії «The Zen of Python» («Дзен Пайтона»)

- Гарне краще за потворне.
- Явне краще за неявне.
- Просте краще за складне.
- Складне краще за заплутане.
- Плоске краще за вкладене.
- Розріджене краще за щільне.
- Легкість читання має значення.
- Особливі випадки не є настільки особливими, щоб порушувати правила. Хоча практичність є важливішою за бездоганність.
- Помилки ніколи не повинні проходити непомітно. Якщо їх приховування не прописано явно.
- Зустрівши неоднозначність, опирайтеся спокусі вгадати.

Розглянемо на прикладі алгоритм обчислення виразу

$$y=\begin{cases} ax, \text{ якщо } x>0 \text{ і } a\geq 0; \\ 2ax, \text{ якщо } x>0 \text{ і } a<0; \\ 2, \text{ якщо } x\leq 0. \end{cases}$$

На рис. 2 зображено фрагмент схеми алгоритму, що реалізує вираз ax, якщо x>0 і a≥0.

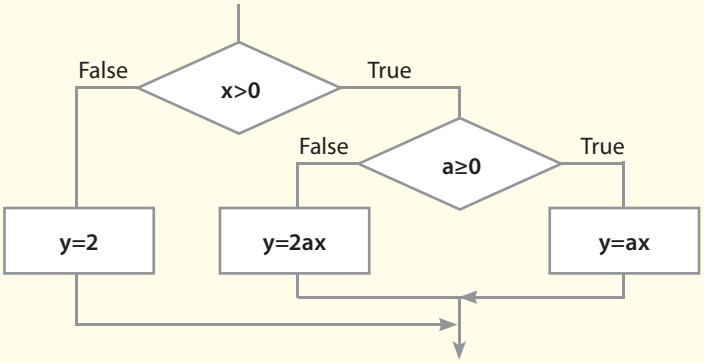


Рис. 2. Графічна схема алгоритму обчислення значення виразу

Мовою Python цей алгоритм можна реалізувати так:

```
If x > 0:                    #перший оператор if
    If a >= 0:               #другий оператор if
      y = a * x
    else:                    #для другого оператора if
      y = 2 * a * x
else:                        #для першого оператора if
    y = 2
```

Алгоритм, наведений на рис. 2, реалізується програмою, яку зображено на рис. 3.

```
a = int(input("Увести a: "))
x = int(input("Увести x: "))
if x > 0:                        # якщо x>0
    if a >= 0:
        y = a * x                # гілка Так другого оператора if
    else:
        y = 2 * a * x            # гілка Ні другого оператора if
else:
    y = 2                        # гілка Ні першого оператора if
print("y = ", y)
input()
```

Рис. 3. Код обчислення значення арифметичного виразу

Інший варіант конструкції з вкладеними операторами умовного переходу подано на рис. 4.

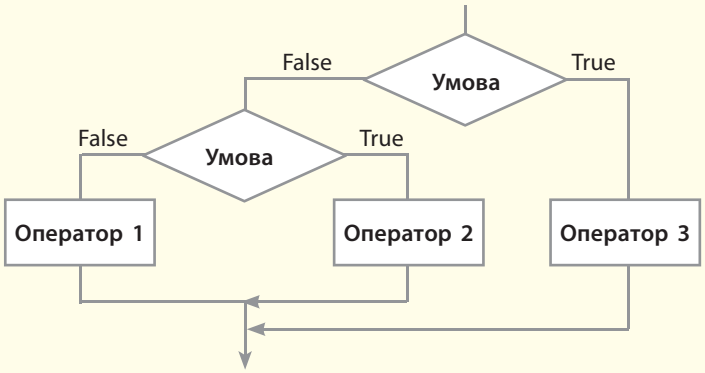


Рис. 4. Графічна схема алгоритму зі вкладеним умовним переходом

Розглянемо ще один приклад застосування конструкції зі вкладеними операторами умовного переходу для розв'язування задачі.

Python використовується для розв'язування великої кількості як наукових, так і бізнес-завдань.

У науковій сфері мову **Python** широко використовують західні вчені-непрограмісти (математики, фізики, біологи) завдяки простоті вивчення та наявності популярних додаткових потужних бібліотек.

Приклад.

За рейтингом УЄФА футбольні команди розташовано так: «Реал», «Барселона», «Баварія», «Челсі», «Атлетіко», «Бенфіка», «Шальке-04», «Порту», «Арсенал», «Манчестер». На рис. 5 наведено програму, яка за номером команди визначає країну.

Наведено один із варіантів виконання програми:

```
Увести номер команди: 9
країна - Англія
```

```
kom = int(input("Увести номер команди: "))
if kom == 1 or kom == 2 or kom == 5:                    # команда Іспанії
    kr = "Іспанія"
else:
        if kom == 3 or kom == 7:                        # команда Німеччини
            kr = "Німеччина"
        else:
                if kom == 4 or kom == 9 or kom == 10:   # команда Англії
                    kr = "Англія"
                else:
                    if kom == 6 or kom == 8:            # команда Португалії
                        kr = "Португалія"
                    else:
                        kr = "Рейтинг невідомий "
print("країна - ", kr)
input()
```

Рис. 5. Код визначення назви країни

Запитання для перевірки знань

1. Поясніть сутність вкладеного оператора умовного переходу.
2. Накресліть графічну схему алгоритму зі вкладеним умовним переходом.
3. Поясніть правило запису блоку операторів у вкладеному операторі умовного переходу.
4. Наведіть приклад алгоритму, що реалізується за допомогою вкладеного оператора умовного переходу.
5. Проаналізуйте порядок виконання алгоритму (див. рис. 2) для випадку x>0, a<0.
6. Проаналізуйте порядок виконання коду (див. рис. 3), якщо вводяться такі значення: a=0, x=0.

Завдання для самостійного виконання

1. Дано три дійсних числа a, b, c. Якщо a>b>c, кожне число збільшується вдвічі, інакше — кожне число зменшується на одиницю. Розробіть програму реалізації цього завдання.
2. Дано три сторони трикутника. Розробіть програму, за допомогою якої визначається, чи є цей трикутник прямокутним.
3. Дано три сторони трикутника. Розробіть програму для визначення, чи є цей трикутник рівнобедреним.
4. Літак вилітає за розкладом із Києва в Мюнхен, якщо швидкість вітру в Києві менше v, а хмарність у Мюнхені не менше h. Розробіть програму для визначення, чи вилетить літак за розкладом.
5. Шкільна команда шахістів із трьох учнів потрапляє до фінальних міських змагань у випадку, якщо у відбірних змаганнях кожен учень набере не менше 5 очок. Розробіть програму, за допомогою якої визначається, чи потрапить команда до фінальних змагань, якщо перший учень набрав a очок, другий — b очок і третій — c очок.
6. У Всеукраїнській учнівській олімпіаді з інформатики з однієї школи беруть участь два учні. Щоб потрапити до української команди для участі в міжнародній олімпіаді, потрібно набрати не менше k балів. Перший учень набрав a балів, а другий — b балів. Розробіть програму, за допомогою якої визначаються всі варіанти потрапляння до цієї команди учнів школи.
7. Перед останнім туром чемпіонату України з футболу склалася така ситуація. Щоб команда «Динамо» (Київ) стала чемпіоном, їй потрібно виграти в команди «Ворскла», і щоб «Шахтар» водночас програв «Дніпру». Розробіть алгоритм зі вкладеним умовним оператором визначення, чи стане «Динамо» чемпіоном.

3.3. Реалізація циклічних алгоритмів

Пригадаєте, яку алгоритмічну структуру називають повторенням (циклом), які існують види циклів.

Для запису алгоритмів із повторенням (циклів) мовою Python використовують два види операторів циклу: з параметром та з умовою.

3.3.1. Цикли з параметрами

Повторення (цикл) — це алгоритмічна структура, за допомогою якої та сама послідовність дій виконується багаторазово для різних значень змінних.

Серію інструкцій, які виконуються багаторазово під час виконання циклу, називають тілом циклу.

У циклах із параметром (циклах зі змінною циклу) кількість повторень інструкцій тіла циклу заздалегідь відома.

Існують різні варіанти циклів із параметрами. Структуру одного з них зображено на рис. 1. Тут змінна i — це параметр циклу (змінна циклу). Блок Оператор є тілом циклу. У блоці може бути одна інструкція або кілька. Змінна n містить кінцеве значення змінної циклу, а змінна a — її початкове значення.

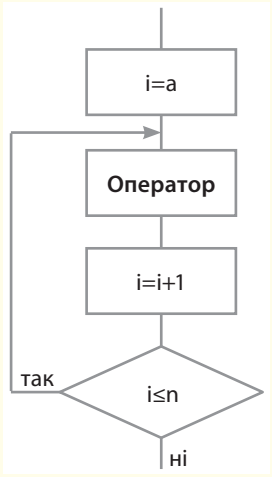


Рис. 1. Фрагмент графічної схеми циклу з параметром

На рис. 2 наведено фрагмент блок-схеми циклічного алгоритму для отримання таблиці множення на 8. Результат множення чисел від 1 до 10 на 8 отримується у змінній p й одразу виводиться. У цьому алгоритмі тілом циклу є інструкції p=8*i, Виведення p, i=i+1, які виконуються 10 разів (можна записати скорочено i+=1). Інструкція i=1 виконує підготовку до реалізації інструкцій тіла циклу, а за допомогою інструкції i≤10 здійснюється перевірка завершення їх виконання.

Рис. 2. Фрагмент графічної схеми циклічного алгоритму

Як бачимо, у цій схемі змінна i виконує функцію лічильника циклів. Як тільки значення цієї змінної стане більше 10, виконання циклу завершується. Такий принцип реалізації циклів із параметром застосовується в багатьох мовах програмування, у тому числі найсучасніших.

У мові Python теж застосовується змінна циклу, але її значення послідовно вибираються зі значень об'єкта.

Цикли з параметром у мові Python реалізуються оператором циклу for (для), який має таку загальну структуру:

```
for <змінна циклу> in <об'єкт>:
        <блок інструкцій тіла циклу>
    [ else:
        <блок інструкцій>      # виконується, якщо
                                 не використовується
                                 оператор break
]
```

Інструкції у квадратних дужках є необов'язковими. Якщо всередині циклу не використовується оператор **break** (із ним ми познайомимося пізніше), то після завершення циклу виконуватиметься блок інструкцій, що міститься після слова **else**.

де:

<об'єкт> — може бути рядок, список, словник та інші типи даних, які підтримують реалізацію циклу. Тип об'єкта програміст може також створити самостійно;

<змінна циклу> — поточне значення об'єкта. Початкове її значення — це перший елемент об'єкта. У другому циклі ця змінна набуде значення другого елемента об'єкта і так далі до останнього.

Блок інструкцій тіла циклу буде виконуватися до тих пір, доки змінна циклу послідовно не набуде усіх значень, що містяться в об'єкті. Наприклад, якщо об'єктом є квадрат, сторони якого набувають таких значень: 2, 5, 9, 10, то цикл буде виконуватися 4 рази для значень змінної циклу 2, 5, 9 і 10.

Найпростіша структура оператора for така:

```
for <змінна циклу> in <об'єкт>:
<блок інструкцій тіла циклу>
```

Розглянемо, як *виконується оператор циклу*.

Крок 1	На початку виконання циклу змінна циклу набуває значення першого елемента об'єкта. Оператор **in** генерує логічне значення **True**, і виконується блок інструкцій тіла циклу.
Крок 2	На цьому кроці змінна циклу набуде значення другого елемента об'єкта й також буде генеруватися значення **True**, у результаті чого буде виконано блок інструкцій тіла циклу.
Крок 3	Після того як усі елементи об'єкта будуть перебрані, оператор **in** згенерує значення **False**, блок інструкцій тіла циклу не виконається, а управління буде передано першій інструкції, що розташована безпосередньо за блоком інструкцій тіла циклу.

Приклад 1

Розробити програму реалізації алгоритму отримання таблиці множення на 8.

Блок-схему алгоритму отримання таблиці множення на 8 зображено на рис. 2, а програму його реалізації — на рис. 3.

У цьому прикладі застосовується функція range(). Загальна структура цієї функції така:

range ([<початок>,] <кінець> [, <крок>])

Як бачимо, обов'язковим є лише параметр кінець. Саме така структура оператора використана в наведеному прикладі. За допомогою функції range(10) формується діапазон чисел від 0 до 10, тому змінна циклу спочатку набуде значення нуль. Але нам не потрібно множити число 8 на нуль. Тому значення змінної і одразу збільшується на одиницю (i = i+1). Після виконання трьох інструкцій тіла циклу змінна і набуде чергового значення функції range(), тобто значення одиниці, яке наступним оператором збільшується на одиницю, у результаті чого число 8 буде помножене на 2. Цей процес повторюється до отримання результату 8*10.

```
# об'єктом в операторі for є цілі числа
for i in range(10):
    i = i + 1    # можна записати як i += 1
    p = 8 * i
    print("8 *", i," = ", p)
input()
```

Рис. 3. Код множення чисел на 8

Приклад 2.

У банк клієнт поклав 10000 грн під 20 % річних. Визначити суму вкладу за кожний із п'яти років. Програму реалізації цього завдання зображено на рис. 4.

```
s = 10000
for i in range(5):
    i = i + 1          # можна записати як i += 1
    s = s + 0.2 * s    # можна записати як s += 0.2 * s
    print("за ", i," рік: ", s )
input()
```

Рис. 4. Код обчислення суми вкладу

Приклад 3.

Дано рядок символів. Програму виведення кожного символу з рядка через один пробіл і підрахунку в ньому кількості символів зображено на рис. 5.

У цьому прикладі об'єктом у структурі оператора for є рядок. Змінна циклу s послідовно набуває значень, починаючи з букви м до букви р. Аргумент end=" " в операторі print забезпечує виведення символів рядка через пробіл в одному рядку.

```
k = 0              # початкова кількість
                   символів
                   # об'єктом в операторі
                   for є рядок
for s in "мікропроцесор":
    print(s, end = " ")
    k = k + 1      # можна записати як k += 1
print()            # перехід на новий рядок
print("Кількість символів = ", k)
```

Рис. 5. Код обчислення кількості букв у рядку

Результат виконання програми має такий вигляд:

```
м і к р о п р о ц е с о р
Кількість символів = 13
```

Приклад 4.

У заданій послідовності комп'ютерних термінів підрахувати кількість конкретного терміна й вивести його стільки разів, скільки він повторюється.

На рис. 6 наведено код, за допомогою якого в заданій послідовності виводиться слово «біт» і підраховується його кількість.

Результат виконання програми має такий вигляд:

```
біт біт біт
Таких слів: три
```

```
k = 0                          # початкова кількість слів
                               # об'єктом у наступному
                               операторі for є список
for p in ['миша', 'біт', 'біт', 'принтер', 'біт']:
    if p == 'біт':
        print(p, end=" ")      # друк через один пробіл
        k += 1                 # обчислення кількості слів
print()                        # перехід на новий рядок
print("Таких слів: ", k)
```

Рис. 6. Код обчислення кількості заданих слів

3.3.2. Цикли з умовою

Існує два види циклів з умовою: цикли з передумовою і цикли з післяумовою.

У циклах з умовою кількість виконання інструкцій тіла циклу заздалегідь невідома: вона завершується після досягнення певної умови.

- Цикли з передумовою

Графічну схему циклу з передумовою зображено на рис. 7.

Блок Оператор — це одна або кілька інструкцій тіла циклу, які виконуються доти, доки умова має значення True (гілка Так). Умова виконання інструкцій тіла циклу перевіряється перед початком їх виконання. Це означає, що оператори тіла циклу можуть бути не виконані жодного разу.

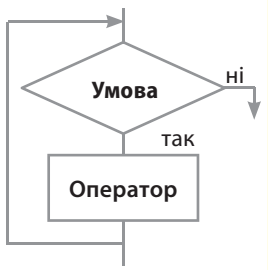


Рис. 7. Графічна схема циклу з передумовою

У мові Python цикли з передумовою реалізуються оператором while, який має таку структуру:

```
<початкове значення>
while <умова>):
        <блок інструкцій тіла циклу>
        <зміна початкового значення>
[ else:
        <інструкції, які не виконуються,
        якщо не використовується оператор break>
]
```

Тут <умова> — це вираз, який має значення True або False.

Приклад 5.

Автомобіль рухається зі швидкістю v км/год і раптом гальмує. За першу секунду його швидкість падає на 10 км/год, а за кожну наступну секунду — зменшується на 10 км/год від значення за попередню секунду. Через скільки секунд він зупиниться?

На рис. 8 наведено код програми моделювання процесу гальмування. Тут змінна k — це кількість секунд гальмування, змінна y — значення швидкості гальмування автомобіля за поточну секунду.

```
v = int(input("Увести швидкість автомобіля v = "))
y = 10          #гальмування за першу секунду
k = 0           #змінна k - кількість секунд
while v > 0:
    v = v - y   #поточне значення швидкості
    y = y + 10 #поточне значення гальмування
    k += 1
print("Автомобіль зупиниться через", k, "сек")
```

Рис. 8. Код обчислення часу гальмування автомобіля

Приклад 6.

Мінімальна кількість розрядів (n) для адресації k байт пам'яті визначається нерівністю $2^n>k$.

Програму визначення кількості розрядів зображено на рис. 9.

```
k = int(input("Увести значення k = "))
n = 0             # n - це кількість розрядів
p = 1             # p - поточне значення
                  кількості байтів
while k > p:
    n = n + 1     # можна записати як n += 1
    p = p * 2     # можна записати як p *= 2
print("Мін. кількість розрядів =", n)
```

Рис. 9. Код визначення кількості розрядів

• **Цикли з післяумовою**

Графічну схему циклу з післяумовою зображено на рис. 10.

У таких алгоритмах спочатку виконуються оператори тіла циклу, а потім перевіряється умова. Якщо умова має значення True (Так), виконання операторів тіла циклу продовжується.

Як тільки умова набуде значення False (Ні), виконання операторів тіла циклу припиняється й управління передається першому оператору, розташованому за оператором циклу. У мові Python відсутній оператор, який безпосередньо реалізує такий варіант циклу.

Рис. 10. Графічна схема циклу з післяумовою

У мові Python цикли з післяумовою можна реалізувати такою конструкцією оператора while:

```
while True :
        <блок інструкцій тіла циклу>
if <умова> : break
```

Приклад 7.

Батискаф заглиблюється в океан. За першу хвилину батискаф заглиблюється на 10 м, а за кожну наступну хвилину він заглиблюється на 10 % менше, ніж за попередню хвилину. Через скільки хвилин батискаф досягне глибини 100 м?

Програмний код моделювання процесу заглиблення батискафа зображено на рис. 11.

Тут k — кількість хвилин заглиблення, h — заглиблення за поточну хвилину, H — поточне значення глибини заглиблення. Оператор break здійснює переривання виконання циклу.

```
h = 10                  # заглиблення за першу хвилину
H = 0                   # початкова глибина заглиблення
k = 0                   # кількість хвилин заглиблення
while True:             # цикл з післяумовою
    H += h              # поточна глибина заглиблення
    k += 1              # поточна кількість хвилин
    h -= 0.01 * h       # заглиблення за поточну хвилину
    if H >= 100: break
print("Через:", k, " хвилин")
```

Рис. 11. Код моделювання процесу заглиблення батискафа

Приклад 8.

У банк покладено 10 000 грн під 15 % річних. Кожного року з рахунка знімається 800 грн. Через скільки років сума вкладу перевищить 14 000 грн? Програмний код розв'язання подано на рис. 12. На екран буде виведено: Через 5 років.

Тут s — поточна сума вкладу, k — кількість років вкладу.

```
s = 10000                       # початкова сума вкладу
k = 0                           # початкова кількість років
while True:                     # цикл з післяумовою
    s = (s + 0.15 * s) - 800    # поточна сума вкладу
    k = k + 1                   # поточна кількість років
    if s > 14000: break         # вихід із циклу
print("Через:", k, " років ")
```

Рис. 12. Код обчислення кількості років вкладу

3.3.3. Оператори continue і break

Оператори continue і break застосовуються всередині операторів тіла циклу та призначені для їх переривання.

- Оператор continue перериває цикл і повертає управління на початок циклу. Це дозволяє перейти до наступної ітерації циклу до завершення виконання всіх інструкцій усередині циклу.

На рис. 13 зображено програму виведення усіх парних чисел від 6 до 30, окрім чисел у діапазоні від 13 до 25 включно.

```
for i in range(6, 31, 2):
    if 12 < i < 26:     # якщо числа в діапазоні 12<i<26
        continue        # перехід на нову інтерацію циклу
    print(i)            # виведення чисел 6,8,10,12,26,28,30
```

Рис. 13. Код із використанням оператора continue

- Оператор break здійснює переривання та вихід із циклу навіть у тому випадку, коли всі ітерації ще не виконані. Цей оператор уже використовувався в розглянутих раніше прикладах.

На рис. 14 зображено ще один приклад програми з використанням оператора break. У ньому використовується нескінченний цикл уведення чисел та обчислення їх суми. Але цикл переривається, якщо буде уведено слово «кінець».

```
s = 0                               # початкова сума
while True:                         # нескінченний цикл
    a = input("увести число ")
    if a == "кінець" :              # якщо введено слово кінець
        break                       # переривання нескінченного циклу
    a = int(a)                      # перетворення рядку у число
    s += a                          # обчислення суми чисел
print("сума чисел = ", s)
input()
```

Рис. 14. Код із використанням оператора continue

3.3.4. Вкладені цикли

Циклічні алгоритми, розглянуті раніше, не містять у собі інших циклів, тому їх називають **простими циклами**.

Вкладеними називають цикли, що містяться в іншому циклі.

Цикл, що входить до складу іншого циклу, називають *внутрішнім*, а цикл, який містить інший цикл, — *зовнішнім*.

На рис. 15 наведено приклад блок-схеми алгоритму зі вкладеним циклом, у якому обчислюються площі 30 прямокутників, сторони яких набувають таких значень:

a = 3, 4, 5, 6, 7;
b = 2, 4, 6, 8, 10, 12.

Код програми, що реалізує цей алгоритм, зображено на рис. 16.

```
for a in range(3, 8, 1):        # зовнішній цикл
    for b in range(2, 13, 2):   # внутрішній цикл
        s = a * b               # обчислення площі
print("s = ", s)
input()
```

Рис. 16. Код обчислення площ прямокутників

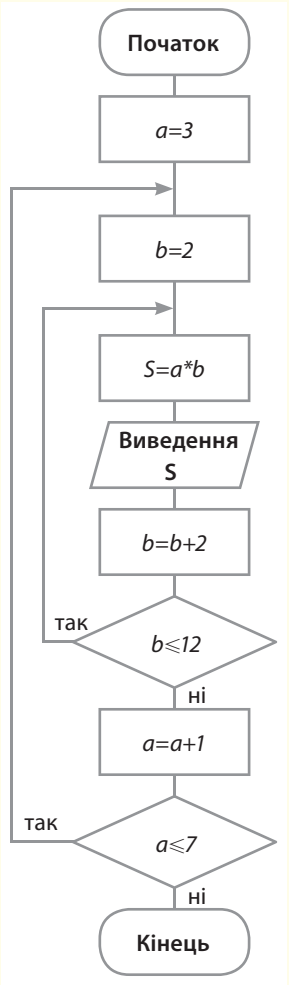


Рис. 15. Графічна схема алгоритму зі вкладеним циклом

Запитання для перевірки знань

1. Які існують види циклічних алгоритмів?
2. Поясніть на прикладі сутність параметра і тіла циклу.
3. Для чого призначено оператор break; оператор continue?
4. Накресліть графічну схему циклу з параметром.
5. Яку структуру має оператор циклу for?
6. Яку структуру має оператор циклу з передумовою?
7. Наведіть приклад вкладеного циклу.
8. Як у мові Python можна реалізувати цикл із післяумовою?

Завдання для самостійного виконання

1. Розробіть код обчислення суми для чисел: 2, 7, 21, 9, 33, 13.
2. Розробіть код, обчислення суми непарних чисел, що більші 7, але менші 25.
3. Розробіть код обчислення суми чисел натурального ряду, максимальне значення якого не перевищує 7.
4. Перший член геометричної прогресії 6, а її знаменник — 0.5. Розробіть код обчислення значень членів прогресії, більших 0.6, і визначення номера останнього члена прогресії, що підсумовується.
5. Дано куб, сторони якого набувають п'ять значень: 3, 4.5, 6, 7.5, 9. Розробіть код визначення об'єму для кожного з них.
6. Радіус першої кулі дорівнює 2 см, а радіус кожної наступної збільшується на 0.5 см. Розробіть код для визначення бокових поверхонь перших шести куль.
7. Відомо результати плавання вільним стилем на дистанції 50 м п'яти учнів кожного з трьох 10 класів школи. Розробіть код, за допомогою якого визначається середній час запливу учнями кожного класу.
8. У банк покладено S грн під N відсотків річних. Розробіть код, за допомогою якого визначається кількість років, через які сума вкладу буде не менше N грн.

4. Вбудовані типи даних та їх опрацювання

У будь-якій мові програмування дані поділяють на дві основні групи: прості (скалярні) та структуровані.

У мові програмування Python також існує поділ на змінювані (mutable) та незмінювані (inmutable) типи даних. Структуровані дані поділяють на лінійні, табличні й ієрархічні.

У лінійній структурі адреса елемента даних однозначно визначається його номером (індексом) у цій структурі. Лінійними структурами є рядкові типи даних і всі типи послідовностей.

У табличних структурах адреса елемента даних визначається кількома індексами. Наприклад, у двовимірних масивах його адреса визначається номером рядка й номером стовпця.

В ієрархічних структурах адреса кожного елемента визначається маршрутом доступу від вершини структури до цього елемента.

Далі розглянемо ті структури даних, які передбачені навчальною програмою для 10 класу, решта вивчається в 11 класі.

У мові програмування **Python** лінійні й табличні структури вбудовані в саму мову, що суттєво полегшує та спрощує користувачу роботу з ними.

4.1. Списки, стеки, черги

Ви неодноразово використовували в повсякденному житті термін «список». Спробуйте дати йому означення. Чи доводилося використовувати його в мові програмування, яку ви вивчали раніше?

4.1.1. Структура списків і операції над ними

Список у мові Python — це впорядкована колекція* об'єктів будь-якого типу у квадратних дужках, які відокремлюються один від одного комою.

Приклад 1.
[5, "файл", "w", 21, [1, 2, 3]].

У прикладі 1 об'єктами списку є числа, рядки та список. У списках можна змінювати значення його елементів, збільшувати та зменшувати кількість елементів, здійснювати пошук потрібних елементів і впорядковувати його елементи. Отже, список є об'єктом, що змінюється, тобто належить до змінюваних типів даних. Самі списки можуть бути вкладені в об'єкти інших типів.

Списки деякою мірою нагадують динамічні масиви вказівників, але в масивах значення елементів можуть бути лише одного типу. Мова **Python** має значну кількість операцій, функцій і методів опрацювання списків.

Список є одним з основних типів даних. Він використовується в мові Python частіше за інші.

На основі списків у програмі можна створювати та опрацьовувати структури даних довільної складності.

Далі розглянемо лише основні операції, якими є, наприклад, звернення до елемента за його індексом, отримання зрізу, конкатенація, повторення, перевірка на входження й ін.

* Змінювана впорядкована колекція (тобто індексована послідовність).

Списки можуть бути *одновимірними* і *багатовимірними*.

Розглянемо спочатку одновимірні списки. Позиція елемента в списку задається індексом, який починається з нуля.

Списки можна створювати простим переліченням елементів списку в квадратних дужках (приклад 2).

Розглянемо **операції, які можна виконувати над списками.**

Приклад 2.

```
>>> al_1 = ["файл", 21, 13, "5"]
>>> al_1
['файл', 21, 13, '5']
```

Щоб звернутися до елемента списку, необхідно у квадратних дужках зазначити індекс елемента	>>> a_1 = [7, 20, "миша"]
	>>> a_1[2]
	'миша'
		# виведення другого елемента

Значення елементів списку можна змінювати шляхом присвоювання їм нових значень	>>> b_1 = [21, 11, 17, 24]
	>>> b_1 [2] = 15
	>>> b_1
	[21, 11, 15, 24]
		# зміна значення другого елемента
		# виведення списку

Елементи списку можна присвоїти кільком змінним	>>> x1, x2, x3 = [3, 7, 5]
	
	>>> x1, x2, x3
	(3, 7, 5)
		# елементи списку присвоюються
		# 3 змінним
		# виведення значень змінних

Якщо елементів списку більше кількості змінних, зайві елементи можна присвоїти одній змінній, поміченій зірочкою	>>> x1, *x2, x3 = [1, 3, 7, 5]
	
	>>> x1, x2, x3
	(1, [3, 7], 5)
		# змінній x2 присвоюються два
		# значення: 3 і 7
		# виведення значень змінних

Операція зрізу (виділення елементів) має формат: <ім'я списку> [початок: кінець: крок]. Усі параметри є необов'язковими	>>> a = [1, 2, 3, 4, 5, 6]
	>>> a[1:4]
	
	[2, 3, 4]
		# виділення елементів з 1 до 3
		# включно

Списки можна об'єднувати	>>> a1 = [1, 2, 3, 4]
	>>> a2 = [5, 6, 7]
	>>> a1 + a2
	[1, 2, 3, 4, 5, 6, 7]
		# об'єднання списків

4.1.2. Функції і методи опрацювання списків

Розглянемо **основні функції опрацювання списків** на прикладах.

Перетворити рядок у список можна за допомогою функції list()	>>> list("процесор") **['п', 'р', 'о', 'ц', 'е', 'с', 'о', 'р']**	# перетворення рядка в список

Отримати кількість елементів у списку можна за допомогою функції len()	>>> a = [1, 4, 6,8]
	>>> len(a)
	4
		# довжина списку

Отримати список із випадковими числами з іншого списку можна за допомогою функції sample (послідовність, кількість елементів) з модуля random	>>> import random
	>>> random.sample(range(51), 6)
	[48, 11, 46, 3, 21, 0]
		# імпортування модуля random
		# шість випадкових чисел
		# у діапазоні 0–51

Перетворити список у рядок можна різними засобами. Найпростіший із них — використання функції str	>>> a1 = ["file1", 25, "file2", 21]
	>>> str(a1)
	"['file1', 25, 'file2', 21]"
		# список
		# перетворення списку в рядок

Максимальне та мінімальне значення в списку можна знайти за допомогою, відповідно; функцій max() і min()	>>> a1 = [44, 20, 3, 17, 25, 16]
	>>> max(a1), min(a1)
	
	(44, 3)
		
		# виведення максимального
		# та мінімального елементів

Для отримання випадкового елемента зі списку слід імпортувати модуль random і скористатися функцією choice (послідовність) або функцією sample (послідовність, кількість елементів)	>>> import random
	>>> random.choice("a", "ab", "c", "ca", "f"])
	'ab'
		# імпортування модуля random
		# виведення випадкового елемента

Розглянемо **основні методи опрацювання списків** на прикладах.

Метод **append(об'єкт)** — додає один об'єкт у кінець списку	>>> a1 = [1, "as", 2]
	>>> a1.append("bmv")
	
	>>> a1
	[1, 'as', 2, 'bmv']
		# список
		# додавання елемента bmv
		# у кінець списку
		# виведення списку

Метод **extend("послідовність")** — додає елементи послідовності в кінець списку	>>> a1 = [1, 2, "web"]
	>>> a1.extend("file, 7")
	>>> a1
	[1, 2, 'web', 'f', 'i', 'l', 'e', ',', ' ', '7']
		# список
		# додавання елементів у список
		# виведення списку

Метод **insert(індекс, об'єкт)** — вставляє один об'єкт у вказану позицію списку	>>> a1 = [3, "vin", 5]
	>>> a1.insert(1, "file")
	
	>>> a1
	[3, 'file', 'vin', 5]
		# список
		# додавання в 1 позицію списку
		# елемента
		# виведення списку

Метод **pop(індекс)** — видаляє елемент зі списку за вказаним індексом. Видалити елемент зі списку можна також за допомогою оператора **del список**[індекс]	>>> b1 = [7, "com", 5]
	>>> b1.pop(2)
	5
	>>> b1
	
	[7, 'com']
	>>> del b1[1]
	>>>
	[7]
		# список
		# виведення видаленого елемента
		
		# виведення списку після видалення
		# елемента
		
		# видалення елемента зі списку за індексом
		# виведення списку після видалення елемента

Метод **remove(значення)** — видаляє зі списку перший елемент, який містить вказане значення	>>> a1 = [3, 5, "lad", 7, "lad"]
	>>> a1.remove ("lad"); a1 # видалення елемента "lad" із другої # позиції
	[3, 5, 7, 'lad']

Метод **clear()** — видаляє зі списку всі елементи	>>> b1 = [5, 11, 8, 7, 20]
	>>> b1.clear(); b1
	[]

Метод **index(значення [,** початок **[,кінець]])** — повертає індекс елемента, який має вказане значення. За замовчуванням параметрів **початок** і **кінець** пошук елемента буде виконуватися від початку до кінця списку	>>> b1 = [10, 7, 55, 16, 8, 20]
	>>> b1.index(16) # виведення індексу елемента # зі значенням 16
	3

Метод **count(значення)** — використовується для визначення кількості елементів зі вказаним значенням. Якщо елемент відсутній у списку, повертається значення 0	>>> c1 = [7, 13, 14, 7, 7, 5]
	>>> c1.count(7) # виведення кількості елементів # зі значенням 7
	3

Сортування елементів списку реалізується за допомогою методу **sort()**, який має таку загальну структуру: **sort(**[key = None][**,**reverse = False]**)**.
Як бачимо, параметри є необов'язковими. За замовчуванням сортування виконується за зростанням значень елементів з урахуванням регістра. Для сортування за зменшенням слід указати другий параметр таким: **reverse=True:**
Зазначимо, що **метод sort()** перетворює старий список у новий
	>>> a1 = [17, 100, 20, 3, 8, 16, 25] # список
	>>> a1.sort(reverse=True) # сортування в порядку зменшення # значень
	>>> a1 # виведення впорядкованого списку
	[100, 25, 20, 17, 16, 8, 3]

Метод **sorted(послідовність [**,key=None**][**,reverse = False**])** — призначений для сортування списку та збереження старого	>>> b1 = [39, 16, 21, 7, 23, 15] # початковий список
	>>> sorted(b1) # повернення впорядкованого
	[7, 15, 16, 21, 23, 39] списку
	>>> b1
	[39, 16, 21, 7, 23, 15] # повернення початкового списку

4.1.3. Стек і черга

Стек — це виділена область оперативної пам'яті.

Робота зі стеком у програмуванні деякою мірою нагадує роботу зі стосом книжок: на першу книжку кладеться друга, на другу — третя, на третю — четверта і так далі. Щоб узяти першу книжку, покладену в стос, необхідно зняти спочатку четверту, потім третю, далі другу і, нарешті, першу.

Стек працює в порядку LIFO (*Last In, First Out*), тобто останній доданий у стек фрагмент пам'яті буде першим у черзі на вихід зі стеку. Кожного разу, коли функція оголошує нову змінну, вона додається в стек. А коли ця змінна стає неактуальною (наприклад, коли функція припиняє роботу), вона автоматично видаляється зі стеку і область пам'яті стає доступною для інших стекових змінних.

Стек часто використовується для організації виклику підпрограм і повернення в основну програму. Для точки основної програми, з якої здійснюється звернення до підпрограми, у стеку запам'ятовується адреса основної програми, до якої слід повернутися після завершення підпрограми.

Під час кожного звернення до підпрограми в стек додаються нові адреси повернення. Після кожного завершення підпрограми зі стеку знімається адреса повернення в основну програму. Ураховуючи те, що звернення до підпрограм виконується досить часто, стек здебільшого реалізується на апаратному рівні, а не програмному.

Програмно стек реалізується на основі *списку* або *масиву*.

Якщо використовується масив, то потрібно визначати його розмір, щоб за потреби мати достатньо комірок. Неправильне визначення розміру масиву може призвести до помилок у роботі програми або до неефективного використання пам'яті.

У списку для кожного елемента відводиться блок пам'яті, обсяг якого повинен бути достатній для збереження значення елемента та посилання на попередній і наступний елементи стеку.

У мові Python стек програмно реалізується здебільшого на основі списку.

Для роботи зі стеком застосовуються такі визначені для списків методи:

- **append()** — для додавання нового елемента в стек;
- **pop()** — для вибірки елемента із вершини стеку (вершиною стеку називають останній уведений елемент).

Реалізація стеку на основі **масиву** простіша й потребує меншого обсягу пам'яті, але необхідно заздалегідь знати розмір масиву.
Реалізація стеку на основі **списку** надійніша, але потребує більшого обсягу пам'яті.

Приклад 3.

```
>>> stack = [2, 5, 7]      # створення списку
>>> stack.append(12)       # додавання у вершину стеку числа 12
>>> stack                  # виклик стеку
[2, 5, 7, 12]              # значення елементів стеку
>>> stack.pop()            # вибір елемента із вершини стеку
12                         # видалений елемент
>>> stack                  # виклик стеку
[2, 5, 7]                  # значення елементів стеку
>>> stack.pop()            # вибір елемента із вершини стеку
7                          # видалений елемент
```

Черга в програмуванні — це структура даних, що працює за принципом «перший прийшов — перший пішов».

Чергу можна порівняти, наприклад, із чергою в залізничну касу: перший клієнт біля каси обслуговується першим. Елемент, що додається до черги, опиняється в її кінці, а елемент, який видаляється із черги, перебуває на її початку.

У мові Python для роботи з чергою також використовуються списки (приклад 4).

Приклад 4.

```
>>> zherg = ["Олег", "Олена", "Ліда"]
>>> zherg.append("Сашко")
>>> zherg.append("Ліза")
>>> zherg
['Олег', 'Олена', 'Ліда', 'Сашко', 'Ліза']
>>> zherg.pop(0)
'Олег'
>>> zherg
['Олена', 'Ліда', 'Сашко', 'Ліза']
```

4.1.4. Багатовимірні списки

У багатовимірних (або вкладених) списках кожна група елементів списку береться у квадратні дужки. Двовимірний список можна створити, наприклад, так:

```
>>> a1 = [[1, 2, 3], [4, "файл", "s"], [5, 6, 7]]
```

Але для наочності краще записувати так:

```
>>> a1 =  [
          [1, 2, 3],
          [4, "файл", "s"],
          [5, 6, 7]]
>>> a1
[[1, 2, 3], [4, 'файл', 's'], [5, 6, 7]]
```

Для звернення до елемента багатовимірного списку слід у квадратних дужках указати всі його індекси. Наприклад, звернутися до першого елемента першої групи двовимірного списку можна так:

```
>>> a1[1][1]
'файл'
```

Звертатися до всіх або частини елементів списку можна за допомогою операторів циклу for і while, а також функції range(), яка розглядалася раніше. Наприклад, за допомогою оператора циклу for до них можна звернутися так:

```
>>> b1 = ["файл", 5, 25, "ц"]
>>> for i in b1: print (i, end=" ")    # натиснути Enter двічі
файл, 5, 25, ц
```

У генераторах списку разом з операторами циклу можуть використовуватися умовні й інші оператори, що дозволяє виконувати складні перетворення списків. Якщо вираз розміщується всередині не квадратних, а круглих дужок, то буде повертатися не список, а значення певного виразу. Такі конструкції у мові Python називають виразами-генераторами.

Виконаємо, наприклад, підсумовування парних чисел:

```
>>> a1 = [6, 5, 2, 1, 8]
>>> sum ((i for i in a1 if i % 2 == 0))
16   # сума нульового, другого і четвертого елементів
```

За допомогою операторів циклу можна не тільки отримати значення його елементів, а й змінити значення кожного з них або окремих елементів. Такі дії можна виконувати різними способами, у тому числі за допомогою так званих **генераторів списку**. Для їх реалізації оператор циклу розміщується всередині списку:

```
>>> a1 = [2, 3, 8, 5]
>>> a1 = [i * 3 for i in a1]
>>> print(a1)
```

У цьому фрагменті коду кожнен елемент списку множиться на 3. У результаті отримаємо значення списку:
[6, 9, 24, 15]

4.1.5. Приклади програм опрацювання списків

Приклад 5.

У списку [27, 3, 12, 22, 37, 8] знайти максимальний елемент, вилучити елемент на другій позиції, упорядкувати новий список у порядку збільшення значень його елементів зі збереженням попереднього списку, вставити в нього на четверту позицію число 5, потім замінити значення першого елемента на число 10.

Програму, що реалізує це завдання, зображено на рис. 1.

```
a = [27, 3, 12, 22, 37, 8]
print(max(a))      # максимальний елемент списку
a.pop(2)           # вилучити елемент на 2 позиції
print(sorted(a)) # сортувати список зі збереженням старого
print(a)           # виведення старого списку
a.insert(4, 5)     # вставити число 5 на 4 позицію
print(a)           # виведення списку зі змінами
a[1] = 10          # замінити перший елемент на число 10
print(a)           # виведення списку зі змінами
```

Рис. 1. Програма опрацювання списку

Далі наведено результат виконання програми:

```
37
[3, 8, 22, 27, 37]
[27, 3, 22, 37, 8]
[27, 3, 22, 37, 5, 8]
[27, 10, 22, 37, 5, 8]
```

Приклад 6.

Дано два списки: ["Python розробив", "1991"] і ["Гвидо ван"]. Розробити програму, за допомогою якої отримується рядок: Мову Python розробив у 1991 році Гвидо Ван Россум. Один із варіантів програми, що реалізує це завдання, зображено на рис. 2.

```
a = ["Python розробив", "1991"]   # перший список
b = ["Гвидо ван"]                 # другий список
a.insert(0, "Мову")               # вставлення слова Мову
a.insert(2, "у")                  # вставлення букви у
a.append("році")                  # додавання слова році
print(a)                          # виведення 1-го рядку
b.append("Россум")                # зміна другого списку
c = a + b                         # об'єднання списків
print(c)                          # виведення нового списку
print(" ".join(c))                # перетворення списку у рядок
```

Рис. 2. Програма об'єднання списків і перетворення списку в рядок

Далі наведено результат виконання програми:

```
['Мову', 'Python розробив', 'у', '1991', 'році']
['Мову', 'Python розробив', 'у', '1991', 'році', 'Гвидо ван', 'Россум']
Мову Python розробив у 1991 році Гвидо ван Россум
```

Приклад 7.

У послідовності [4, 6, 13, 9, 5, 16, 11] знайти числа, більші 6, збільшити їх удвічі, вивести на екран і обчислити їх суму.

Варіант програми, що реалізує це завдання, зображено на рис. 3.

```
a = [4, 6, 13, 9, 5, 16, 11]   # створення списку
s = 0                          # початкове значення суми
for i in range(len(a)):        # перебрати елементи списку
    if a[i] > 6:               # якщо число більше 6
        a[i] = 2 * a[i]        # збільшення значення елемента
        print(a[i])            # виведення чисел біше 6
        s = s + a[i]           # обчислення суми
print("сума = ", s)
input()
```

Рис. 3. Програма збільшення та обчислення суми чисел, більших 6

Далі наведено результат виконання програми:

```
26
18
32
22
сума= 98
```

Запитання для перевірки знань

1. Що називають списком?
2. Як створюється список?
3. Як можна замінити значення елемента списку?
4. Як здійснюється об'єднання списків?
5. Для чого призначена функція list()?
6. Як перетворюється список у рядок?
7. Як можна отримати випадкові елементи з іншого списку?
8. Для чого призначений метод append()?
9. Поясніть сутність методу insert().
10. Для чого призначений метод remove()?
11. Поясніть, як виконується сортування елементів списку.

Завдання для самостійного виконання

1. Дано список [13, 19, 11, 7, 18]. Вилучте елемент на першій позиції і після цього знайдіть максимальний елемент.
2. Дано список міст України ["Херсон", "Житомир", "Ужгород", "Харків"]. Уставте місто Луцьк на другу позицію і після цього відсортуйте список.
3. Дано список [9, 2, 5, 6]. Замініть число на першій позиції числом 12. Відсортуйте новий список зі збереженням старого.
4. Дано список ["В. Глушков", "вчений України"]. Внесіть у нього зміни, щоб отримати такий список ["В. Глушков — великий", "вчений України"]. Перетворіть список на рядок.
5. Дано два списки: ["Мова", "Pascal"] і ["— це мова", "процедурного програмування"]. Об'єднай-те списки в один і перетворіть його на рядок.
6. Дано список [5, 7, 8, 12, 4]. Обчисліть суму елементів, значення яких більші 5.

4.2. Кортежі, діапазони, множини

Ви вже знайомі з деякими структурами даних, зокрема із списками. Які, на вашу думку, ще структури даних доцільно підтримувати сучасним мовам програмування?

Кортеж у математиці — упорядкована та скінченна сукупність елементів (нескінченний кортеж називається сімейством). Кількість елементів у кортежі визначає його довжину. Наприклад, кортеж із двох елементів (тобто довжини 2) називають двійкою, із трьох елементів — трійкою і т. д. Кортеж із n елементів називають n-кою.

Кортеж — це незмінна впорядкована колекція об'єктів будь-якого типу в круглих дужках (або без них), які відокремлюються один від одного комою.

Кортежі схожі на списки, але відрізняються від них тим, що кортежі є незмінними послідовностями і замість квадратних дужок застосовуються звичайні дужки.

У мові Python **кортеж** (англ. *tuple*) відрізняється від списку тим, що елементи кортежу після його створення не можна змінювати жодним чином.

Кортежі підтримують функції len(), min(), max(), методи index() і count(), сутність яких розглядалася в процесі опису списків.

Діапазони підтримують доступ до елемента за його індексом, отримання зрізу, перевірку на входження і невходження, функції **len()**, **max()**, **min()**, методи **index()** і **count()**.

Діапазон — це незмінна послідовність цілих чисел із початковим, кінцевим значеннями і кроком їх зміни.

Для створення діапазону призначена функція range() такої структури:

range([початок,] кінець[, крок])

Як бачимо, обов'язковим є лише параметр кінець. За замовчуванням значення параметра початок дорівнює нулю, а крок — одиниці.

Множина — це невпорядкована колекція унікальних (тих, що не повторюються) об'єктів будь якого типу.

Множини в мові **Python** створюються абсолютно випадковим чином, згодом вони все одно розташуються у випадковому порядку.

Існує два типи множин: *змінна* (set) і *незмінна* (frozenset()). Множина змінного типу створюється за допомогою вбудованої функції set, генераторів множин, літералів множин та інших.

Ця функція дозволяє також перетворювати послідовність у множину:

```
>>> set("монітор")                  # перетворення рядка
                                      в множину
{'і', 'т', 'м', 'н', 'о', 'р'}
>>>set([5, 8, 4, 8, 2, 4, 7])      # перетворення списку
                                      в множину
{8, 2, 4, 5, 7}
>>> set()                           # пуста множина може бути
set()                                 задана тільки так
```

Для виконання операцій над множинами в мові Python існують різні оператори, функції і методи.

Метод **union()** — призначений для об'єднання множин в одну	>>> a = set([3, 5, 7])
	>>> a.union(set([7, 9, 11]))
	{3, 5, 7, 9, 11}
		# множина a
		# об'єднання 2 множин
		# результат об'єднання записується у множину a

Метод **add()** — додає елемент до множини	>>> b = set([1, 2, 3])
	>>> b.add(4)
	>>> b
	{1, 2, 3, 4}
		# множина b
		# додавання у множину b значення 4
		# виведення множини

Функція **len()** — визначає кількість елементів у множині	>>> len(set([4, 5, 7, 9, 10])) **5**	# визначення кількості елементів # у множині

Метод **a.update(b)** — додає елементи множини **b** до множини **a**	>>> a = set([1, 2])
	>>> a.update(set([5, 6, 7]))
	>>> a
	{1, 2, 5, 6, 7}
		# множина a
		# додавання множини до множини a
		# виведення нової множини

Оператор **in** — перевіряє наявність елемента в множині. Повертає значення **True**, якщо множина містить елемент, інакше — значення **False**	>>> a = set([1, 2, 3, 4, 5])
	>>> 3 in a, 6 in a
	
	(True, False)
		# множина a
		# перевірка наявності в множині
		# чисел 3 і 6

Оператор перевірки на рівність **(==)** — повертає значення **True**, якщо множини рівні, інакше — значення **False**	>>> set([1, 2, 3]) == set([3, 4, 5]) **False**	# перевірка на рівність двох множин

Метод **discard(елемент)** — видаляє з множини елемент, якщо його значення міститься в множині	>>> a = set([1, 2, 3, 4, 5])
	>>> a.discard(2)
	>>> a
	{1, 3, 4, 5}
		# множина a
		# видалення елемента зі значенням 2
		# виведення нової множини

Метод **remove()** — видаляє елемент зі множини та повертає KeyError, якщо такого елемента не існує	>>>> a = set([0, 1, 4, 81] >>>> remove(6) Keyerror	# множина a # видалення елемента зі значенням 6 # такого елемента не існує

Мова Python підтримує також генератор множин, синтаксис якого схожий на синтаксис генератора списку, але вираз береться не у квадратні дужки, а у фігурні:

```
>>> {a * 2 for a in [2, 3, 4, 3, 4]}
{4, 6, 8}
```

Із використанням кортежів (приклад 1–4), діапазонів (приклади 5–7), множин (приклад 8) можна ознайомитися на сайті interactive.ranok.com.ua

Запитання для перевірки знань

1. Що називають діапазоном?
2. Що називають множиною?
3. Як створити кортеж?
4. Як створити діапазон?
5. Як перетворити діапазон на кортеж?
6. Як перетворити список на множину?

4.3. Словники. Функції, операції і методи опрацювання словників

Згадайте, як за терміном (словом) знаходили відповідний пояснювальний текст або його переклад.

Словник у мові Python реалізований у вигляді хеш-таблиці — невпорядкована колекція об'єктів будь-якого типу, доступ до яких здійснюється не за допомогою індексу, а за допомогою ключа.

Ключем може бути будь-який незмінний тип даних, наприклад число, рядок або кортеж, тобто будь-який об'єкт із незмінюваним типом. Значення елементів словника можна змінювати. Словники не є послідовностями, тому такі загальні операції, як отримання зрізу, конкатенації та інші вони не підтримують.

Хеш-таблиця — структура даних, яка реалізує інтерфейс асоціативного масиву. Вона дозволяє зберігати пари (ключ, значення) і здійснювати три операції: операцію додавання нової пари, операцію пошуку та операцію видалення за ключем.

Під час хешування може статися, що різні значення ключів перетворюються в однаковий двійковий код. Такі випадки називають **колізією**. Існують різні алгоритми боротьби з колізією. Один із найнадійніших — так званий алгоритм відкритої адресації.

Для зручності можна уявити словник як невпорядковану множину пар виду ключ і значення. Пари розділяються комами, весь словник перебуває у фігурних {} дужках.

Словники можна створювати різними способами.

- Використати **функцію dict()**. Ця функція має чотири формати. Нижче наведено два основні формати:

1) dict (ключ1=значення1, …, ключN=значенняN):

```
>>> a1 = dict(a2=5, a3=7, a4=8)          # створення словника
>>> a1                                   # виведення словника
{'a4': 8, 'a3': 7, 'a2': 5}
```

2) dict (словник) :

```
>>> b1 = dict({"b2": 5, "b3": 7, "b4": 8})   # створення словника
>>> b1                                       # виведення словника
{'b3': 7, 'b2': 5, 'b4': 8}
```

- Указати всі елементи словника всередині фігурних дужок:

```
>>> a1 = {"a2": 5, "a3": 7, "a4": 8}     # створення словника
>>> a1                                   # виведення словника
{'a4': 8, 'a3': 7, 'a2': 5}
```

Цей спосіб застосовується найчастіше. Щоб створити порожній словник, можна використати функцію dict():

```
>>> a0 = dict()                          # створення пустого словника
>>> a0                                   # виведення словника
{}
```

або

```
>>> a0 = {}                              # створення пустого словника
>>> a0                                   # виведення словника
{}
```

- Об'єднати два списки в список кортежів можна за допомогою **функції zip()**. Створимо два списки, об'єднаємо їх у список кортежу, а потім створимо словник:

```
>>> a1 = ["m", "n"]                      # список із ключами
>>> a2 = [3, 6]                          # список зі значеннями
>>> list(zip(a1, a2))                    # створення списку кортежу
[('m', 3), ('n', 6)]
>>> a3 = dict(zip(a1, a2))               # створення словника
>>> a3                                   # виведення словника
{'n': 6, 'm': 3}
```

Ознайомимося з аспектами створення словника на прикладах.

Щоб звернутися до елемента словника, потрібно вказати його ключ (число, рядок або кортеж) у квадратних дужках. Якщо елемент зі вказаним ключем у словнику відсутній, генерується виняток **KeyError**	>>> a = {2: "файл", "n": "пам'ять", (1, 3): "процесор"}
	>>> a[2], a["n"], a[(1, 3)]
	('файл', 'пам'ять', 'процесор')
		# словник
		
		# звернення до елементів словника

Перевірити наявність ключа у словнику можна за допомогою оператора **in**. Якщо ключ у словнику є, генерується значення **True**, інакше — **False**	>>> a = {1: "as", "kl": 5}
	>>> "kl" in a
	True
		# словник
		# перевірка наявності у словнику ключа kl

Метод **get(ключ)** повертає значення, яке відповідає ключу. Якщо ключ відсутній, повертається значення **None**	>>> a = {"p": 21, 37: "байт", "sk": "біт"}
	>>> a.get("p"), a.get("sk"), a.get(38)
	(21, "біт", None)
		# словник
		# повернення значень словника
		# елемента з ключем 38 немає

Змінити елемент словника або додати новий елемент можна за допомогою його ключа	>>> a = {"b": 25, "c": 100}
	>>> a["b"] = 300
	
	>>> a[2] = "astra"
	>>> a
	{2:'astra', 'c': 100, 'b': 300}
		# словник a
		# зміна значення, що має ключ "b"
		# додавання нового елемента
		# виведення оновленого словника

За допомогою функції **len()** у словнику можна визначити кількість ключів	>>> a = {1: "один", 2: "два", 3: "три"}
	>>> len (a)
	3
		# словник a
		# визначення кількості ключів у словнику

За допомогою оператора **del** можна видалити елемент зі словника, вказавши його ключ у квадратних дужках	>>> a = {"sd": "система", "sf": "принтер", "sh": 25}
	>>> del a["sf"]
	>>> a
	{'sd': 'система', 'sh': 25}
		# словник a
		
		# видалення елемента з ключем sf
		# виведення нового словника

Метод **discard** (елемент) видаляє з множини елемент, якщо його значення міститься в множині	>>> a = set([1, 2, 3, 4, 5])
	>>> a.discard(2)
	>>> a
	{1, 3, 4, 5}	

Метод **keys()** повертає об'єкт із ключами словника; його можна використати для виведення всіх елементів словника за допомогою оператора циклу for.	>>> a = {"x":1, "y":2, "z":3}
	>>> for i in a.keys(): print("({0} : {1})". format(i, a[i]), end=" ")
	
	(y:2) (z:3) (x:1)	

Звернемо увагу на те, що елементи словника виведені в довільному порядку. Це зумовлене тим, що словники є невпорядкованими структурами. Якщо потрібно вивести елементи словника впорядкованими за значенням ключа, то слід отримати список ключів, а потім скористатися функцією **sorted()**.

Метод **update()** додає елементи в словник. Одна з найпростіших структур методу така: update (словник)	>>> x = {1:"a", 2:"b"} # словник x
	>>> x.update({3:"c", 4:"d"}) # додавання у словник нових елементів
	>>> x # виведення нового словника
	{1: 'a', 2: 'b', 3: 'c', 4: 'd'}

Метод **values()** повертає об'єкт dict_values, який містить усі значення словника	>>> x = {"a": 1, "b": 2}
	>>> x.values() # об'єкт dict_values ()
	dict_values([1, 2])
	>>> list(x.values()) # отримано список значень
	[1, 2]

Метод **pop**(ключ [, значення за замовчуванням]) вилучає зі словника елемент зі вказаним ключем і повертає його значення. Якщо ключ у словнику відсутній, то повертається значення другого параметра, а якщо відсутній ключ і не вказаний другий параметр, то генерується виняток **KeyError**	>>> x = {"a": 5, "b": 6, "c": 7} # словник x
	>>> x.pop("b"), x.pop("d", 8) # видалення елементів з указаними ключами
	(6, 8)
	>>> x # виведення нового словника
	{'c': 7, 'a': 5}

Приклад 2.

```
>>> x = ["a", "b", "c"]
>>> values = [5, 6, 7]
>>> {m: n for (m, n) in zip(x, values)}
{'c': 7, 'a': 5, 'b': 6}
```

Мова Python підтримує також генератор словників, синтаксис якого схожий на синтаксис генератора списку, але вираз заключається не у квадратні, а у фігурні дужки всередині виразу перед циклом for (приклад 2).

Із прикладами створення програм опрацювання словників можна ознайомитися на сайті interactive.ranok.com.ua

Запитання для перевірки знань

1. Який тип даних називають словником?
2. Як оголошуються словники?
3. Як можна звернутися до елемента словника?
4. За допомогою якого оператора перевіряється наявність ключа в словнику?
5. Як можна видалити елемент зі словника?
6. Наведіть приклад використання генератора словника.

Завдання для самостійного виконання

1. Ринкова ціна на посібники у гривнях така: з інформатики — 45, географії — 53, історії — 40, хімії — 47. Складіть програму визначення мінімальної і максимальної ціни посібників і ціну посібника з хімії.
2. Створіть словник, об'єктами якого є номери потягів і міста, до яких вони слідують: 23 — Львів, 7 — Одеса, 15 — Харків. Додайте до словника два потяги: 37 — Херсон, 29 — Полтава і вилучіть потяг 15 — Харків.

4.4. Масиви

Для розв'язування математичних та інших задач у кожній мові програмування досить часто використовують масиви. У мові Python вони досить подібні до вбудованого в мову структурованого типу «списки», але з обмеженням на тип даних і розмір кожного елемента.

Масив — це структурований тип даних, усі елементи якого мають лише один тип, наприклад int, char та ін.

Структура масиву може бути *одновимірною* (лінійною), *двовимірною* (табличною) та *багатовимірною*. Доступ до елементів масиву отримується відповідно за одним або кількома індексами.

У мові Python масиви визначено в модулі array, який містить величезну кількість методів і функцій їх опрацювання. Таким чином користувачу не потрібно розробляти масиви, слід просто знати, як їх використовувати.

Далі розглянемо та опишемо класичні засоби опрацювання масивів.

У мові **Python** із масивами можна працювати й без модуля **array**. Власне модуль **array** використовується досить рідко, у випадках коли потрібно досягти високої швидкості роботи. В інших випадках масив типу **array** краще замінити іншими типами даних: списками, кортежами, рядками. Найчастіше для представлення масиву в **Python** використовуються списки.

4.4.1. Одновимірні масиви

Чи можна назвати перелік навчальних предметів у 10 класі масивом? Наведіть приклади масивів, із якими вам доводилося стикатися в процесі вивчення географії. Які операції ви виконували над цими масивами?

Загальну структуру одновимірного масиву можна позначити так: x[0], x[1], x[2], …, x[n-1], x[n]. У квадратних дужках зазначено **індекси** (номери позицій) елементів у масиві. Індексами елементів масивів можуть бути дані будь-якого типу, у тому числі вирази, але найчастіше ними є цілі числа.

Будь-якому елементу масиву можна надати нове значення за допомогою оператора присвоювання, наприклад, mas[4]=5 — четвертому елементу одновимірного масиву mas присвоєно значення 5 (це можливо лише у випадку, коли елемент із цим індексом уже існував, інакше — помилка).

Створити масив у мові Python можна різними способами, наприклад, можна вводити значення елементів із клавіатури, обчислювати значення за певною формулою та присвоювати їх елементам масиву й ін.

Розглянемо створення, введення та виведення елементів у масивах на прикладах.

Одновимірний масив — це нумерована послідовність однотипних елементів. Такий масив можна уявити, наприклад, як таблицю, що містить один рядок або стовпець.
Він є динамічним — ви можете змінювати його довжину, додавати або видаляти елементи й ін. Щоб зберігати групу однотипних об'єктів у **Python**, оптимально використовувати тип даних **list**(список).

Приклад 1.

Найпростішим способом створення масиву є перелічення у квадратних дужках значень його елементів праворуч від оператора присвоювання. Наприклад, у результаті виконання оператора a = [5, 9, 3, 12] елементи масиву з іменем a набудуть таких значень: a[0]=5, a[1]=9, a[2]=3, a[3]=12. Зверніть увагу, що нумерація елементів у масивах мови Python починається з нуля.

Приклад 2.

На рис. 1 наведено програмний код створення масиву, елементами якого є рядки. Елементи масиву за допомогою оператора циклу виводяться на екран.

```
# Найпростіший спосіб створення масиву
mas = ['File', 'Edit', 'Run', 'Windows', 'Help']
n = len(mas)              # довжина масиву
for i in range(n):        # цикл перегляду елементів масиву
    print(mas[i])         # виведення елементів масиву
# або
for i in mas:             # цикл перегляду елементів масиву
    print(mas)            # виведення елементів масиву
```

Рис. 1. Створення масиву в процесі його оголошення

Приклад 3.

На рис. 2 зображено програмний код, у якому циклічно обчислюються значення виразу, кожне з яких присвоюється елементу масиву. Після створення масиву елементи виводяться на екран.

```
# Створення масиву шляхом обчислення його елементів
n = int(input('увести довжину масиву: '))
a = int(input('увести число a: '))
g = int(input('увести число g: '))
mas = []                  # порожній масив
for i in range(n):        # цикл формування масиву
    a = a*g               # обчислення виразу
    mas.append(a)         # формування масиву
for i in range(n):        # цикл виведення масиву
    print(mas [i])        # виведення масиву
```

Рис. 2. Створення масиву шляхом обчислення значень його елементів

Приклад 4.

На рис. 3 зображено програмний код, у якому масив створюється шляхом уведення значень його елементів із клавіатури. Після введення масиву обчислюється загальна сума й середнє значення елементів масиву.

```
N = int(input('Кількість елементів: '))
a = []                    # порожній масив
for i in range(N):        # цикл уведення
    b = int(input('Увести число: '))
    a.append(b)           # додавання елементу до масиву
s = 0                     # початкове значення суми
for i in range(N):        # цикл виведення та обчислення
    print('Елемент на позиції ', i, '=', a[i])
    s = s + a[i]          # обчислення суми масиву
c = s / N                 # обчислення середнього значення
print('Середнє значення масиву = ', c)
```

Рис. 3. Уведення масиву з клавіатури та обчислення середнього значення

Запитання для перевірки знань

1. Що називають масивом?
2. Які існують структури масивів?
3. Як можна звернутися до окремого елемента масиву?
4. Яку структуру має одновимірний масив?
5. Якими способами можна створити одновимірний масив?
6. Назвіть порядок створення масиву шляхом уведення даних.

Завдання для самостійного виконання

1. Створіть масив, елементами якого є рядки: 'звук', 'колонка', 'кодування', 'модель'. Виведіть масив на екран.
2. Створіть масив, елементами якого є цілі числа: 21, 40, 53, 17, 33, що вводяться з клавіатури. Виведіть масив на екран.
3. Створіть масив, елементами якого є 10 випадкових чисел у діапазоні від 4 до 10. Виведіть масив на екран.
4. Створіть масив, елементами якого є 5 перших членів арифметичної прогресії. Перший член 3, а різниця 4. Виведіть масив на екран.
5. Створіть масив із семи випадкових чисел у діапазоні від 2 до 6 обчисліть суму елементів.
6. Обчисліть середнє значення масиву, елементами якого є перші шість членів геометричної прогресії. Перший член 3, знаменник 2.

4.4.2. Двовимірні масиви

Пригадайте, з якими двовимірними масивами ви стикалися під час вивчення хімії. Які операції ви виконували в цих масивах?

Двовимірний масив, так само як і одновимірний, може містити елементи будь-якого типу, але лише одного. Масив складається з фіксованої кількості рядків і стовпців. У побуті масиви часто називають таблицями, а в математиці — матрицями. Елементи двовимірного масиву беруться у квадратні дужки, елементи кожного рядка також беруться у квадратні дужки, які відокремлюються комою. Усередині рядка елементи також відокремлюються комою. Подання аналога двовимірного масива в Python найкраще реалізується за допомогою списка списків (тип даних list, кожен елемент якого також типу list).

Двовимірний масив — це ненумерована послідовність однотипних елементів, кожен із яких визначається двома індексами. Такий масив можна подати, наприклад, як таблицю, що містить n рядків і m стовпців.

Приклад 5.

На рис. 4 наведено масив цілих чисел із трьома рядками й чотирма стовпцями.

У мові Python нумерація рядків і стовпців починається з нуля. Масив, який наведено на рис. 4, матиме такий вигляд:

[[34, 23, 6, 9], [35, 11, 54, 23], [15, 40, 3, 44]].

Для наочності краще подати його таким чином:

[[34, 23, 6, 9],
[35, 11, 54, 23],
[15, 40, 3, 44]].

Рядки	Стовпці			
	0	1	2	3
0	34	23	6	9
1	35	11	54	23
2	15	40	3	44

Рис. 4. Масив цілих чисел

Приклад 6.

```
>>> mas = [[34, 23 ,6 ,9],
[35, 11, 54, 23],[15 ,40, 3, 44]]
>>> mas [1][2]
54.
```

Із прикладами 7–9 можна ознайомитися на сайті interactive.ranok.com.ua.

Тут використано прямокутні масиви (кількість стовпців у кожному рядку однакова). Деякі мови, наприклад Java, підтримують масиви з різною кількістю стовпців у рядках, — їх називають **нерегулярними**. Крім того, підтримуються **динамічні** масиви — кількість стовпців і рядків визначається в процесі виконання програми.

Звернення до елемента масиву здійснюється за формою:

<ім'я масиву>[номер рядка][номер стовпця] (приклад 6).

Над елементами двовимірних масивів можуть виконуватися ті самі операції, що й над елементами одновимірних масивів.

- **Обчислення загальної суми й середнього значення елементів масиву.**

Обчислити суму значень елементів масиву можна різними способами. Але найчастіше застосовується алгоритм «накопичення», за яким до початкового значення суми додається елемент, розташований у нульовому рядку нульового стовпця. Потім до отриманої суми поступово додається решта елементів стовпців нульового рядка, здійснюється перехід на наступний рядок, виконуються аналогічні дії і т. д.

- **Обчислення суми значень елементів кожного рядка й загальної суми масиву**

Сума значень елементів рядків двовимірного масиву обчислюють так само, як і для одновимірного. Після завершення обчислення суми елементів одного рядка здійснюється перехід до обчислення суми елементів наступного рядка.

- **Обчислення кількості заданого елемента в масиві**

Алгоритм обчислення кількості заданого елемента у двовимірному масиві відрізняється від аналогічного алгоритму для одновимірного масиву лише тим, що пошук елемента виконується не в одному, а в кількох рядках.

Запитання для перевірки знань

1. Як у мові Python нумеруються рядки та стовпці двовимірного масиву?
2. Який тип можуть мати елементи двовимірного масиву?
3. Як здійснюється звернення до елементів двовимірного масиву?
4. Які операції можуть виконуватися над елементами двовимірного масиву?
5. Сформулюйте алгоритм обчислення загальної суми двовимірного масиву.
6. Поясніть алгоритм пошуку максимального елемента двовимірного масиву.

Завдання для самостійного виконання

1. Із двовимірного масиву [['команда','файл','біт'], ['смартфон','миша','байт']] виведіть на екран перший елемент нульового рядка та другий елемент першого рядка.
2. У двовимірному масиві [[5, 3, 12],[13, 7, 7], [21, 6, 8]] визначте загальну суму його чисел.
3. У двовимірному масиві [[34, 11, 23, 19], [18, 19, 37, 51],[77, 20, 35, 55]] визначте мінімальний елемент у кожному рядку.
4. У двовимірному масиві [[77, 32, 23 ,3], [44, 21, 23, 9], [80, 5, 2, 4]] визначте рядок із мінімальною сумою його чисел.
5. Знайдіть в Інтернеті п'ять кращих футбольних команд України за 2015, 2016 і 2017 роки. Створіть двовимірний масив, елементами якого є назви цих команд. Визначте, скільки разів у ньому є команда «Дніпро».

4.5. Вказівники

Вказівники мають фундаментальне значення в будь-якій мові програмування.

Вказівник — це тип даних, який використовується для зберігання адрес змінних і об'єктів.

Адреса містить номери комірок пам'яті або спеціальне значення (часто нульове), яке свідчить про те, що звернення до комірки пам'яті не може бути виконано. Якщо вказівник містить будь-яку адресу, то кажуть, що він посилається на відповідний об'єкт.

Сутність і принципи використання вказівників у типізованих мовах програмування (Pascal, Java, C++ та ін.) майже ідентичні.

У мовах із динамічною типізацією даних, якою є мова Python, наявна суттєва відмінність — змінні не оголошуються, а вказівник присвоюється об'єктам. Вказівники не мають типу, тип мають об'єкти, на які вони посилаються.

Пояснимо сутність вказівників мови Python на прикладі.

У змінній зберігається не сам об'єкт, а посилання на нього, тобто адреса пам'яті, у якій зберігається об'єкт.

Приклад.

Припустимо, що для обчислення уводиться вираз 8**3. Спочатку в комірки пам'яті заносяться значення 8 і 3, потім обчислюється вираз 8**3 і значення 512 також буде збережено в комірці. Це значення буде виведено на екран і зміст усіх цих комірок буде стерто. Отже, у мові Python автоматично збирається сміття, тобто пам'ять звільняється від усього, що не використовується.

Щоб зберегти об'єкт у пам'яті, потрібно встановити на нього вказівник. Як уже зазначалося, іменування об'єктів реалізується за допомогою оператора присвоювання (наприклад, zmk = 21). Водночас літерал 21 створює в пам'яті об'єкт типу int(), а zmk є вказівником на цей об'єкт.

Таким чином, 21 — це реально існуючий об'єкт зі власними атрибутами, а zmk — вказівник на нього.

Якщо після цього виконати zmk = '21', то буде створено об'єкт str('21'), на який посилається вказівник zmk, а попередній об'єкт 21 буде видалено з пам'яті.

Розпізнати тип об'єкта, на який посилається цей вказівник, можна за допомогою функції type(zmk).

Запитання для перевірки знань

1. Дайте означення вказівника мови програмування.
2. Поясніть сутність вказівника мови Python.

5. Функції користувача та модулі мови Python

5.1. Функції

Пригадаємо, що в алгоритмізації часто застосовуються допоміжні алгоритми, які в мовах програмування реалізуються підпрограмами. Які види підпрограм ви вивчали у 8 і 9 класах?

У багатьох мовах програмування підпрограми оформлюються у вигляді функцій і процедур. У мові Python вони реалізуються лише за допомогою функцій.

У мові Python існує чимало вбудованих функцій, які вже нами неодноразово використовувалися, наприклад, **del(), len()**. Ці функції не потрібно розробляти — їх слід правильно використовувати; їх може створювати і сам програміст. Такі функції називають **користувацькими**.

Користувацька функція — це невеликий, логічно завершений програмний код, до якого можна звертатися багаторазово з різних місць основної програми.

У процесі кожного звернення до функції вона виконує одні й ті самі дії над різними значеннями даних і повертає отримане значення до основної програми. Наприклад, функція обчислення суми членів арифметичної прогресії під час кожного звернення до неї може обчислювати суму її членів із різними значеннями різниці та кількості членів прогресії.

Сутність програми з двома функціями пояснюється схемою (рис. 1).

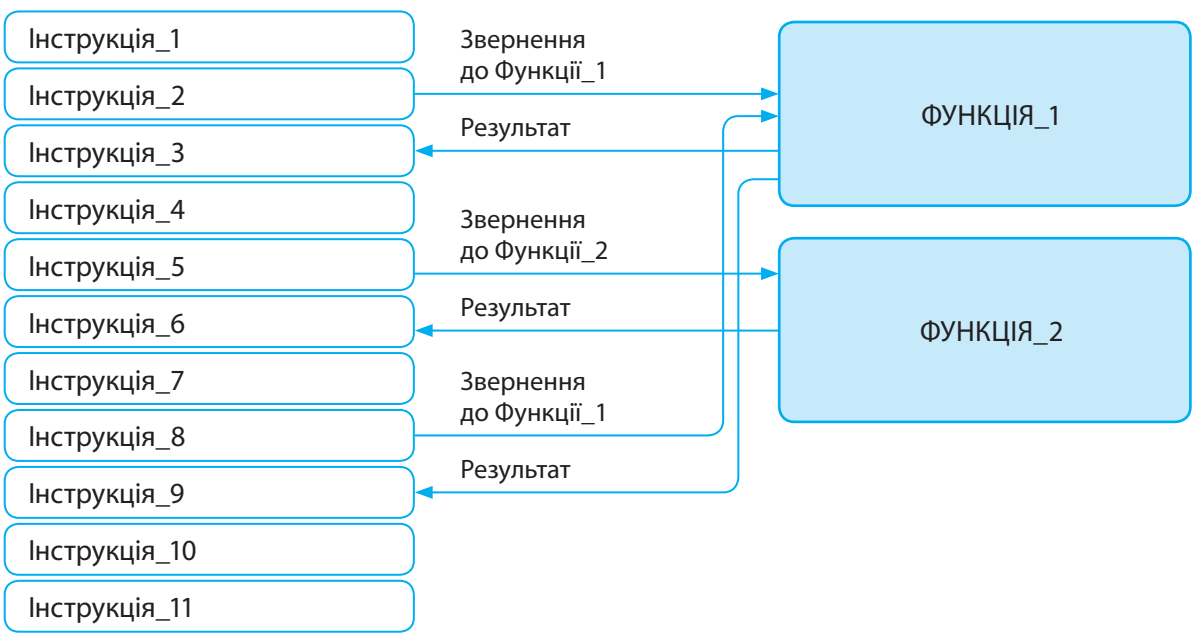


Рис. 1. Приклад структури програми з двома функціями

Як бачимо, основна програма містить 11 інструкцій.

Звернення до першої функції виконується двічі: перший раз з інструкції_2, другий — з інструкції_8. До другої функції звернення виконується один раз — з інструкції_5. Після того

як завершується виконання кожної функції, результат повертається до основної програми в інструкції, що розташовані за інструкцією виклику.

Функції — це універсальний засіб структурування програм, вони дають змогу розбити складну програму на окремі частини, можуть викликатися у виразах і застосовуватися в умовних інструкціях **if** та інструкціях циклу **while**.

Використання функцій позбавляє необхідності вставляти до основної програми копії блоків одного й того самого програмного коду. За рахунок цього зменшується загальний обсяг програми і, відповідно, зменшується обсяг пам'яті, потрібної для її збереження. Окрім цього, зменшуються трудовитрати програміста. Наприклад, якщо необхідно змінити інструкцію, то вона змінюється лише один раз у самій функції, а не в багатьох місцях основної програми.

Користувацькі функції оголошуються (визначаються) за допомогою інструкції def, яка має таку структуру:

```
def  <ім'я функції> ([параметри]):
     <тіло функції>
     [return<результат>]
```

Усі інструкції тіла функції відокремлюються зліва однаковою кількістю пробілів (бажано дотримуватися чотирьох пробілів). Кінцем функції є перша інструкція, яка має меншу кількість пробілів.

Як бачимо, існують функції *з параметрами* і *без параметрів*.

Ім'я функції — це звичайний унікальний ідентифікатор латинськими буквами. Ім'я функції складається з малих символів.

Тіло функції — це сукупність інструкцій, які реалізують певне завдання, наприклад, інструкції обчислення середнього значення списку чисел.

Параметри — це імена об'єктів (змінних, списків тощо), які отримують конкретні значення (атрибути) під час звернення до функцій. Параметри відокремлюються один від одного комою. Якщо функція не має параметрів, зазначаються лише порожні круглі дужки.

Якщо тіло функції не містить інструкцій, слід розмістити оператор pass, який жодних дій не виконує. Наприклад:

```
def  funkt_1:
     pass
```

Інструкція **return** повертає результат виконання функції в основну програму. Інструкції, що розташовані в тілі функції після **return**, не виконуються. Якщо інструкцію **return** не зазначено, повертається значення **None**.

Оголошення функцій визначається зазвичай на початку програми, одразу після імпортування необхідних модулів.

Приклад 1.

На рис. 2 наведено програмний код, у якому оголошено дві функції: функція без параметрів — funct_03 і функція з двома параметрами x і y — funct_04. Звернення до функції funct_03 здійснюється один раз, а до функції funct_04 — двічі. Після звернення до функції funct_03 виконується множення числа 2.3 на число 3.5 і виведення отриманого результату на екран. Далі управління передаватиме в основну програму інструкції a, b = 2, 3, у результаті чого змінна a набуде значення 2, змінна b — значення 3.

```
def funct_03():          # оголошення функції без параметрів
    y = 2.3 * 3.5        # тіло функції, обчислення виразу
    print(y)             # тіло функції, виведення результату
def funct_04(x, y):      # оголошення функції з параметрами
    z = x * x + y * y    # тіло функції, обчислення виразу
    return z             # тіло функції, повернення результату
funct_03()               # звернення до функції funct_03
a, b = 2, 3              # значення аргументів функції funct_04
print(funct_04(a, b))    # перше звернення до функції funct_04
a, b = 3, 4              # нові значення аргументів функції funct_04
print(funct_04(a, b))    # друге звернення до функції funct_04
```

Рис. 2. Програма з двома функціями

У результаті першого звернення до функції funct_04 параметр x набуде значення 2, а параметр y — значення 3. У функції funct_04 є оператор return, тому отриманий результат повертається в програму, що її викликала, і виводиться на екран. Потім змінна x набуває значення 3, а змінна y — 4.

У процесі другого звернення до функції funct_04 змінній x передається значення 3, а змінній y — 4. Отриманий результат повертається в основну програму й виводиться на екран.

Результат виконання програми наведено на рис. 3.

```
8.049999999999999
12
25
```

Рис. 3. Виконання програми з двома функціями

Одна й та сама функція може виконувати різні дії над об'єктами. Яка конкретно дія буде виконуватися, залежить від типу об'єктів, над якими виконуються відповідні операції.

Поліморфізм є одним із фундаментальних понять об'єктно-орієнтованого програмування.

Змінні поділяються на *локальні* та *глобальні*.

Локальні змінні — це змінні, які оголошені всередині функції. У вже розглянутому прикладі у функції funct_03 локальною змінною є y, а у функції funtc_04 — змінні x, y і z. Локальні змінні доступні тільки програмному коду всередині функції. Вони існують тільки під час виконання функції.

Глобальні змінні — це змінні, які оголошені в основній програмі, тобто за межами функції. Вони доступні в будь-якій частині програми, у тому числі всередині функції.

Локальні та глобальні змінні можуть мати однакові імена. Але операції над локальними змінними всередині функцій не впливають на значення однойменних глобальних змінних. Однак для запобігання плутанині краще не користуватися однаковими іменами локальних і глобальних змінних.

Якщо у функції обчислюється значення виразу x + y та ці об'єкти є числами, то буде виконано їх додавання, а якщо об'єктами є рядки — їх конкатенація. Якщо у функції обчислюється значення виразу x * y і ці об'єкти є числами, то буде виконано їх множення, а якщо один із них число, а другий — рядок, то повторення рядка.

Отже, у мові Python саме об'єкти визначають синтаксичну сутність операцій, які будуть виконуватися над ними.

Явище, коли синтаксична сутність операцій залежить від типу об'єктів, які опрацьовуються, називають **поліморфізмом**.

Поліморфними є фактично всі операції у мові Python, а також відповідні функції. Інтерпретатор мови сам перевіряє сумісність типів об'єктів. Якщо виявиться, що вони несумісні, видає повідомлення про синтаксичну помилку.

Приклад 2.

На рис. 4 зображено програму, у якій параметрам функції передаються значення кортежів і списків.

У результаті виконання програми отримаємо:

3.0
6.0

```
def funct_05(a, b, c):          # функція має три параметри
    return (a + b + c) / y      # обчислення і повернення результату
x1 = (2, 3, 4)                  # кортеж
x2 = [5, 6, 7]                  # список
y = len(x1)                     # довжина кортежу
print(funct_05(*x1))            # перше звернення до функції
y = len(x2)                     # довжина списку
print(funct_05(*x2))            # друге звернення до функції
```

Рис. 4. Програма з передаванням функції значень кортежів і списків

Параметрам функцій можна передавати не лише значення літералів, змінних, а й значення кортежів, списків, словників. Якщо параметрам функції передаються значення словника, то в інструкції звернення до функції перед її іменем ставлять два символи зірочка (**). А якщо списку або кортежу, то одну зірочку. Одночасно можна передавати параметрам функції значення різних типів.

Приклад 3.

На рис. 5 зображено програму, у якій під час першого звернення до функції параметрам передаються значення словника, а під час другого звернення — значення кортежу та словника.

```
import math                     # імпортування модуля math
def func_06(a, b, c):           # функція з 3 параметрами
    s = a + b + c               # обчислення суми чисел
return math.sqrt(s)             # обчислення кореня квадратного
d1 ={"a": 1, "b": 2, "c": 3}    # словник
# параметри функції набудуть значень словника
print(func_06(**d1))            # перше звернення до функції
# параметри функції набудуть значень кортежу і словника
t, d2 = (5, 6), {"c": 7}        # кортеж і словник
print(func_06(*t, **d2))        # друге звернення до функції
```

Рис. 5. Програма з передаванням функції значень словника й кортежу

Результат виконання програми наведено на рис. 6.

```
2.449489742783178
4.242640687119285
```

Рис. 6. Виконання програми з передаванням функції значень словника й кортежу

Інструкція **def** у процесі оголошення функції створює об'єкт, який має тип **function**, і зберігає посилання на цей об'єкт в ідентифікаторі, яким є ім'я функції. Це посилання на функцію можна зберегти в іншій змінній.

Приклад 4.

На рис. 7 зображено програму зі зверненням до функції за допомогою змінної.

```
import math                          # імпортування модуля math
import random                        # імпортування модуля random
def func_07(a):                      # функція з одним параметром
    return math.factorial(a)         # обчислення факторіалу
z = func_07                          # посилання на об'єкт у змінній z
x = random.randint (3, 9)            # x - випадкове число від 3 до 9
print(x)                             # виведення випадкового числа
print(z(x))                          # перше звернення до функції
x = random.randint(2, 7)             # x - випадкове число від 2 до 7
print(x)                             # виведення випадкового числа
print(z(x))                          # друге звернення до функції
```

Рис. 7. Програма з посиланням на функцію за допомогою змінної

Структуру виведення та можливі значення виконання програми наведено на рис 8.

```
9
362880
5
120
```

Рис. 8. Виконання програми з посиланням на функцію за допомогою змінної

Запитання для перевірки знань

1. Які функції називають користувацькими?
2. Що називають тілом функції?
3. Яка різниця між аргументами й параметрами?
4. Чи завжди функції мови Python мають тіло функції?
5. Для чого застосовується оператор pass?
6. Поясніть сутність звернення до функції із основної програми.
7. Які переваги надає використання функції у програмуванні?
8. Яку роль виконує інструкція return у тілі функції?
9. Поясніть різницю між локальними та глобальними змінними.
10. Поясніть сутність поліморфізму.

Завдання для самостійного виконання

1. Складіть програму з використанням функції без параметрів для обчислення об'єму конуса.
2. Складіть програму з використанням функцій із параметрами для обчислення двох різних за розміром прямокутних трикутників із відомими значеннями їх катетів і обчисліть площі кола з відомим радіусом.
3. Кубик із цифрами від 1 до 6 підкидають п'ять разів. Складіть програму обчислення середнього значення цих чисел із використанням функції з параметрами.
4. Складіть програму обчислення значення виразу a*b-c/d із використанням функції з параметрами для значень списку [3, 5, 2, 7] і кортежу (4, 1, 5, 6).
5. Генеруються три цілі випадкові числа в діапазоні від 3 до 7. Складіть програму з використанням функції з параметрами для обчислення суми квадратів цих чисел.
6. З'ясуйте в Інтернеті, через які області України протікає Дніпро, та знайдіть їх площі. Складіть програму визначення областей із максимальною та мінімальною площами. Обчисліть середню площу. Використайте функції з параметрами.

5.1.2. Розширені можливості функцій

Який, на вашу думку, недолік мають уже описані користувацькі функції? Чи відомо вам про випадок, коли кількість параметрів і аргументів була неоднаковою?

Пригадаємо, що в розглянутих прикладах кількість параметрів в оголошенні функції і кількість аргументів у інструкції звернення була однаковою. Водночас значення першого аргументу передається першому параметру функції, значення другого аргументу — другому параметру.

Наприклад, якщо оголошено функцію def sum(x, y):

```
def sum(x, y):
return x + y,
```

і виконується до неї звернення: sum(a, b), то значення аргументу *a* передається параметру *x*, а значення *b* — параметру *y*.

Такий варіант є основним (класичним), і застосовується в багатьох мовах програмування. Разом із тим, можливості функцій мови Python значно ширші за можливості класичних функцій.

У випадку 1) функція може бути реалізована за умови, що значення деяким параметрам присвоєне в процесі оголошення функції.

Якщо оголошено функцію def sum(x, y=5):

```
def sum(x, y=5):
return x + y,
```

то до неї можна звернутися так: sum(a). До того ж параметр *x* набуде значення *a*, параметр *y* — значення 5.

Параметри, яким присвоюються значення в процесі оголошення функції, називають необов'язковими. Необов'язкові параметри повинні розміщуватися після обов'язкових.

Зокрема мова Python дає змогу:

- в інструкції звернення до функції мати меншу кількість аргументів, аніж кількість параметрів, зазначених у процесі оголошення функції;
- передавати параметрам значення аргументів не послідовно, а в довільному порядку;
- використовувати змінну кількість параметрів у функції;
- під час оголошення не вказувати ідентифікатор функції.

Приклад 5.

На рис. 9 зображено програму з функцією обов'язкових і необов'язкових параметрів.

Під час першого звернення до функції число 8 передається тільки параметру x (параметр y матиме значення 3). Під час другого звернення параметр x набуде значення 27, а параметр y — значення 4, незважаючи на те, що в оголошенні функції йому присвоєно значення 3. У результаті <виконання програми> отримаємо числа: 2.0 і 3.0.

```
import math                    # імпортування модуля math
def func_08(x, y=3):           # функція з необов'язковим параметром
    return math.fmod (x, y)    # остача від ділення x на y
a = func_08(8)                 # число 8 передається параметру x
print(a)                       # виведення результату 1-го звернення
# параметру x передається число 27, параметру y - число 4
print(func_08(27,4))           # виведення результату 2-го звернення
```

Рис. 9. Програма з обов'язковими й необов'язковими параметрами функції

У другому випадку в інструкції звернення до функції вказуються ті самі ідентифікатори аргументів, що й ідентифікатори параметрів, а також присвоюються відповідні їм значення.

Наприклад, у фрагменті програми:

```
def ant(x, y):
    return x * y
print(ant(y=7, x=5))
```

параметр при x набуде значення 5, а параметр при y — значення 7. Такий варіант доцільно використовувати, якщо функція має кілька необов'язкових параметрів.

Розглянемо функції, у яких параметрам можна передавати довільну кількість значень аргументів. Для того щоб параметри могли набувати довільної кількості аргументів, їх у оголошенні функції необхідно вказувати з символом зірочка (*).

Приклад 6.

На рис. 10 зображено програму з функцією, параметр якої може набувати довільної кількості аргументів. У цій програмі в процесі першого звернення до функції параметр x набуває двох значень: 9 і 6, а в процесі другого звернення — чотирьох значень: 3, 4, 7, 10. У результаті виконання програми після першого звернення до функції отримаємо значення 20, а після другого — 29.

```
def func_09(*x):                # параметр зі змінною кількістю аргументів
    s = 5                       # початкове значення змінної
    for i in x:                 # цикл
        s = s + i               # обчислення суми s+=i
        return s                # повернення результату
print(func_09(9, 6))            # параметру передаються 2 оргументи
print(func_09(3, 4, 7, 10))     # параметру передаються 4 аргументи
```

Рис. 10. Програма з функцією зі змінною кількістю аргументів

В одній функції можна використовувати комбінації різних параметрів: *звичайні* (обов'язкові) параметри, *необов'язкові* (параметри за замовчуванням) і параметри зі змінною *кількістю аргументів*.

Приклад 7.

Програму з такою функцією зображено на рис. 11.

У процесі першого звернення до функції параметр при x набуває значення 7, параметр при y за замовчуванням має значення 4, а параметр при z — значення 0. Тому інструкція s=s+i в операторі циклу не виконується.

У процесі другого звернення до функції параметр при x набуває значення 5, параметр при y — значення 8, а параметр при z — двох значень: 9 і 12. Таким чином, вираз s=s+i виконується двічі: для значення змінної при i, яка дорівнює 9 і 12. У результаті виконання програми отримаємо значення 15 і 42.

```
def func_10(x, y=4, *z):              #комбінація різних параметрів
    s = 2 * y + x                     #обчислення виразу для значень x і y
    for i in z:                       #цикл for
#вираз s=s+i не виконується під час першого звернення до функції
#вираз s=s+i виконується двічі під час другого звернення до функції
        s = s + i                     #обчислення суми для значень z
    return s                      #повернення результату
print(func_10(7))                     #перше звернення до функції
print(func_10(5, 8, 9, 12))           #друге звернення до функції
```

Рис. 11. Програма, функція якої має комбінацію параметрів

Значення, що повертається, — це вираз, результат виконання якого повертається функцією. Це значення можна зберегти в змінній або передати як параметр у іншу функцію.

Мова Python реалізує *анонімні* функції, які ще називають *лямбда-функціями*. Ці функції не мають імені. Вони оголошуються за допомогою ключового слова lambda за такою структурою:

lambda [<параметр1> [,…,<параметрN>]]:
<значення, що повертається>

Приклад 8.

На рис. 12 зображено програму з анонімними функціями.

```
a1 = lambda: 5 * 7 + 9                  # функція без параметрів
a2 = lambda: x, y: x * x + y / 2        # функція з 2 араметрами
a3 = lambda: x, y, z = 3: x / y + 2 * z # функція з необов'язковим параметро
print(a1())                             # звернення до першої функції
print(a2(7, 8))                         # звернення до другої функції
print(a3(5, 6))                         # звернення до третьої функції
```

Рис. 12. Програма з анонімними функціями

У результаті виконання програми отримаємо такі результати (рис. 13):

44,
53.9,
6.83.

Рис. 13. Виконання програми з анонімними функціями

Анонімні функції викликаються так само, як і звичайні функції. Вони можуть бути з параметрами та без параметрів. Як і у звичайних функціях, деякі параметри анонімних функцій можуть бути необов'язковими.

Запитання для перевірки знань

1. Як передаються значення аргументів необов'язковим параметрам?
2. Як у функції розміщуються необов'язкові та обов'язкові параметри?
3. Які функції називають анонімними?
4. Як оголошують анонімні функції?
5. Як реалізується передавання параметрам у довільному порядку значення аргументів?
6. Як можна реалізувати передавання параметрам довільної кількості значень аргументів?
7. Поясніть порядок використання у функції комбінації різних параметрів.

Завдання для самостійного виконання

1. Складіть програму обчислення значення виразу a * b2 – 2 * a, якщо a > b і виразу a / b, якщо a <= b із використанням функції з одним обов'язковим та одним необов'язковим параметрами.
2. Складіть програму з використанням однієї функції зі змінною кількістю аргументів для обчислення сум чисел 1, 3, 5, 7, 9 і 6, 8, 10.
3. Складіть програму з використанням анонімних функцій без параметрів і з параметрами для обчислення значень виразів 3 / 7 + 5 * 8 / 3, 2 * x + x / y і (3 * x + 5) / (y – 2) для відомих значень x і y.
4. Функція має три параметри: обов'язковий, необов'язковий, зі змінною кількістю аргументів. Складіть програму обчислення добутку чисел 3 на 7 і добутку чисел 4, 6, 7, 9, 10.
5. У змаганнях із підняття гирі від 10-А класу беруть участь 5 учнів, а від 10-Б класу — 4 учні. Відомо скільки разів підняв гирю кожен учень. Розробіть програму визначення різниці піднять гирі учнями обох класів.
6. Прибуток від двох фірм Кагарлицького району Київської області постійний і складає, відповідно, 10 і 15 млн грн на рік. Ще від двох фірм цього району прибуток визначається наприкінці року. Постійний прибуток також мають дві фірми Обухівського району Київської області й становить, відповідно, 9 і 17 млн грн на рік, а від однієї фірми прибуток визначається наприкінці року. Розробіть програму обчислення загального прибутку фірмами з кожного району. Виберіть і обґрунтуйте раціональний варіант програмного коду.

5.2. Рекурсивні функції

У 8 і 9 класах ви вже ознайомилися з терміном "рекурсія". Спробуйте сформолювати означення рекурсії.

Приклад 1.
Речення «Тип об'єктів визначає синтаксичний смисл оператора» може бути розширено до такого: «Сашко зрозумів, що тип об'єктів визначає синтаксичний смисл оператора», а останнє речення розширено до такого: «Тепер Катря знає, що Сашко зрозумів, що тип об'єктів визначає синтаксичний смисл оператора».

Рекурсія (від латин. *recursio* — повернення) у широкому розумінні — це опис або зображення об'єкта (процесу, явища) через самого себе. Рекурсія застосовується в різних галузях людської діяльності, найчастіше — в математиці й інформатиці.

У лінгвістиці рекурсією називають можливість мови породжувати нові мовні конструкції на основі попередньої (приклад 1).

Класичним прикладом рекурсії в математиці є визначення чисел Фібоначчі, кожне з яких обчислюється як сума двох найближчих попередніх чисел, починаючи з числа, розташованого на третій позиції (n>2), тобто таких, які обчислюються за допомогою формули f(n) = f(n–2) + f(n–1), де n>2. Ці числа мають такі значення: 1, 1, 2, 3, 5, 8, 13, 21...

Як бачимо з прикладу 2, на кроці 4 знайдено опорне значення функції f(3), яке дорівнює 2, а потім у зворотному порядку слід виконати відповідні дії, тобто

2 + 1 = 3;
3 + 2 = 5;
5 + 3 = 8.

Рекурсія в інформатиці — це спосіб організації обчислювального процесу, за яким програма в процесі виконання звертається сама до себе з різними значеннями вхідних параметрів. Цей процес може бути нескінченним, тому для переривання цього процесу в програмі повинна бути умова його переривання.

У стеку спочатку запам'ятовуються всі дії, що виконуються до моменту обчислення опорного значення, а потім у зворотному порядку вони виконуються.

Наприклад, якщо рекурсивна функція обчислює числа Фібоначчі, то запам'ятовуються всі дії f(n) = f(n–2) + f(n–1) до тих пір, поки не буде знайдене опорне значення, а потім у зворотному порядку виконується обчислення.

Отже, для обчислення рекурсивних функцій потрібен стек і виконання щонайменше удвічі більше операцій, ніж, наприклад, у процесі використання рекурентних обчислень.

Пригадаємо, що рекурентне обчислення значення функції на кожному кроці здійснюється через її значення на попередньому кроці.

Уже відомий вам алгоритм *рекурентного обчислення виразу* $y = x^n$ можна подати так:

Крок 1	Увести n, x
Крок 2	y = 1
Крок 3	i = 1
Крок 4	y *= x
Крок 5	i += i
Крок 6	Якщо i <= n, то п. 4, інакше п. 7
Крок 7	Виведення y
Крок 8	Кінець

Приклад 2.
Щоб знайти значення числа на шостій позиції (число 8), потрібно спочатку запам'ятати виконання таких дій:

1) f(6) = f(5) + f(4);
2) f(5) = f(4) + f(3);
3) f(4) = f(3) + f(2);
4) f(3) = f(2) + f(1).

Програми обчислення рекурсій оформлюються у вигляді підпрограм (у мові **Python** — функцій). Реалізація таких функцій заснована на структурі даних, що називають стеком.

У наведенному алгоритмі після того, як вираз **i<=n** на кроці 6 набуде значення **False**, виконання алгоритму завершується і змінна **y** міститиме результат обчислення.

Для рекурсивного обчислення значення x^n (n — додатне ціле, x — може бути дійсне) необхідно обчислити значення x^{n-1}, оскільки $x^n = x^{n-1} \cdot x$. У свою чергу, для обчислення x^{n-1} потрібно обчислити x^{n-2}, оскільки $x^{n-1}=x^{n-2} \cdot x$ і т. д. Процес обчислення припиняється після обчислення x^0, оскільки воно має значення 1.

Усі ці дії запам'ятовуються в стеку. Але отримано не результат, а лише опорне значення функції. Для отримання кінцевого значення функції потрібно реалізувати зворотний процес обчислення, що міститься в стеку.

Приклад 3.

Розглянемо сутність рекурсії на прикладі обчислення значення виразу x^n. Програму його обчислення зображено на рис. 1.

```
#рекурсивне обчислення числа x у степені n
def st(x, n):                       # заголовок функції
    if n == 0:                      # чи дорівнює число нулю
        return 1                    # повернення одиниці
    else:                           # початок обчислення числа в степені
        return st(x, n-1)*x         # обчислення і повернення результату

print("уведіть число x")            # повідомлення про уведення числа x
x = int(input())                    # уведення числа x
print("уведіть число n")            # повідомлення про введення числа n
n = int(input())                    # уведення числа n
print("результат =", st(x, n))      # звернення до рекурсивної функції
```

Рис. 1. Програма обчислення значення виразу x^n

Опишемо порядок виконання програми.

1. В основній програмі уводяться числа x і n. Припустимо, що уведені значення x=2 і n=3. Після цього здійснюється звернення до функції st із параметрами st(2, 3). У результаті в стеку виділяється перший блок пам'яті (вершина стеку) й управління передається умовному оператору. Оскільки на цьому етапі вираз n -= 0 набуває значення False, то виконується гілка else. Тут у виразі st(x, n–1)*x робиться спроба обчислити значення st(2, 2), тобто виконується звернення до самої функції st із новими значеннями параметрів. Тому обчислення переривається.

2. Другий раз звернення до функції st виконується з параметрами st(2, 2). У стеку також виділяється наступний блок пам'яті, а в умовному операторі також виконується гілка else і робиться спроба обчислити значення функції st із параметрами st(2, 1).

3. Третій раз звернення до функції st виконується з параметрами st(2, 1). У стеку також виділяється блок пам'яті й знову виконується гілка else з параметрами st(2, 0).

4. Четвертий раз виклик функції виконується з параметрами st(2, 0). У стеку виділяється четвертий блок пам'яті, але виконується гілка then, а функція st набуває значення 1. У результаті функція st більше не викликається і здійснюється повернення в основну програму, яка викликала цю функцію.

5. Далі зі стеку вибирають опорне значення st(2, 0), яке дорівнює одиниці, оскільки в стеку реалізується принцип «останній прийшов — перший вийшов».

6. На цьому кроці обчислюється значення st(2, 1), яке дорівнює 2, потім значення st(2, 2), яке дорівнює 4, нарешті — st(2, 3), яке дорівнює 8. Це значення повертається в основну програму.

Динаміку описаного процесу «заглиблення» й виходу зі стеку для описаного прикладу наведено в таблиці (с. 73).

Рівень рекурсії	Рекурсивне «заглиблення»	Рекурсивне повернення
0	Виклик st(2, 3) ↓	→→Вихід st = 8
1	st(2, 3) = st(2, 2)*2 ↓	↑ st(2, 3) = 2*4 (8)
2	st(2, 2) = st(2, 1)*2 ↓	↑ st(2, 2) = 2*2 (4)
3	st(2, 1) = st(2, 0)*2 ↓	↑ st(2, 1) = 1*2 (2)
4	st(2, 0) = 1 →→	→

Класичним варіантом рекурсивної функції є обчислення факторіалу заданого числа n, яке виконується за формулою n! = n * (n–1)!, якщо n>0, і n! = 1, якщо n=0.

Програму обчислення факторіалу наведено на рис. 2.

```
# програма обчислення факторіалу уведеного числа
def fact(n):                        # заголовок функції
 if n == 0:                         # пошук опорного значення
      return 1                      # повернення опорного значення
 else:                              # починається обчислення факторіалу
      return n * fact(n-1)          # обчислення і повернення факторіалу
print("уведіть число")              # повідомлення про уведення числа
n = int(input())                    # введення числа
print(fact(n))                      # звернення до рекурсивної функції
```

Рис. 2. Програма обчислення факторіалу з використанням рекурсивної функції

Приклад 4.

Дано два цілі додатні числа. Розробити програму обчислення найбільшого спільного дільника з використанням рекурсивної функції.

Програма знаходження найбільшого спільного дільника двох цілих чисел без використання рекурсивних підпрограм уже розглядалася раніше. Розв'язання задачі засноване на використанні алгоритму Евкліда. Програму реалізації цього алгоритму з використанням рекурсивної функції зображено на рис. 3. Функція має ім'я nod, а числа, для яких обчислюється найбільший спільний дільник, позначені змінними a і b.

```
# рекурсівне обчислення найбільшого спільного дільника
def nod(a, b):                      # заголовок функції
    if b == 0:                      # перевірка чисел на рівність
        return a                    # повернення числа a
    else:                           # починається обчислення nod
        return nod(b, a % b)        # обчислення і повернення числа nod
    print("уведіть число a")        # повідомлення про уведення числа a
a = int(input())                    # уведення числа a
    print("уведіть число b")        # повідомлення про уведення числа b
b = int(input())                    # уведення числа b
    print("nod = ", nod(a, b))      # звернення до рекурсивної функції
```

Рис. 3. Програма обчислення найбільшого спільного дільника

Рекурсія є складним процесом, який вимагає додаткової пам'яті та часу її реалізації. Тому частіше використовується рекурентне обчислення. Однак без рекурсії інколи обійтися досить складно, оскільки без неї алгоритм має заплутану логіку.

Запитання для перевірки знань

1. Що називають рекурсією в програмуванні?
2. Із якою метою в тілі рекурсії передбачається умовний оператор?
3. На якій структурі даних засновано реалізацію рекурсивних функцій?
4. Які дефекти має рекурсивне обчислення?
5. Поясніть сутність «заглиблення» та виходу зі стеку в процесі реалізації рекурсії.

Завдання для самостійного виконання

1. Розробіть програму з рекурсивною функцією обчислення n перших членів геометричної прогресії, перший член якої дорівнює a і знаменник q.
2. Розробіть програму з використанням рекурсивної функції обчислення кількості цифр у десятковому натуральному числі a.
3. Для одновимірного числового масиву довжиною n розробіть програму з використанням рекурсивної функції обчислення добутку значень його елементів.

5.3. Модулі

Які переваги, на вашу думку, надає модульний принцип будови програмного коду? Із якими модулями стандартної бібліотеки мови Python ви вже стикалися?

Ми вже розглядали загальну структуру програми мовою Python, яка складалася з кількох файлів. Кожен файл є модулем. Використовувалися окремі модулі зі стандартної бібліотеки, але сутність процесів імпортування модулів не описувалася.

Програма мовою Python складається зі множини текстових файлів. Один із файлів програми є головним, до нього підключаються (імпортуються) додаткові файли, тобто модулі.

Програма запускається з головного файла, який і визначає порядок виконання інших модулів. Файли модулів містять компоненти, які використовує головний файл, а модулі використовують компоненти, що визначені в інших модулях. Але для того щоб їх використати, ці модулі слід імпортувати.

Розглянемо сутність імпортування модулів і використання їх компонентів на прикладі структури найпростішої програми (рис. 1).

У модулі визначаються імена (функції, змінні тощо), які називають **атрибутами модулів**. Вони стають доступними іншим модулям після їх імпортування. Навіть у тому випадку, коли вдається програму логічно оформити в одному модулі, часто доводиться використовувати в ньому вже наявні модулі, зокрема модулі стандартної бібліотеки.

Рис. 1. Структура найпростішої програми

Із рис. 1 видно, що програма складається з модуля головного файла (a.py), додаткового модуля (b.py) і модулів стандартної

бібліотеки. Припустимо, що в модулі b.py визначена функція без параметрів funct_1 (у цьому випадку ця функція є атрибутом модуля b.py):

```
def funct_1():
    y = 7.7 / 3.2
    print(y)
```

Для того щоб функцію funct_1 можна було використовувати в модулі a.py, спочатку необхідно імпортувати модуль b.py до модуля a.py, а потім можна її використовувати за допомогою інструкції b.funct_1().

Імпортування модуля реалізується інструкцією import <ім'я модуля>. За допомогою однієї інструкції import можна імпортувати кілька модулів, наприклад, імпортувати модулі math і random можна за допомогою інструкції:

```
import math, random
```

Після імпортування модуля можна отримати доступ до його атрибутів, які відокремлюються від назви модуля крапкою, наприклад, math.pi:

```
>>> import math
>>> math.pi
3.141592653589793
```

Якщо атрибут не знайдено, генерується виняток AttributeError. Для того щоб цей виняток не генерувався, можна у функції getattr() указати значення за замовчуванням, наприклад:

```
>>> import math
>>> print(getattr (math, "ant", "Відсутній"))
Відсутній
```

Значення атрибута модуля можна отримати також за допомогою функції getattr () за його назвою.
Функція має такий формат:
getattr(модуль, назва атрибута [,значення за замовчуванням]):

```
>>> import math
>>> print(getattr(math, "pi"))
3.141592653589793
```

Перевірити наявність атрибута в модулі можна за допомогою функції hasattr(модуль, назва атрибута). Якщо атрибут існує, повертається значення True.

```
>>> import math
>>> hasattr(math, «pi»)
True
```

Інколи доцільно користуватися не іменем модуля, а його псевдонімом. Наприклад, псевдоніми доцільно застосовувати, якщо ім'я модуля довге. Але якщо оголошено псевдонім модуля, то після цього іменем модуля користуватися не можна. Псевдонім указується в інструкції імпортування модуля за такою структурою: as <ім'я модуля>:

```
>>> import math as p
>>> print(p.pi)
3.141592653589793
```

Стандартна бібліотека Python містить колекцію майже з 200 різноманітних модулів, які забезпечують незалежну платформу підтримки багатьох задач програмування (пошук за шаблоном, створення графічного інтерфейсу й ін.). На цьому етапі нам потрібні лише модулі math і random, модуль sys, що містить інформацію про середовище виконання програм інтерфейсу Python.

Якщо в програму необхідно імпортувати не весь модуль, а лише окремі його ідентифікатори, то можна скористатися інструкцією from. Один із можливих її форматів є таким:

```
from <ім'я модуля>  import <ідентифікатор1> [as <псевдонім1>]
    [,…,<ідентифікаторN> [as <псевдонімN>]]
```

Приклад.

```
>>> from random import randint, uniform as un      # імпортування двох функцій
>>> print(randint(2, 10))                          # звернення до функції randint
3
>>> print(un(1, 9))                                # звернення до функції uniform через псевдонім
1.207729134588777
```

Із наведеного прикладу видно, що різниця між інструкціями import і from полягає також і в порядку звернення до атрибутів імпортованих модулів. Якщо імпортування модуля виконується за допомогою інструкції import, для звернення до його атрибутів необхідно вказувати через крапку ім'я цього модуля, наприклад, math.pi. Якщо ж імпортування здійснюється за допомогою інструкції from, указувати ім'я модуля не потрібно.

Відзначимо також, що в процесі першого запуску програми під час виконання інструкції імпортування модуля спочатку здійснюється його пошук, потім трансляція в байт-код, після чого запускається програмний код модуля. Під час наступних запусків ці дії не виконуються, а програмний код модуля просто вибирається з пам'яті.

Такий варіант застосовується для невеликих програм, зокрема для програм навчального призначення, які зберігаються в одній папці. Система в такому разі сама автоматично здійснює їх пошук. Цей варіант вважається стандартним шляхом пошуку файлів. Файли реальних програм можуть зберігатися в різних папках і тому в процесі імпортування необхідно вказувати шлях до кожного з них. Ці питання тут не розглядаються.

Зазначимо, що в розглянутих прикладах програм використовувалося імпортування окремих модулів і їх атрибутів. Але імпортувати можна також і каталоги, які в мові Python називають пакетами. Найчастіше пакети застосовуються для групування модулів за їх функціональним призначенням.

Пакети мають деякі специфічні особливості, характерні для мови Python. Вони застосовуються переважно в професійному програмуванні й тому тут не розглядаються.

У процесі збереження розроблених файлів програмного коду розширення .py не застосовувалося. Це пояснюється лише тим, що програми містили лише головні файли програм, які не передбачалося імпортувати до інших модулів. Але для файлів, які планується імпортувати в інші, розширення .py слід обов'язково вказувати.

Запитання для перевірки знань

1. Для чого в мові Python застосовується імпортування модулів?
2. Який файл називають головним файлом програми?
3. За допомогою яких інструкцій імпортуються модулі?
4. Назвіть складові програми мовою Python.
5. Поясніть на прикладі сутність імпортування модулів.
6. Із якою метою застосовуються псевдоніми модулів?
7. Поясніть різницю між інструкціями import і from.
8. За допомогою якої інструкції перевіряється наявність атрибута в модулі?
9. У яких випадках у файлі програмного коду не застосовується розширення .py?

6. Класи, об'єкти, наслідування

Основний дефект процедурного стилю програмування полягає в тому, що він слабко пристосований для розроблення великих програм. Із часом програми стають настільки складними, що розібратися в них непросто. Одним із напрямків розв'язування цієї проблеми є об'єктно-орієнтоване програмування (ООП).

6.1. Елементи теорії об'єктно-орієнтованого програмування

Із якими об'єктами ОПП ви вже ознайомилися?

ООП базується на трьох **основних принципах**:

- наслідування;
- поліморфізм;
- інкапсуляція.

Ці принципи є досить абстрактними та складними для розуміння на початкових етапах ознайомлення з мовою. В ООП загалом використовується значна кількість нових термінів. Щоб свідомо опанувати їх, розглядатимемо їх поступово на прикладах.

До основних понять об'єктно-орієнтованого програмування належать **об'єкт**, **клас**, **метод**.

Для загального ознайомлення наведемо основні терміни.

Клас (Class) — об'єкт, що складається із сукупності методів і змінних (атрибутів), які описують цей об'єкт.

Метод (Method) — сукупність інструкцій мови опрацювання даних.

Змінна класу (Class variable) — звичайна змінна, що визначається всередині класу (а не всередині методів класу), яка є доступною для всіх екземплярів цього класу.

Екземпляр класу (Instance) — окремий об'єкт класу.

Змінна екземпляра класу (Instance variable) — змінна, що визначена всередині методу класу і яка належить тільки цьому класу.

Сутність ООП у різних джерелах тлумачиться по-різному. Здається, що найбільш удало та просто описав її Алан Кей. Він зазначив, що мову можна назвати ООП, якщо вона відповідає певним вимогам.

Мова ООП має відповідати таким вимогам:

- усі дані подаються об'єктами;
- програма є набором об'єктів, які взаємодіють між собою та надсилають один одному необхідні повідомлення;
- кожен об'єкт може мати у своєму складі інші об'єкти і для всіх них виділяється власна частина пам'яті;
- кожен об'єкт належить одному типу (класу), який задає поведінку об'єктів, створених на їх основі;
- об'єкти одного типу можуть виконувати одні й ті самі дії.

Алан Кертіс Кей — американський інформатик, відомий своїми працями в галузі ООП, президент дослідного інституту В'юпоїнта, ад'юнкт-професор інформатики в Каліфорнійському університеті (США). Кей сказав, що найкращий спосіб спрогнозувати майбутнє — винайти його.

Використання принципів ООП у мові Python не є обов'язковим. Залежно від особливостей задачі, програму можна розробляти на основі принципів як процедурного програмування, так і ООП. Така унікальність мови дає змогу розробляти досить компактні програми, адже не для кожної задачі ОПП є оптимальним.

Гібридна мова поєднує властивості як процедурного, так і об'єктно-орієнтованого програмування. Саме до них належить мова Python.

Вимоги до мов ООП у різних мовах програмування реалізуються по-різному: найскладніше в мовах Java і C++, значно простіше — у мові Python. Уже розроблені нами програми складалися з окремих частин (фрагментів) на основі функцій і модулів, до яких можна було звертатися багаторазово. Зважаючи на те, що в цих програмах застосовувалися об'єкти та методи (що є базовими поняттями ООП), такі програми не можна в повному розумінні віднести до ООП.

Досі ми не використовували такі фундаментальні поняття, як клас, екземпляр класу, наслідування й ін. А без них про реалізацію ООП не може навіть ітися.

Переваги ООП стають відчутними лише в процесі розв'язування великих і складних завдань. Для багатьох задач середньої складності, особливо навчального призначення, не тільки можна, а й доцільно застосовувати гібридні мови.

Така особливість мови Python, як гібридність, дала змогу на початкових етапах її вивчення основну увагу приділити синтаксичним конструкціям, функціям і модулям, без яких не може існувати не лише процедурне, а й об'єктно-орієнтоване програмування. Розуміння цих питань допоможе тепер якісно оволодіти й методами ООП.

Характерними особливостями ООП мовою Python можна вважати такі:

- основною складовою мови ООП є клас, головне призначення якого — створення та маніпулювання об'єктами, а також підтримка механізму наслідування як способу адаптації програмного коду для розв'язування однотипних задач і неодноразового його використання;
- об'єкти мови Python нагадують убудовані типи, які було описано раніше;
- програма складається з окремих, достатньо незалежних частин, що суттєво полегшує її розуміння й читання;
- нова програма може розроблятися не «з нуля», а шляхом адаптування вже наявного програмного коду, без його зміни.

Під час кожного звернення до класу створюється новий об'єкт — екземпляр класу, тому можна отримати необмежену кількість екземплярів на основі цього класу.

Отже, основою ООП є клас — це складний тип даних із певним набором змінних (атрибутів) і функцій (методів) опрацювання значень, що зберігаються в цих змінних.

Запитання для перевірки знань

1. На яких основних принципах базується ООП?
2. Для яких задач відчутні переваги ООП?
3. Яким основним вимогам має відповідати ООП?
4. Які мови називають гібридними?
5. Які фундаментальні поняття застосовуються в ООП?
6. Назвіть основні особливості ООП мовою Python.

6.2. Створення класів і об'єктів

Пригадайте, які елементи ООП ви вже використовували.

У мові Python реалізація принципів ООП практично зводиться до виконання виразу: <об'єкт>.<атрибут>. Цей вираз уже неодноразово використовувався нами для доступу до атрибутів модуля, звернення до методів об'єктів тощо. Але реалізація таких виразів відбувалася за відсутності в програмі класів.

Якщо програма містить класи, то створюється дерево пов'язаних об'єктів. У такому разі в процесі реалізації виразу <об'єкт>.<атрибут> інтерпретатор починає пошук зазначеного атрибута в дереві пов'язаних об'єктів у такому порядку: спочатку атрибут відшукується в цьому самому об'єкті, потім — у дереві об'єктів знизу догори і зліва направо. Пошук завершується, щойно буде знайдено першу появу атрибута.

Тіло класу містить змінні та функції класу. Функції в ООП називають **методами**, а змінні класу — **атрибутами**. Атрибути створюються як звичайні змінні, а методи — як звичайні функції за допомогою інструкції def.

Об'єкти в програмі можна створювати лише на основі класів, тому програма починається зі створення класу. Класи можна розробляти самостійно або імпортувати з інших модулів. Вони розміщуються на початку програмного коду.

Щоб визначити, який саме екземпляр класу слід опрацьовувати, у методі обов'язково слід указати перший параметр з іменем self (інші параметри можуть бути відсутні). Цьому параметру автоматично передається посилання на відповідний екземпляр класу.

За допомогою параметру self здійснюється доступ до змінних екземпляра класу та методів класу всередині визначеного методу. Параметр self відокремлюється від змінної або методу крапкою. Наприклад, до змінної ask із методу класу можна звернутися так: self.ask.

Класи створюються за допомогою інструкції class і мають таку загальну структуру:
class <ім'я класу> [(класи, що є нащадками)]:
<тіло класу>

Структуру класу можна подати детальніше:

```
class <ім'я класу> ([суперклас_1, суперклас_2,...]):
    змінна = значення
    def ім'я методу (self,...):
        self.змінна = значення
...
```

Необхідно чітко розуміти, що в результаті виконання інструкції class буде створено новий екземпляр класу, який також є об'єктом, посилання на який присвоюється імені змінної. Якщо цей клас має батьківські класи (суперкласи), від якого він є нащадком, то вони вказуються в круглих дужках через кому. Якщо він не є нащадком, то круглі дужки не вказуються.

Методи можна розглядати як невеликі програми, призначені для опрацювання об'єктів, тобто вони створюють нові властивості об'єктів, змінюють наявні тощо. Методи фактично є аналогами підпрограм у мовах процедурного програмування.

Для того щоб можна було використовувати атрибути та методи класу, необхідно створити екземпляр класу за такою структурою:

<екземпляр класу> = <ім'я класу> ([параметри])

Метод викликається з основного блоку програми за допомогою інструкції:
<екземпляр класу>.<ім'я методу> ([параметри])

Тут <екземпляр класу> є звичайним ідентифікатором. У результаті кожного виконання цієї інструкції у пам'яті комп'ютера виділяється ділянка пам'яті, доступ до якої здійснюється за допомогою пов'язаного з ним ідентифікатора, тобто в пам'яті створюється об'єкт. Цей об'єкт має характеристики того класу, у якому його створено, тобто він отримує атрибути свого класу.

Під час кожного звернення до класу створюється новий екземпляр класу з різними значеннями атрибутів. Наприклад, якщо в класі обчислюється об'єм піраміди з висотою h, основою якої є квадрат зі стороною a, то під час кожного звернення до класу буде створюватися такий самий об'єкт з атрибутами h і a, але значення їх будуть різними.

Зазвичай класи без методів не використовуються.

Приклад 1.

Для усвідомлення сутності ООП спочатку розробимо найпростішу програму без методу (рис. 1).

```
class Kl7_01:          # клас з іменем Kl7_01
    a = "гарно"        # змінна a класу kl7_01

ob1 = Kl7_01()         # екземпляр ob1 класу Kl7_01
ob2 = Kl7_01()         # екземпляр ob2 класу Kl7_01
print(ob1.a)           # виведення значення змінної a екземпляра ob1
print(ob2.a)           # виведення значення змінної a екземпляра ob1
```

Рис. 1. Найпростіша програма без методу

У цій програмі оголошено змінну (атрибут) з іменем a і два екземпляри ob1 і ob2 класу Kl7_01. Імена екземплярів різні, тому для них буде відведено дві ділянки пам'яті, але їх зміст однаковий і має значення змінної a, тобто гарно. Виведення атрибута цих об'єктів здійснюється двічі за допомогою вбудованої функції print. У результаті буде двічі виведено слово гарно.

Додамо в цю програму метод з іменем func і параметром self (рис. 2). В інструкції цього методу виводиться значення змінної a, до якого додається слово і правильно.

```
class Kl7_02:                          # клас з іменем Kl7_02
    a = "гарно"                        # змінна a класу kl7_02
    def func(self):                    # метод func із параметром self
        print(self.a + "і правильно")          # інструкція методу

ob1 = Kl7_02()                         # екземпляр ob1 класу Kl7_02
ob2 = Kl7_02()                         # екземпляр ob2 класу Kl7_02
# виведення значення змінної a екземпляра ob1 за допомогою вбудованої функції
print(ob1.a)
# виведення значення змінної a екземпляра ob2 за допомогою вбудованої функції
print(ob2.a)
ob1.func() # виклик методу func екземпляра ob1 для виведення нового значення
ob2.func() # виклик методу func екземпляра ob2 для виведення нового значення
```

Рис. 2. Найпростіша програма з одним методом

Метод func викликається в програмі двічі: з екземплярів класу ob1 і ob2. Результат виведенння зображено на рис. 3.

```
гарно
гарно
гарно і правильно
гарно і правильно
```

Рис. 3. Виконання програми з одним методом

Як бачимо у програмі (див. рис. 3) використовувався один метод лише зі стандартним параметром self.

Приклад 2.

Розглянемо програму з двома методами, у кожному з яких, окрім параметра self, використовується ще один параметр (рис. 4.).

```
class Kl7_03:                          # клас Kl7_03
    predm = "інформатика"              # змінна predm за межами методів
    klas = 8                           # змінна klas за межами методів
    def func_1(self, a1):              # метод func_1 з параметром a1
        self.predm = a1                # зміна значення змінної predm
    def func_2(self, a2):              # метод func_2 з параметром a2
        self.klas = a2                 # зміна значення змінної klas

ob1 = Kl7_03()                         # екземпляр ob1 класу Kl7_03
ob2 = Kl7_03()                         # екземпляр ob2 класу Kl7_03
# виведення значень змінних predm і klas екземпляра ob1
print(ob1.predm, ob1.klas)
# виведення значень змінних predm і klas екземпляра ob2
print(ob2.predm, ob2.klas)
# звернення до методу func_1 екземпляра ob1 зі значенням аргументу "математика"
ob1.func_1("математика")
# звернення до методу func_1 екземпляра ob2 зі значенням аргументу "історія"
ob2.func_1("історія")
# звернення до методу func_2 екземпляра ob2 зі значенням аргументу 9
ob2.func_2(9)
# виведення значень змінних predm і klas після їх зміни в методі func_1
print(ob1.predm, ob1.klas)
# виведення значень змінних predm і klas після їх зміни в методі func_2
print(ob2.predm, ob2.klas)
```

Рис. 4. Програма з двома методами, у кожному з яких два параметри

Основна ідея цієї програми полягає в демонстрації порядку зміни властивостей створених об'єктів за допомогою методів. Дві змінні predm і klas оголошені поза межами методів. Тому обидва об'єкти (ob1 і ob2) набувають властивості, які мають ці змінні, тобто інформатика і 8. А потім за допомогою методів ці властивості змінюються.

У програмі метод func_1 має параметри self і a1, а метод func_2 — параметри self і a2. Ці методи виконують тільки одну дію, а саме: метод func_1 змінює значення змінної predm, а метод func_2 — значення змінної klas.

У програмі створено два екземпляри класів: ob1 і ob2. Поза межами методів змінна predm набуває значення «інформатика», а змінна klas — значення 8. За допомогою інструкцій print (ob1.predm, ob1.klas) і print (ob2.predm, ob2.klas) здійснюється виведення значень цих змінних.

Зважаючи на те, що в першій інструкції змінні викликаються з екземпляра ob1, а в другій інструкції вони викликаються з екземпляра ob2, значення змінних однакове, тому двічі буде виведено інформатика 8.

Далі за допомогою інструкції ob1.func_1 («математика») з екземпляра ob1 викликається метод func_1 з аргументом математика. Цей аргумент передається параметру a1, а за допомогою інструкції self.predm = a1 змінна predm набуває нового значення, а саме — математика.

Звернемо увагу, що значення змінної klas екземпляра ob1 не змінювалося, тому за допомогою інструкції print (ob1.predm, ob1.klas) буде виведено математика 8.

Після цього за допомогою інструкцій ob2.func_1 («історія») і ob2.func_2 (9) викликаються методи func_1 і func_2 з екземпляра ob2. У результаті обидві змінні змінюють своє значення і буде надруковано історія 9. Результат виконання програми наведено на рис. 5.

```
інформатика 8
інформатика 8
математика 8
історія 9
```

Рис. 5. Виконання програми з двома методами

Розглянемо ще один приклад розв'язування такої задачі.

Приклад 3.

Дано два конуси різних розмірів із відомими радіусом основи й висоти. Потрібно обчислити об'єм одного з них і загальну площу другого. Програму розв'язування цього завдання зображено на рис. 6.

```
class Kl7_04:                                  # клас Kl7_04
  def func_1 (self, a1, a2):                   # метод func_1 з параметрами self, a1 і a2
      volume = 1/3 * (3.14 * a1 * a1 * a2)     # обчислення об'єму конуса
      return volume                            # повернення результату
  def func_2 (self, a3, a4):                   # метод func_2 з параметрами self, a3 і a4
      v = a3 * a3 + a4 * a4                    # сума квадратів радіуса і висоти
      surface =  3.14*a3*(a3+math.sqrt(v))     # обчислення повної площі конуса
      print("площа = ", surface)               # виведення площі конуса

import math                                    # імпортування модуля math
ob1 = Kl7_04()                                 # екземпляр ob1 класу Kl7_04
ob2 = Kl7_04()                                 # екземпляр ob2 класу Kl7_04
# звернення до методу func_1 і виведення результату
print ("об'єм = ", ob1.func_1(3, 4))
ob2.func_2(4, 5)                               # звернення до методу func_2
```

Рис. 6. Програма з двома методами, кожен із яких має три параметри

Для обчислення об'єму конуса використовують метод func_1 із параметрами self, a1 і a2, а для обчислення повної поверхні конуса — метод func_2 із параметрами self, a3 і a4.

Звернемо увагу, що в програмі відсутні змінні класу, тобто змінні за межами методів. У процесі звернення до методу func_1 параметру a1 передається число 3, а параметру a2 — число 4. Результат обчислення об'єму конуса повертається в інструкцію, з якої відбулося звернення до методу, тобто до інструкції print ("об'єм = ", ob1.func (3, 4)).

Отже, результат обчислення буде виведено на екран. Після звернення до методу func_2 параметр a3 набуде значення 4, а параметр a4 — значення 5. Результат обчислення не повертається в інструкцію, що його викликала, а виводиться за допомогою інструкції print ("площа = ", surface) у цьому самому методі.

Результат виконання програми наведено на рис. 7.

```
об'єм = 37.67999999999999
площа = 130.66324042215658
```

Рис. 7. Виконання програми з двома методами, кожен із яких має три параметри

Запитання для перевірки знань

1. За допомогою якої інструкції створюється клас?
2. Які основні складові містить тіло класу?
3. Що називають методом класу?
4. Із якою метою в методі класу використовується параметр self?
5. За допомогою якої інструкції створюється метод класу?
6. Як створюється об'єкт класу?
7. Яка інструкція використовується для звернення до методу класу?

Завдання для самостійного виконання

1. Змінна *a1* набуває значення «інформатика», а змінна *a2* — «фізика». Створіть програму з двома об'єктами, але без методу, за допомогою якої з використанням першого об'єкта виводиться слово інформатика, а з використанням другого — фізика.
2. Змінна *a1* набуває значення «байт». Створіть програму з двома об'єктами і функцією. Метод викликається спочатку з першого, потім — із другого об'єкта. Кожного разу виводиться значення змінної *a1*.
3. Змінна *a* набуває значення «Україна», а змінна *b* — «Франція». За допомогою інструкції input() уводиться довільне ціле число. Розробіть програму без методів, у якій створюються два об'єкти. Якщо уведене число більше 5, виводиться значення з використанням першого об'єкта, інакше — з використанням другого об'єкта.
4. Створіть програму, у якій одна змінна набуває значення «файл», а друга — «папка». Метод, який викликається з обох об'єктів, виводить ці слова.
5. Розробіть програму з двома методами. У першому з них обчислюється площа (у га), що припадає на одного мешканця чи мешканку Дніпропетровської області, а в другому — Хмельницької області.
6. Розробіть програму з використанням двох методів. В одному з них обчислюється об'єм кулі радіусом *r*, *a* в другому — площа трикутника зі сторонами *a*, *b*, *c*.

6.3. Конструктор класу

Пригадаємо, що ініціалізація змінних класу здійснювалася під час кожного створення його екземплярів, що є суттєвим дефектом. Для його подолання в мові Python застосовується конструктор класу.

У мові Python є спеціальний метод __init__ (зліва і справа по два знаки підкреслення), який називають **конструктором класу**.

Цей метод викликається автоматично під час кожного створення екземпляра класу. Якщо метод __init__ містить параметри, то в інструкції створення екземпляра класу в круглих дужках указуються аргументи, які, відповідно, передаються параметрам.

У нашому випадку екземпляр класу створюється за такою структурою:

<екземпляр класу>=<ім'я класу>(аргумент1,…,аргументN)

Метод __init__ викликається автоматично. Наразі виконуються такі дії. Посилання на екземпляр класу передається першому параметру методу __init__ (таким параметром є self). Це означає,

Загальна форма методу конструктором класу така:

```
def __init__(self
[,<параметр1>[,…,<параметрN>]]):
    <тіло методу>
```

що параметр self указує на об'єкт, із якого викликається метод, а всі інші значення аргументів, перелічені в круглих дужках, передаються іншим параметрам методу.

Під час створення екземпляра класу щоразу викликається метод __init__. Новий екземпляр класу передається як аргумент параметру self, а значення аргументів — відповідним параметрам методу.

Значення параметрів у методі __init__ можна присвоювати не тільки в процесі створення об'єкта, а й безпосередньо в самому методі. Наприклад:

```
def __init__(self, a1="файл", a2=25):
```

Приклад 1.

На рис. 1 зображено найпростішу програму, у якій демонструється сутність описаних процесів:

```
class Kl7_05:                          # клас Kl7_05
    def __init__(self, a1, a2):        # конструктор класу
        self.p1 = a1  # змінна екз. класу p1 набуває значення параметру a1
        self.p2 = a2  # змінна екз. класу p2 набуває значення параметру a2

# аргумент Вивчаємо передається параметру a1, аргумент інформатику - a2
ob1 = Kl7_05("Вивчаємо", "інформатику")    # екземпляр класу з аргументами
print(ob1.p1, ob1.p2)                  # Виведення значень змінних екз. Класу
```

Рис. 1. Найпростіша програма з конструктором

Перш за все відзначимо, що змінні p1 і p2 є змінними екземпляра класу, в цьому випадку — екземпляра ob1. Тепер чітко зрозуміла різниця між змінними класу та змінними екземпляра класу. За допомогою інструкції ob1 = Kl7_05 («Вивчаємо», «інформатику») створюється екземпляр класу, аргументами якого є слова Вивчаємо й інформатику.

Під час створення екземпляра класу автоматично викликається метод __init__, у результаті чого посилання на екземпляр ob1 буде присвоєно параметру self, слово Вивчаємо передається параметру a1, а слово інформатику — параметру a2. Наступними інструкціями (self.p1=a1 і self.p2.a2) значення цих слів присвоюються, відповідно, змінним p1 і p2 екземпляра ob1. У процесі виконання інструкції print (ob1.p1, ob1.p2) буде виведено Вивчаємо інформатику. Отже, атрибути (змінні) p1 і p2 екземпляра класу ініціалізуються при створенні саме екземпляра класу.

Приклад 2.

Для свідомішого розуміння сутності конструктора й доцільності його використання складемо програму, яка виконує ті самі дії, що й попередня програма, але без конструктора (рис. 2).

```
class Kl7_06:                          # клас Kl7_06
    def stm(self, a1, a2):             # метод stm з параметрами
        self.p1 = a1                   # змінна p1 екз. класу набуває значення параметру a1
        self.p2 = a2                   # змінна p2 екз. класу набуває значення параметру a2

ob1 = Kl7_06()                         # екземпляр класу ob1 класу Kl7_06 без аргументів
ob1.stm("Вивчаємо","інформатику")      # звернення до методу stm
print(ob1.p1, ob1.p2)                  # виведення значень змінних p1  p2
```

Рис. 2. Програма без конструктора

Як бачимо, ця програма складніша. У ній екземпляр класу створюється без аргументів і необхідно викликати метод stm, у якому значення його аргументів присвоюються змінним екземпляра, які потім виводяться на екран.

Якщо метод викликається з екземпляра класу без аргументів, то значення, що вказані в методі, зберігаються. Якщо ж у екземплярі класу вказані аргументи, то вони передаються в метод, а значення, що вказані в методі, не враховуються.

Приклад 3.

На рис. 3 зображено програму, у якій демонструються сформульовані положення.

```
class Kl7_07:                                          # клас Kl7_07
    def __init__(self, a1="Харків", a2="Одеса"): # конструктор
        self.p1 = a1  # змінна p1 екз. класу набуває значення параметра a1
        self.p2 = a2  # змінна p2 екз. класу набуває значення параметра a2

ob1 = Kl7_07("Київ", "Полтава")   # екз. класу ob1 з двома аргументами
ob2 = Kl7_07("Львів")             # екз. класу ob2 з одним аргументом
ob3 = Kl7_07()                    # екз. класу ob3 без аргументів
print(ob1.p1, ob1.p2)             # виведення Київ Полтава
print(ob2.p1, ob2.p2)             # виведення Львів Одеса
print(ob3.p1, ob3.p2)             # виведення Харків Одеса
```

Рис. 3. Програма з різними варіантами конструктора

Звернемо увагу, що екземпляр класу ob3 не має аргументів, тому вказані в методі значення параметрів (Харків і Одеса) зберігаються. В екземплярі ob2 є тільки один аргумент (Львів). Тому змінюється лише значення параметра a1, а значення параметра a2 залишається незмінним. На рис. 4 наведено результат виконання програми.

```
Київ Полтава
Львів Одеса
Харків Одеса
```

Рис. 4. Виконання програми з різними варіантами конструктора

Конструктор може одночасно містити як параметри, яким не присвоюються значення, так і параметри, яким присвоюються відповідні значення. У такому разі параметри, яким не присвоюються значення, вказуються першими, а параметри, яким присвоюються значення — після них.

Приклад 4.

На рис. 5 зображено програму з такими параметрами.

```
class Kl7_08:                                  # клас Kl7_08
    def __init__(self, a1, a2=10):       # конструктор
        self.p1 = a1      # змінна p1 екз. класу набуває значення параметра a1
        self.p2 = a2      # змінна p2 екз. класу набуває значення параметра a2
ob1 = Kl7_08("принтер", "P1006")       # екз. класу ob1 з двома аргументами
ob2 = Kl7_08("Windows")   # екз. класу ob2 з одним аргументом
print(ob1.p1, ob1.p2)     # виведення значень змінних екз. класу ob1
print(ob2.p1, ob2.p2)     # виведення значень змінних екз. класу ob2
```

Рис. 5. Програма з різними параметрами

Результат виконання програми наведено на рис. 6.

```
принтер P1006
Windows 10
```

Рис. 6. Виконання програми з різними параметрами

Приклад 5

Проводиться жеребкування чемпіонату Європи з футболу серед 16 команд, які залежно від отриманого номера потрапляють в одну з чотирьох груп. Якщо команда отримує номер менший 5, вона потрапляє в першу групу, якщо номер більший 4, але менший 9, то в другу групу, якщо номер більший 9, але менший 13, то в третю групу, інакше — в четверту групу. На рис. 7 зображено програму реалізації результатів жеребкування для чотирьох команд.

```
class K17_09:                                  # клас K17-09
    def __init__(self, a1, a2, a3):           # конструктор
        self.p1 = a1        # змінна p1 екз. класу набуває значення параметра a1
        self.p2 = a2        # змінна p2 екз. класу набуває значення параметра a2
        self.func1(a3)      # звернення до функції func1 з певного екз. класу
    def func1(self, a3):    # функція func1 з двома параметрами
        if a3 <= 4:         # якщо значення a3 менше 4
            self.p3 = "-перша група"     # значення змінної p3 певного екз. кл.
        elif 4 < a3 <= 8:                # якщо 4<a3<=8
            self.p3 = "-друга група"     # значення змінної p3 певного екз. кл.
        elif 8 < a3 <= 12:               # якщо 8<a3<=12
            self.p3 = "-третя група"    # значення змінної p3 певного екз. кл.
        else:                            # якщо значення параметра a3 більше 12
            self.p3 = "-четверта група" # значення змінної p3 певного екз. кл.

ob1 = K17_09("Динамо", "Київ", 7)          # екземпляр класу ob1 з 3 аргументами.
ob2 = K17_09("Реал", "Мадрид", 3)          # екземпляр класу ob2 з 3 аргументами.
ob3 = K17_09("Баварія", "Мюнхен", 9)       # екземпляр класу ob3 з 3 аргументами.
ob4 = K17_09("Інтер","Мілан", 5)           # екземпляр класу ob4 з 3 аргументами.
print(ob1.p1, ob1.p2, ob1.p3)              # виведення даних про Динамо
print(ob2.p1, ob2.p2, ob2.p3)              # виведення даних про Реал
print(ob3.p1, ob3.p2, ob3.p3)              # виведення даних про Баварію
print(ob4.p1, ob4.p2, ob4.p3)              # виведення даних про Інтер
```

Рис. 7. Програма жеребкування команд чемпіонату Європи з футболу

Після запуску програми в процесі створення екземпляра ob1 класу K17_09 викликається конструктор, у результаті чого параметр self набуває значення екземпляра класу ob1, параметр a1 — значення Динамо, параметр a2 — значення Київ, параметр a3 — значення 7. Потім викликається функція func1 із параметром a3. У ній визначається, до якої групи потрапляє команда «Динамо» і назва цієї групи запам'ятовується в змінній p3 екземпляра класу ob1. Далі виконуються аналогічні дії для екземплярів класу ob2, ob3 і ob4. У результаті для кожного об'єкта буде виділено ділянки пам'яті, у яких будуть зберігатися значення змінних p1, p2 і p3. Наступними чотирма інструкціями значення змінних кожного екземпляра класу виводяться на екран. Результат виконання програми наведено на рис. 8.

```
Динамо Київ – друга група
Реал Мадрид – перша група
Баварія Мюнхен – третя група
Інтер Мілан – четверта група
```

Рис. 8. Виконання програми жеребкування команд ліги чемпіонату Європи з футболу

Запитання для перевірки знань

1. Що називають конструктором класу в мові Python?
2. Яку загальну структуру має метод __init__?
3. У яких випадках викликається метод __init__?
4. Які дії виконуються в процесі створення об'єкта і наявності методу __init__?
5. Що передається параметру self у процесі створення об'єкта?
6. Наведіть загальну структуру інструкції створення об'єкта класу.
7. Як можна присвоїти значення параметрам методу __init__ у самому методі?

Завдання для самостійного виконання

1. Створіть програму з конструктором, за допомогою якої із набору «Київ», « — столиця», «України» створюється повідомлення Київ — столиця України.
2. Створіть програму з конструктором і методом обчислення значення виразу (a2 + b2) / 2, якщо значення a і b уводяться з клавіатури.
3. Конструктор має такі значення параметрів: self, a1 = 3, a2 = 4. Створіть програму множення значень параметрів конструктора, за умови, що він викликається — з першого об'єкта без аргументів, з другого об'єкта — з аргументом 5 і 6, а з третього — з аргументом 7.
4. Розробіть програму, у якій конструктор має параметри self, a1 і a2. Він викликається з двох об'єктів, аргументами яких є два цілих числа. Якщо значення першого аргумента більше другого, ці числа множаться, інакше — значення першого ділиться на друге.
5. Конструктор має такі значення параметрів: self, a1 = "Дністер" a2 = "Десна". Розробіть програму виведення всіх можливих результатів на екран, у якій конструктор викликається з першого об'єкта з аргументами Дніпро та Ворскла, з другого об'єкта — з аргументом Дніпро, та з третього об'єкта без аргументів.
6. Конструктор має параметри зі значеннями: self, a1 = "Операційна" a2 = "система". За допомогою інструкції input() уводиться слово. Якщо це слово Linux, виводиться повідомлення Операційна система Linux, інакше — повідомлення Операційна система Windows.

6.4. Наслідування

У різних частинах програмного коду часто використовуються одні й ті самі атрибути, що суттєво впливає на загальний обсяг програми. Як, на вашу думку, можна цього позбавитися?

Наслідування в мові Python — це здатність об'єктів класу застосовувати атрибути цього самого класу, а також здатність одними класами застосовувати атрибути інших класів.

Розглянемо спочатку здатність об'єкта класу наслідувати атрибути цього самого класу.

Приклад 1.

На рис. 1 подано програму обчислення площ двох прямокутних трикутників із відомими катетами.

```
class K17_10:                              # клас K17_10
    def __init__(self , a1, a2,a3):       # конструктор
        self.p1 = a1                  # змінна p1 набуває значення параметра a1
        self.p2 = a2                  # змінна p2 набуває значення параметра a2
        self.p3 = a3                  # змінна p3 набуває значення параметра a3
        self.func1 ()                 # звернення до функції func1
    def func1(self):                  # функція func1 з параметром self
        s = (self.p2*self.p3)/2       # обчислення площі трикутника
        print(self.p1, s)             # виведення площі трикутника

ob1 = K17_10("площа першого =", 3, 4) # екземпляр класу ob1 з 3 аргументами
ob2 = K17_10("площа другого =", 5, 6) # екземпляр класу ob2 з 3 аргументами
ob1.form = "перший зафарбований"      # властивість екземпляра класу ob1
print(ob1.form)                       # виведення властивості екз. класу ob1
```

Рис. 1. Програма з наслідуванням об'єктами атрибутів одного класу

У програмі обидва екземпляри класу ob1 і ob2 наслідують властивості p1, p2 і p3, а екземпляр класу ob1 наслідує ще й властивість form. Екземпляр класу ob2 цієї властивості не має, оскільки атрибути класу наслідуються об'єктами, створеними лише на його основі. Атрибути конкретного об'єкта не залежать від атрибутів інших об'єктів і мають власний простір імен об'єкта. Це означає, що об'єкти одного класу можуть мати різні значення атрибутів. Результат виконання програми наведено на рис. 2.

```
площа першого = 6.0
площа другого = 15.0
перший зафарбований
```

Рис. 2. Виконання програми з наслідуванням об'єктами атрибутів одного класу

Розглянемо тепер приклад програми з наслідуванням атрибутів одного класу іншим класом.

Клас, від якого наслідується інший клас, називають суперкласом, а клас, що унаслідується, — підкласом. В оголошенні підкласу в круглих дужках указуються ім'я суперкласу, від якого він наслідується.

Приклад 2.

На рис. 3 зображено програму, у якій клас Kl7_11 є суперкласом, а клас Kl7_11a — підкласом. Клас Kl7_11a має доступ до всіх атрибутів класу Kl7_11.

```
class Kl7_11:                          # суперклас Kl7_11
    def func1(self, a1, a2):           # метод func1 суперкласу
        s = a1 + a2                    # сума чисел
        print("сума =",s)              # виведення суми
    def func2(self, a1, a2):           # метод func2 суперкласу
        s = a1 * a2                    # добуток чисел
        print("добуток =", s)          # виведення добутку
class Kl7_11a(Kl7_11):                 # клас Kl7_11a наслідує клас Kl7_11
    def func3(self, a1, a2):   # метод func3 підкласу
        s = a1 / a2                # ділення чисел
        print ("ділення =", s) # виведення результату ділення
ob1 = Kl7_11a()                        # екземпляр підкласу
ob1.func1(2, 3)                        # звернення до методу func1 суперкласу
ob1.func2(4, 5)                        # звернення до методу func2 суперкласу
ob1.func3(6, 7)                        # звернення до методу func3 підкласу
```

Рис. 3. Програма з наслідуванням методів одного класу іншим класом

У цій програмі клас Kl7_11a має доступ до методів func1 і func2 класу Kl7_11, а також до власного методу func3. Результат виконання програми подано на рис. 4.

```
сума =5
добуток = 20
ділення = 0.8571428571428571
```

Рис. 4. Виконання програми з наслідуванням методів одного класу іншим класом

Якщо ім'я методу в суперкласі збігається з іменем методу підкласу, то в процесі звернення до нього буде використовуватися метод із підкласу. Щоб викликати однойменний метод із суперкласу, слід указати також перед цим методом назву суперкласу. У першому параметрі методу необхідно вказати посилання на екземпляр класу.

Приклад 3.

Відомі всі три сторони двох прямих контейнерів (паралелепіпедів), а також їх реальна вага. Обчисліть об'єм кожного контейнера, а також можливість їх використання. Контейнери можна використовувати за умови, що їх реальна вага не перевищує контрольну, значення якої уводиться з клавіатури. Програму реалізації завдання зображено на рис. 5.

```
class Kl7_12:                   # суперклас Kl7_12
    contr = int(input("увести контрольну вагу : "))
    def __init__(self, a1, a2, a3):          # конструктор
        self.p1 = a1            # змінна p1 набуває значення параметра a1
        self.p2 = a2            # змінна p2 набуває значення параметра a2
        self.p3 = a3            # змінна p3 набуває значення параметра a3
    def func1(self):            # функція func1 класу Kl7_12
        v = self.p1 * self.p2 * self.p3      # обчислення об'єму
        print("Об'єм", m, " =", v)           # виведення об'єму

class Kl7_12a(Kl7_12):          # підклас kl7_12a суперкласу Kl7_12
    def func2(self, vaga):      # функція func2 класу Kl7_12a
        if vaga < Kl7_12.contr:              # реальна вага менше контрольної?
            self.p4 = "можна використовувати" #
        else:                   # вага більше контрольної
            self.p4 = "використовувати не можна"   #
    def func3(self):            # функція func3 класу Kl7_12a
        print(self.p4)          # виведення результату

ob1 = Kl7_12a(2, 3, 4)          # екземпляр ob1 класу Kl7_12a
m = "першого"                   # ознака номера паралелепіпеду
ob1.func1()                     # звернення до функції func1 із екз. кл. ob1
ob2 = Kl7_12(3, 4, 5)           # екземпляр ob2 класу Kl7_12
m = "другого"                   # ознака номеру паралелепіпеда
ob2.func1()                     # звернення до функції func1 із екз. кл. ob2
ob1.func2(5)                    # звернення до функції func2 із екз. кл. ob1
ob1.func3()                     # звернення до функції func3 із екз. кл. ob1
```

Рис. 5. Програма з наслідуванням атрибутів одного класу іншим класом

У програмі використано такі змінні: vaga — реальна вага контейнера, contr — контрольна їх вага, p1, p2, p3 — розміри сторін контейнерів, m — змінна ознаки номера контейнерів, v — об'єм поточного контейнера, p4 — результат можливості використання контейнерів. Програма складається з двох класів: суперкласу Kl7_12 і підкласу Kl7_12a. Підклас наслідує обидва методи суперкласу (__init__ і func1), а також змінну contr. Екземпляр класу ob1 належить підкласу Kl7_12a і наслідує всі методи цього класу та суперкласу. Екземпляр класу ob2 належить суперкласу. До підкласу Kl7_12a він жодного стосунку не має й тому не може звертатися до методів func2 і func3.

Опишемо порядок виконання програми на прикладі.

Приклад 4.

1. Спочатку в суперкласі виконується інструкція contr = int (...). Після уведення цілого числа виконання програми продовжується. Під час створення екземпляра класу ob1 автоматично викликається метод __init__, у результаті чого аргумент 2 передається параметру a1 цього методу, аргумент 3 — параметру a2, аргумент 4 — параметру a3. Наступними трьома інструкціями значення цих параметрів присвоюються, відповідно змінним p1, p2 і p3 екземпляра класу ob1.

2. Далі управління передається інструкції m = «першого» і далі здійснюється звернення до функції func1. У результаті обчислюється й виводиться на екран об'єм першого паралелепіпеда. Після цього створюється екземпляр класу ob2 і виконуються дії, аналогічні діям, що виконувалися в процесі створення екземпляра класу ob1. У результаті буде обчислено й виведено значення об'єму другого контейнера.

3. Далі управління передається інструкції ob1.func2. За допомогою методу func2 перевіряється можливість використання контейнерів і відповідне значення присвоюється змінній p4. Після цього викликається метод func3, за допомогою якого відповідний результат виводиться на екран.

Один із варіантів виконання програми може бути таким, як наведено на рис. 6.

```
увести контрольну вагу : 8
Об'єм першого = 24
Об'єм другого = 60
можна використовувати
```

Рис. 6. Варіант виконання програми

Запитання для перевірки знань

1. Які можливості надає наслідування в мові Python?
2. Які класи називають суперкласами й підкласами?
3. Як оголошується підклас, який унаслідується від суперкласу?
4. Наведіть приклад наслідування атрибутів в одному класі.
5. Як здійснити звернення до однойменних методів у підкласі й суперкласі?
6. Наведіть приклад наслідування одного класу іншим.

Завдання для самостійного виконання

1. Розробіть програму без конструктора, у якій один клас наслідує атрибути іншого. У суперкласі два числа множаться, а в підкласі виводиться результат множення.
2. Обчисліть площі двох прямокутних трикутників із відомими значеннями катетів із використанням суперкласу, підкласу та конструктора. У суперкласі обчислюються площі трикутників, а в підкласі — виведення значень обчислених їх площ.
3. Конструктор має параметри self, a1, a2, який викликається з двох об'єктів. Перший об'єкт має аргументи «Київ» і «Вінниця», другий — «Львів» і «Харків». Розробіть програму виведення значень параметрів конструктора для обох об'єктів, а також виведення значення першого об'єкта: міста України.
4. Розробіть програму з суперкласом і підкласом. У суперкласі за допомогою одного методу обчислюється сума значень трьох параметрів методу, а за допомогою другого — їх добуток. У підкласі від суми значень перших двох параметрів віднімається значення третього. Звернення до всіх методів виконується з одного об'єкта з трьома аргументами.
5. Червона куля має радіус r1, а жовта — r2. Розробіть програму з використанням принципу наслідування, за допомогою якої визначається різниця об'ємів між червоною і жовтою кулею.
6. З'ясуйте, які обласні центри України розташовані на широтах між 49° і 48°. Розробіть програму визначення різниці в широтах між ними і Києвом. Для обчислення й виведення різниці використайте суперклас, підклас і конструктор.

7. Поліформізм, перевизначення методів, модулі користувача

7.1. Поліморфізм

Пригадайте, із якими принципами об'єктно-орієнтованого програмування ви вже знайомі. Поясніть їх сутність.

Поліморфізм (від грец. πολύς — багато, μορφή — форма) означає "багато форм". Кожна мова ООП має свої особливості поліморфізму.

Розглянемо основні прояви поліморфізму в мові Python.

- **Тип об'єктів визначає синтаксичний смисл оператора, який виконується над об'єктами.**

Оператор + є лише вказівкою для об'єктів, а яка буде виконуватися над об'єктами операція, залежить від типу самих об'єктів (приклад 1).

Приклад 1.
Якщо операція додавання виконується над двома об'єктами типу int, то буде виконуватися операція їх додавання, а якщо об'єкти мають тип str — то операція об'єднання рядків.

Приклад 2.
На рис. 1 зображено програму, у якій демонструється виконання операторів додавання та множення над об'єктами типу int і str.

```
class Kl8_01:                          # клас Kl8_01
    a = 4                              # змінна a набуває ціле число
    b = 5                              # змінна b набуває ціле число
    print("результат = ", a + b)       # результат = 9
    a = "4"                            # змінна a набуває значення рядка
    b = "5"                            # змінна b набуває значення рядка
    print("результат = ", a + b)       # результат = 45
    a = 7                              # змінна a набуває ціле число
    b = 8                              # змінна b набуває ціле число
    print("результат = ", a * b)       # результат = 56
    b = "8"                            # змінна b набуває значення рядка
    print("результат = ", a * b)       # результат = 8888888
```

Рис. 1. Програма з різними діями над об'єктами операції додавання та множення

Як бачимо, в останньому рядку програми об'єкт b, який має тип str, повторюється 7 разів, тобто стільки разів, які має об'єкт a типу int.

- **Методи з однаковими іменами в різних класах можуть виконувати різні дії.**

Приклад 3.
На рис. 2 зображено програму, у якій метод func класу Kl8_02a обчислює корінь квадратний із суми квадратів двох чисел, а метод із таким самим іменем класу Kl8_02b обчислює середнє значення суми трьох чисел.

```
class Kl8_02a:                          # клас Kl8_02a
    def func(self, a1, a2):             # функція func класу Kl8_02a
# метод func обчислення кореня квадратного суми квадратів двох чисел
        self.func = int(math.sqrt(a1 * a1 + a2 * a2))

class Kl8_02b:                          # клас Kl8_02b
    def func(self, a1, a2, a3):         # функція func класу Kl8_02b
# метод func обчислення середнього значення трьох чисел
        self.func = int((a1 + a2 + a3) / 3)

import math          # імпортування модуля math
ob1 = Kl8_02a()      # екземпляр класу Kl8_02a
ob2 = Kl8_02b()      # екземпляр класу Kl8_02b
ob1.func(3, 4)       # звернення до методу func об'єкта ob1
ob2.func(6, 8, 13)   # звернення до методу func об'єкта ob2
print(ob1.func)      # виведення результату викон. методу func об'єкту ob1
```

Рис. 2. Програма з однойменними методами у двох класах

У програмі створюється екземпляр ob1 класу Kl8_02a і екземпляр ob2 класу Kl8_02b.

Метод func класу Kl8_02a викликається з об'єкта ob1 і аргумент 3 передається параметру a1, а аргумент 4 — параметру a2 цього класу. Далі в цьому методі обчислюється корінь квадратний суми квадратів цих чисел.

Метод func класу Kl8_02b викликається з об'єкта ob2 і аргумент 6 передається параметру a1, аргумент 8 — параметру a2, а аргумент 13 — параметру a3. Далі в цьому методі обчислюється середнє значення суми трьох чисел.

- **За допомогою оператора if можна створити один і той самий об'єкт одного або іншого класу, у яких використовуються методи з однаковими іменами, але які виконують різні дії.**

Приклад 4.

На рис. 3 зображено програму з двома класами, у кожного з яких є метод __init__. У методі класу Kl8_03a обчислюється сума введених чисел, а в методі класу Kl8_03b — їх добуток.

```
class Kl8_03a:                          # клас Kl8_03a
    def __init__(self, a1, a2):         # конструктор класу Kl8_03a
        self.s = a1 + a2                # обчислення суми чисел

class Kl8_03b:                          # клас Kl8_03b
    def __init__(self, a1, a2):         # конструктор класу Kl8_03b
        self.s = a1 * a2                # обчислення добутку чисел

b1 = int(input("увести число b1 "))     # введення числа b1
b2 = int(input("увести число b2 "))     # введення числа b
if b1 < b2:                             # якщо b1 менше b2
    ob1 = Kl8_03b(b1, b2)               # екземпляр ob1 класу Kl8_03b
else:                                   # b1 не менше b2
    ob1 = Kl8_03a(b1, b2)               # екземпляр ob1 класу Kl8_03a
print(ob1.s)                            # виведення результату
```

Рис. 3. Програма з конструкторами в різних класах

Якщо перше уведене число (b1) менше другого уведеного числа (b2), то створюється екземпляр ob1 класу Kl8_03b, у результаті чого здійснюється звернення до методу цього самого класу та обчислюється добуток уведених чисел, інакше створюється екземпляр ob1 класу Kl8_03, а потім обчислюється й виводиться сума введених чисел. Зауважимо, що в процесі звернення до методів __init__ обох класів, аргумент b1 передається параметру a1, а аргумент b2 — параметру a2.

Отже, у наведеному прикладі поліморфізм проявляється в тому, що за допомогою оператора if створюється об'єкт ob1 класу Kl8_03b або класу Kl8_03, а один і той самий метод __init__ у двох різних класах виконує різні дії над уведеними числами.

- **Перезавантаження операторів**

Сутність перезавантаження операторів полягає в тому, що за допомогою спеціальних методів, які інколи називають магічними, одні й ті самі оператори виконують різні дії над об'єктами. Ці методи, так само, як і метод __init__, на початку й наприкінці мають два знаки підкреслення. За аналогією з методом __init__, вони викликаються автоматично, коли екземпляр об'єкта бере участь у відповідній операції, наприклад, у операції додавання, множення й ін.

У мові Python існує величезна кількість магічних методів, наведемо лише деякі з них:

Метод	Опис
__add__	Викликається автоматично, коли екземпляр класу бере участь в операції додавання (+)
__sub__	Викликається автоматично, коли екземпляр класу бере участь в операції віднімання (–)
__mul__	Викликається автоматично, коли екземпляр класу бере участь в операції множення (*)
__truediv__	Викликається автоматично, коли екземпляр класу бере участь в операції ділення (/)
__str__	Викликається, коли об'єкт перетворюється в рядок для виведення викликом убудованої функції str

Перш ніж розглянути приклад програми з перезавантаженням операторів, опишемо призначення операторів % і %s.

Оператор % (а також метод format) дає змогу вставити в рядок дані, отримані в процесі виконання програми.

Якщо необхідно вставити дані, які мають сприйматися звичайним рядком, використовується комбінація символів %s.

Сутність цих операторів пояснюється таким прикладом:

```
>>>'Наталя %s Олега' % 'подруга'
Наталя подруга Олега
```

Із прикладу видно, що після оператора % вказується, що потрібно вставити (це може бути результат, отриманий у процесі виконання програми), а комбінація символів %s — куди необхідно вставити, у цьому випадку символ s свідчить про те, що результат, який вставляється, сприймається як рядок.

Френсіс Елізабет Аллен — американська вчена, піонерка в галузі оптимізації компіляторів, стала перша жінка, яка отримала премію Тюрінга.

Приклад 5.

На рис. 4 наведено приклад програми з використанням методів __add__ і __str__ .

```
class K18_04c:                  # клас K18_04c
  def __init__(self, a1):       # конструктор класу K18_04c
      self.p1 = a1              # змінна p1 набуває значення параметра a1
  def __add__(self, a2):        # перезавантаження оператора +
      self.p1 = self.p1 + a2    # додавання або з'єднання значень
  def __str__ (self):           # перетворення об'єкта в рядок
      return '%s' % self.p1     # повернення звичайного рядка

ob1 = K18_04c(21)               # екземпляр ob1 класу K18_04c
ob1 + 15                        # до значення екземпляра ob1 додається 15
print(ob1)                      # виведення значення екземпляра ob1
ob2 = K18_04c("Пере")           # екземпляр ob2 класу K18_04c
ob2 + "завантаження"            # до значення екз. ob2 приєднується завантаження
print(ob2)                      # виведення значення екземпляра ob2
ob3 = K18_04c([19, 7, 12])      # екземпляр ob3 класу K18_04c
ob3 + [5, 9]                    # до значення екземпляра ob3 приєднується [5, 9]
print(ob3)                      # виведення значення екземпляра ob3
```

Рис. 4. Код програми з методами __add__ і __str__

Опишемо порядок виконання програми.

1. Після запуску програми створюється екземпляр класу ob1. Викликається конструктор, і параметр a1 набуває значення 21; це значення присвоюється змінній p1 об'єкта ob1.

Далі виконується інструкція ob1+15. Операція додавання (+) перехоплюється методом __add__ і за допомогою наступної операції виконується звичайна операція додавання двох чисел (числа 21 до числа 15).

2. Далі управління передається інструкції print (ob1), здійснюється перехоплення методом __str__, число 36 перетворюється у звичайний рядок і за допомогою інструкції print (ob1) виводиться на екран.

3. Створюється екземпляр класу ob2. Виконуються дії у тому самому порядку, як описано в пункті 1, але для значень 'Пере' і 'завантаження'.

4. Створюється екземпляр класу ob3. Виконуються такі самі дії над списками [19, 7, 12] і [5, 9].

Результат виконання програми наведено на рис. 5.

```
36
Перезавантаження
[19, 7, 12, 5, 9]
```

Рис. 5. Виконання програми з методами __add__ і __str__

Перезавантаження операторів є потужним засобом програмування, але ним не слід зловживати. Якщо є можливість обійтися без перезавантаження операторів, то краще його не використовувати.

Запитання для перевірки знань

1. Як проявляється поліморфізм у мові Python?
2. Від чого залежить сутність виконання оператора множення?
3. Чи можуть в одній програмі використовуватися методи з однаковими іменами?
4. Як за допомогою оператора if можуть створюватися об'єкти з одним іменем?
5. Яке призначення має метод __str__?
6. Поясніть сутність перезавантаження методів.
7. Які дії виконують оператори % і %s?

Завдання для самостійного виконання

1. Розробіть програму, у якій виконується оператор (+) над даними: 21.5, 4, 37 ; «Київ», «–21», а також оператор (*) над даними: 48, 5; «7», «Україна».
2. Розробіть програму з використанням методу __add__ додавання чисел 143.5 і 32.4, а також об'єднання рядків «ай» і «фон».
3. Дано масив чисел a[1], a[2], a[3], a[4], a[5]. Якщо перше число більше останнього, то в першому класі обчислюється добуток чисел масиву, інакше — в другому класі — їх сума.
4. Розробіть програму з двома класами, у першому з яких за допомогою методу func1 обчислюється сума чисел одновимірного масиву, а в другому класі за допомогою методу func1 — їх добуток.
5. Формується масив чотирма випадковими цілими числами в діапазоні від 7 до 14. Якщо друге число більше третього, то обчислюється сума всіх чисел масиву, інакше — в другому класі — їх добуток.

7.2. Перевизначення та розширення можливостей методів

Значну кількість об'єктів побудовано за ієрархічною структурою. В ООП методи теж можуть мати ієрархічну структуру. Для звернення до конкретного методу в цій структурі необхідно мати відповідні правила. Як, на вашу думку, можна організувати звернення до методу в такій ієрархії?

Поліморфізм і наслідування класів дають змогу реалізувати перевизначення методів.

Сутність перевизначення методів полягає в тому, що в ієрархічній структурі класів, якщо підклас не містить методу, до якого виконується звернення із певного об'єкта, то здійснюється пошук такого методу в найближчому зверху батьківському класі. Якщо й у ньому відсутній такий метод, то він відшукується також у найближчому до нього зверху батьківському класі.

Принцип перевизначення методів пояснюється схемою (рис. 1). Як бачимо, в процесі звернення до методу_3 з підкласу_2 цей метод виконується одразу.

Аналогічно виконується й метод_2 у процесі звернення до нього з підкласу_1.

У процесі звернення до методу_1 із підкласу_1 цей метод відшукується в самому підкласі, а потім виконується звернення до суперкласу, де й виконується цей метод.

Нарешті, у процесі звернення до методу_4 з підкласу_2 метод відшукується спочатку в цьому підкласі, потім у підкласі_1 і виконується в суперкласі.

Метод виконується в тому класі, у першому з яких буде знайдено цей метод. Якщо в жодному батьківському класі такого методу не буде знайдено, то буде видано повідомлення про синтаксичну помилку.

Пошук методу виконується за принципом «знизу догори».

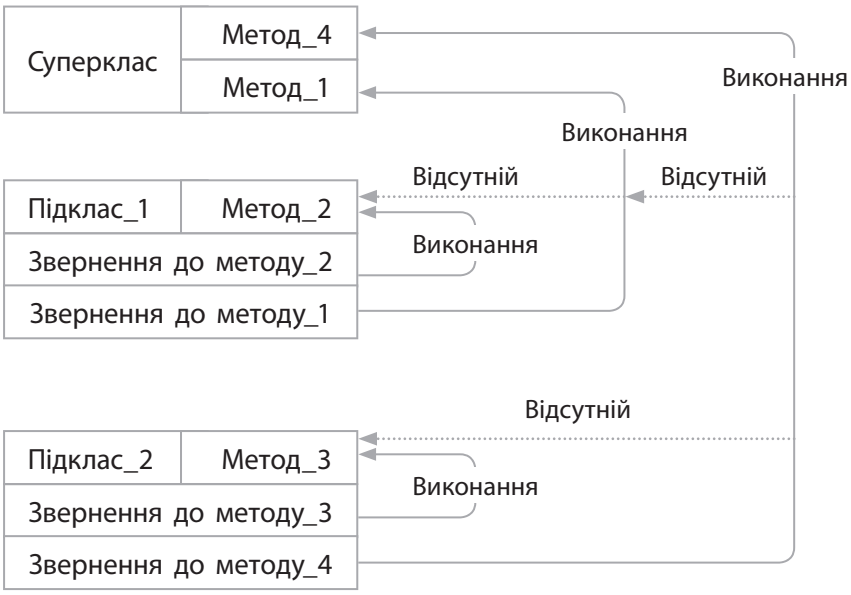


Рис. 1. Варіант перевизначення методу

Приклад 1.

На рис. 2 зображено програму з перевизначенням методу. У програмі є один суперклас і два підкласи. У суперкласі використовуються методи __init__ і func1, у підкласі Kl8_04a один метод func2 і в підкласі Kl8_04b методи func3 і func1.

```
class Kl8_04:                          # супеклас Kl8_04
    def __init__(self, a1):            # конструктор суперкласу Kl8_04
        self.p = a1                    # змінна p набуває значення параметру ф1
    def func1(self):                   # метод func1 суперкласу Kl8_04
        print("у класі Kl8_04 =", self.p - 13 )    # виведення значення змінної p

class Kl8_04a(Kl8_04):                 # підклас Kl8_04a суперкласу Kl8_04
    def func2(self, a2):               # метод func2 підкласу Kl8_04a
        self.p *= a2                   # еквівалентно self.p=self.p*a2
        print("у підкласі Kl8_04a =", self.p)      # виведення значення змінної p

class Kl8_04b(Kl8_04):                 # підклас Kl8_04b суперкласу Kl8_04
    def func3(self):                   # метод func3 підкласу Kl8_04b
        self.func3 = list(str(self.p))             # перетворення рядку у список
        print("список: =", self.func3)             # виведення списку рядку 'байт'
    def func1(self):                   # метод func1 підкласу Kl8_04b
        i = 0                          # початкове значення змінної
        while i < len(self.func3):                 # доти, доки і менше довжини списку
            print(i, "елемент: ", self.func3[i])   # виведення букв у стовпець
            i += 1                                 # еквівалентно i = i+1

ob1 = Kl8_04a(37)              # екземпляр ob1 підкласу Kl8_04a
ob2 = Kl8_04b('байт')          # екземпляр ob2 підкласу Kl8_04b
ob1.func2(3)                   # звернення до методу func2 із об'єкта ob1
ob1.func1()                    # звернення до методу func1 із об'єкта ob1
ob2.func3()                    # звернення до методу func3 із об'єкта ob2
ob2.func1()                    # звернення до методу func1 із об'єкта ob2
```

Рис. 2. Програма з перевизначенням методу

Опишемо порядок виконання програми.

1. У процесі створення об'єкта ob1 класу Kl8_04a автоматично викликається метод __init__ і аргумент 37 передається параметру a1 цього методу. За допомогою наступної інструкції self.p = a1 змінна p об'єкта ob1 набуває значення 37. А в процесі створення об'єкта ob2 підкласу Kl8_04b змінна p об'єкта ob2 набуває значення рядка 'байт'.

2. Далі здійснюється звернення до методу func2 з об'єкта ob1 підкласу Kl8_04a. Оскільки цей метод є в цьому підкласі, він одразу виконується. У результаті чого параметру a2 передається значення аргумента, тобто число 3. Наступною командою в цьому підкласі значення змінної p об'єкта ob1 (це значення дорівнює 37) множиться на 3 і на екран виводиться результат 111.

3. Після цього здійснюється звернення до методу func1 з об'єкта ob1 підкласу Kl8_04a. Але такого методу в цьому підкласі немає. Тому здійснюється пошук такого методу за принципом «знизу догори». Такий метод є в суперкласі Kl8_04, який є батьківським підкласу Kl8_08a. Цей метод і буде виконуватися. У результаті від значення p (тобто від 111) віднімається число 13 і виводиться на екран. Отже, виконалося перевизначення методу func1.

4. Далі з об'єкта ob2 підкласу Kl8_04b викликається метод func3. Він є в цьому підкласі, тому одразу й виконується. У результаті змінна p об'єкта ob2, яка має значення рядка 'байт', перетворюється на список і виводиться на екран.

5. Останньою командою програми викликається метод func1 з об'єкта ob2 підкласу Kl8_04b. Цей метод також є в підкласі Kl8_04b і він також одразу виконується. У результаті в стовпець виводяться елементи списку ['б','а','й','т'].

Результат виконання програми наведено на рис. 3. Таким чином, у програмі здійснилося перевизначення методу func1.

```
у підкласі Kl8_04a = 111
у класі Kl8_04 = 98
список: = ['б', 'а', 'й', 'т']
0 елемент: б
1 елемент: а
2 елемент: й
3 елемент: т
```

Рис. 3. Виконання програми з перевизначенням методу func1

Розглянемо механізм розширення можливостей методів. До цього вдаються, коли виникає потреба звернутися від методу одного класу до методу іншого. Наприклад, із методу підкласу потрібно звернутися до методу суперкласу для розширення можливостей підкласу.

Приклад 2.

У першому риболовецькому господарстві виловили 10 т риби, у другому — 7 т. До кожного з них може надійти замовлення на постачання риби від 1 до 6 магазинів. Риба між замовниками розподіляється порівну. Визначте, скільки тонн риби дістанеться кожному замовнику для таких варіантів: до першого господарства може надійти замовлення від 1 або 2 магазинів; до другого господарства — від 3 або 4 магазинів; до першого господарства — від 5 або 6.

Можна виконати операцію ділення в такому порядку: поділити число 10 на 1 і на 2, потім число 7 на 3 і на 4, потім число 10 на 5 і на 6. Програму реалізації завдання подано на рис. 4.

```
class Kl8_06:                          # суперклас Kl8_06
  def __init__(self, a1, a2):          # конструктор суперкласу Kl8_06
# змінні p1 і p2 об'єкту ob1 набувають значень параметрів a1 і a2
      self.p1 = a1
      self.p2 = a2
  def func1(self):                     # метод func1 суперкласу Kl8_06
      self.p1 =self.p1 / self.p2       # ділення значення p1 на p2
      print(self.p1)                   # виведення значення змінної p1

class Kl8 06a(Kl8 06):                 # підклас Kl8 06a суперкласу Kl8 06
  def func1(self):                     # метод func1 підкласу Kl8_06a
      print("перехід 1")               # виведення повідомлення
      Kl8_06.func1(self)               # звернення до методу func1 суперкласу
      print("перехід 2")               # виведення повідомлення

i = 1                                  # початкове значення змінної
while i < 7:                           # доти, доки і менше 7
  if  2 < i and i < 5:                 # умова
      ob1 = Kl8_06a(7 , i)             # об'єкт ob1 підкласу Kl8_06a
  else:                                # інакше
      ob1 = Kl8_06(10, i)              # об'єкт ob1 суперкласу Kl8_06
  i += 1                               # еквівалентно i = i + 1
  ob1.func1()                          # звернення до методу func1 із об'єкта ob1
```

Рис. 4. Код програми з розширенням можливостей методу

Опишемо порядок виконання програми. Зазначимо, що, запустившись, програма починає виконуватися з інструкції i=1.

Приклад 2 (продовження).

I цикл. Оскільки результатом оператора if є значення False, здійснюється перехід на гілку else. У результаті створюється екземпляр класу ob1 суперкласу Kl8_06. Автоматично викликається метод __init__ , параметр a1 набуває значення 10, а параметр a2 — значення 1. Значення цих параметрів за допомогою наступних двох інструкцій цього методу присвоюються, відповідно, змінним p1 і p2.

Далі управління передається інструкції i += 1 основної програми. Змінна i набуває значення 2, а за допомогою інструкції ob1.func1() викликається з об'єкта ob1 метод func1 суперкласу і значення змінної p1 ділиться на значення змінної p2 (число 10 ділиться на 1) і результат (10) виводиться на екран. Після цього управління передається оператору циклу.

II цикл. Починається з перевірки умови (2<i) and (i<5), яке також має значення False. Так само реалізується гілка else і так само, як і в попередньому випадку, викликається метод __init__, у результаті чого параметр a1 набуде значення 10, а параметр a2 — значення 2. Відповідно, змінні p1 і p2 набудуть значень 10 і 2. Знову виконується інструкція i += 1 і змінна i набуде значення 3, потім здійснюється звернення до методу func1 суперкласу, у результаті чого буде виведено число 5 (у результаті ділення: 10/2=5). Як і в попередньому випадку, далі управління передається оператору циклу.

III цикл. Нагадаємо, що на цей момент змінна i має значення 3, тому результат оператора if має значення True і створюється екземпляр ob1 підкласу Kl8_06a. Так само автоматично викликається метод __init__, у результаті чого змінна p1 набуде значення 7, а змінна p2 — значення 3. Потім виконується інструкція i =+ 1 і змінна i набуде значення 4.

Далі викликається метод func1, але цього разу з підкласу. Тому виводиться повідомлення 'перехід 1' і за допомогою інструкції Kl8_06.func1 (self) цього методу викликається метод func1 із суперкласу, де значення змінної p1 (воно дорівнює 7) ділиться на значення змінної p2 (тобто на 3) і результат (2.33) виводиться на екран. Управління передається інструкції print ("перехід 2") і буде виведено перехід 2. Після цього управління знову передається оператору циклу.

IV цикл. Змінна i на цей момент має значення 4, тому виробляється значення True і створюється екземпляр ob1 підкласу Kl8_06a. Викликається метод __init__, у результаті чого змінна p1 набуде значення 7, а змінна p2 — значення 4. Потім виконується інструкція i += 1, змінна i набуває значення 5 і викликається метод func1 підкласу. У результаті друкується перехід 1 і за допомогою наступної команди викликається метод func1 суперкласу, у якому число 7 ділиться на число 4 і результат 1.75 виводиться на екран. Потім управління передається інструкції print ("перехід 2") і перехід 2 виводиться на екран. Далі управління знову передається оператору циклу.

V цикл. На цей момент змінна i має значення 5. У результаті перевірки умови if 2<i<5, як і в перших двох циклах, виробляється значення False і створюється екземпляр ob1 об'єкта суперкласу. Викликається метод __init__, у результаті чого змінна p1 буде мати значення 10, а змінна p2 — значення 5. Далі виконується оператор i += 1, змінна i набуває значення 6, викликається метод func1 суперкласу і значення змінної p1 ділиться на p2. Результат 2.0 виводиться на екран. Далі управління передається оператору циклу.

VI цикл. Змінна i має значення 6. Цикл виконується аналогічно до попереднього. Буде виведено значення 1.66. Під час виконання циклу змінна i матиме значення 7, тому оператор while завершує своє виконання.

Результат виконання програми подано на рис. 5.

```
10.0
5.0
перехід 1
2.3333333333333335
перехід 2
перехід 1
1.75
перехід 2
2.0
1.6666666666666667
```

Рис. 5. Виконання програми з розширенням можливостей методу

Запитання для перевірки знань

1. Поясніть, у чому полягає сутність перевизначення методів.
2. У чому полягає сутність розширення можливостей методу?
3. Накресліть схему класів із прикладом перевизначення методів.
4. Наведіть приклад реалізації розширення можливостей методу.

Завдання для самостійного виконання

1. Дано числа 16 і 7. Розробіть програму визначення їх суми, добутку, різниці та ділення першого на друге з використанням перевизначення методу. У програмі передбачте створення суперкласу і двох його підкласів, а також чотирьох методів у різних класах.
2. Дано два рядки «екземпляр» і «підкласу». Розробіть програму об'єднання цих рядків, перетворення першого рядка в список і вилучення з другого рядка останньої букви. У програмі передбачте перевизначення методу з використанням суперкласу і його підкласу.
3. У суперкласі та його підкласі використовуються два методи з однаковими іменами. У суперкласі за допомогою методу обчислюється середнє значення суми трьох чисел, а в підкласі здійснюється звернення до методу суперкласу. Розробіть програму реалізації цієї ситуації.
4. Генеруються два цілі випадкові числа в діапазоні від 7 до 15. Якщо їх сума більше 17, числа складаються, інакше множаться. Розробіть програму реалізації завдання з використання суперкласу та двох підкласів.
5. Модифікуйте програму так, щоб у ній використовувався її суперклас і підклас Kl8_06b, зберігся принцип перевизначення методу і програма повинна видавати той самий результат, що й початкова програма.

7.3. Композиційний підхід в ООП мовою Python

Раніше нам уже багаторазово доводилося створювати програмний код з окремих функцій, методів і модулів. Ще один варіант створення цілісного програмного коду реалізується на основі композиційного підходу.

Сутність композиційного підходу в програмуванні мовою Python полягає в тому, що в класі, який називається *класом-контейнером*, створюються інструкції виклику інших класів. Такий підхід дає змогу на основі створених об'єктів класу-контейнера отримати і об'єкти класів, що з нього викликаються.

Композиція в програмуванні — це вбудовування інших об'єктів у об'єкт-контейнер і використання їх для реалізації методів класу-контейнера.

Композицію інколи ще називають **агрегуванням**.

На практиці композиція забезпечує формування цілого фрагмента коду з окремих частин.

Розглянемо сутність композиції на прикладі.

Приклад.

На аркуші паперу з відомими розмірами накреслено два кола радіусом r1, два кола радіусом r2 та один рівносторонній трикутник зі стороною a. Усі ці фігури не перетинаються одна з одною. Необхідно обчислити загальну площу аркуша паперу й залишок площі аркуша. Зрозуміло, що для обчислення залишку площі аркуша потрібно обчислити площу кожного кола та площу трикутника. Обчислення цих площ фактично і є тими частинами, на основі яких можна обчислити залишок площі.

Програму для розв'язування цієї задачі можна розробити різними способами. Можна обійтися і без композиційного підходу, але ми розробимо програму саме на його основі.

У нашій задачі фактично є три об'єкти: об'єкт-аркуш, об'єкт-коло і об'єкт-трикутник. Відповідно, в одному класі (класі-контейнері) будемо оперувати з площею аркуша, у другому — із площею кола й у третьому — із площею трикутника.

Площу кожного кола обчислимо в класі Kl8_07a, а площу трикутника — у класі Kl8_07b.

Звернення до класу Kl8_07a для обчислення площі першого кола виконаємо з об'єкта p1 класу Kl8_07a, який є об'єктом класу Kl7_07 (має позначення self), а звернення до цього самого класу для обчислення площі другого кола — з об'єкта p2.

Звернення до класу Kl8_07b для обчислення площі трикутника виконаємо з об'єкта p3 класу Kl8_07b, який також є об'єктом класу Kl8_07 і також має позначення self.

Обчислення залишку площі виконаємо за допомогою методу func2, а виведення результатів — за допомогою методу func3.

З урахуванням викладеного програму реалізації завдання зображено на рис 1.

Звернемо увагу на те, що в методі func1 створюються три об'єкти: p1, p2 і p3, а також атрибути k1 і k2, у яких міститься кількість кіл, відповідно, з радіусами r1 і r2.

Також звернемо увагу на те, як виконується в методі func2 звернення до обчислених площ кіл і трикутника: указується об'єкт класу Kl8_07, який у класі його замінює self, далі — об'єкт класу Kl8_07a або класу Kl8_07b (мають позначення p1, p2, p3) і потім сам атрибут (plcolo або pltrik).

```
class Kl8_07a:                          # клас Kl8_07a
    def __init__(self, x):              # конструктор класу Kl8_07a
        self.plcolo = 3.14 * x * x      # обчислення площі кола
class Kl8_07b:                          # клас Kl8_07b
    def __init__(self, x):              # конструктор класу Kl8_07b
        self.pltrik = math.sqrt(3) * x * x / 4     # площа трикутника
class Kl8_07:                           # клас Kl8_07
    def __init__(self, x, y):           # конструктор класу Kl8_07
        self.plzag = x * y              # загальна площа аркуша
    def func1(self, r1, r2, a, m=2, n=2):          # метод func1
        self.p1 = Kl8_07a(r1)           # звернення до класу Kl8_07a
        self.p2 = Kl8_07a(r2)           # звернення до класу Kl8_07a
        self.p3 = Kl8_07b(a)            # звернення до класу Kl8_07b
        self.k1 = m                     # кількість перших кіл
        self.k2 = n                     # кількість других кіл
    def func2(self):                    # метод func2
        # обчислення залишку площі аркуша
        self.func2 = self.plzag - self.p1.plcolo * self.k1\
                     -self.p2.plcolo * self.k2 - self.p3.pltrik
    def func3(self):                    # метод func3
        # виведення загальної площі аркуша
        print("Загальна площа = ", str(self.plzag), "см. кв.")
        # виведення залишку площі аркуша
        print("Залишок площі = ", str(self.func2), "см. кв.")
import math                             # імпортування модуля math
ob = Kl8_07(30, 20)                     # об'єкт класу Kl8_07
ob.func1(4, 6, 5)                       # звернення до методу func1
ob.func2()                              # звернення до методу func2
ob.func3()                              # звернення до методу func3
```

Рис. 1. Програма, побудована на основі композиційного підходу

Опишемо порядок виконання програми.

1. Після запуску програми виконується інструкція ob = Kl8_07 (30, 20), створюється об'єкт ob класу Kl8_07 і викликається його конструктор. У результаті аргумент 30 передається параметру x, а аргумент 20 — параметру y. Обчислюється загальна площа аркуша, яка зберігається в змінній plzag цього об'єкта (self.plzag).

2. Управління передається інструкції ob.func1 (4, 6, 5), викликається метод func1 класу Kl8_07 і значення 4 передається параметру r1, значення 6 — параметру r2 і значення 5 — параметру a.

3. Далі створюється об'єкт p1 класу Kl8_07a (який, у свою чергу, є об'єктом класу Kl8_07), автоматично викликається конструктор цього класу, обчислюється площа кола з радіусом r1, яка зберігається в змінній plcolo об'єкта self.p1. За аналогією виконується інструкція self.p2 = Kl8_07b(r2), обчислюється площа кола з радіусом r2, яка зберігається в змінній plcolo об'єкта self.p2.

4. Виконується інструкція self.p3 = Kl8_07b (a). Викликається конструктор класу Kl8_07b, обчислюється площа трикутника, яка зберігається в змінній pltrik об'єкта self.p3. Далі в змінних self.k1 і self.k2 запам'ятовуються кількість кіл, відповідно, першого і другого типу.

5. Управління передається інструкції ob.func2(). Здійснюється звернення до функції func2, де обчислюється залишок площі аркуша.

6. Управління передається інструкції ob.func3. Викликається метод func3 і на екран виводиться загальна площа аркуша й залишок площі.

Результат виконання програми подано на рис. 2.

```
Загальна площа =  600 см. кв.
Залишок площі =  262.6146824526945 см. кв.
```

Рис. 2. Виконання програми, побудованої на основі композиційного підходу

Запитання для перевірки знань

1. У чому полягає сутність композиційного підходу в програмуванні?
2. Який клас називають класом-контейнером?
3. Із якою метою застосовується композиційний підхід?
4. Поясніть сутність композиційного підходу на конкретному прикладі.

Завдання для самостійного виконання

1. На письмовому столі розміром l*p лежить папір формату A4 і книжка формату A5. Розробіть програму на основі композиційного підходу обчислення площі, яку займають книжка й папір, і вільну площу поверхні столу.
2. У порожній басейн розміром l, p, h одночасно одна помпа починає закачувати воду в басейн продуктивністю v1 $м^3$ на годину, а друга викачувати воду з басейну продуктивністю v2 $м^3$ на годину (v1>v2). Розробіть програму на основі композиційного підходу визначення об'єму води в басейні через 3 год і об'єм води, який ще може бути закачаний у басейн.
3. У залізничний вагон потрібно завантажити P т зернових. Із однієї фірми доставлено p1 т зерна, із другої — p2 т і з третьої — p3 т. Розробіть програму на основі композиційного підходу визначення загальної кількості тонн доставленого зерна й кількості тонн, які ще потрібно завантажити, або кількості тонн залишку зерна.
4. Дано два масиви цілих чисел. Розробіть програму на основі композиційного підходу визначення середнього значення кожного масиву. На скільки кожне з них відрізняється від заданого числа P?
5. Формуються два масиви чотирма 4 випадковими цілими числами в діапазоні 1–16. Розробіть програму на основі композиційного підходу визначення різниці суми чисел цих масивів.

7.4. Створення та використання модулів користувача

Бібліотека мови Python містить велику кількість модулів, окремі з яких ми імпортували у власні програми. Чому, на вашу думку, виникає необхідність розроблення модулів користувача?

Раніше вже неодноразово зазначалося, що мова Python реалізує модульний принцип організації програм, який передбачає імпортування в основний файл коду вмісту інших файлів. У розглянутих програмах уже не раз здійснювалося імпортування стандартних модулів у програму, що розроблялася. Але імпортувати можна не лише стандартні модулі, а й власні модулі, а також модулі, розроблені іншими програмістами. Відзначимо, що під час трансляції в машинний код перетворюються основна програма, а також файли, що імпортуються.

Як відомо, імпортування одного файла в інший реалізується за допомогою інструкцій import і from. Інструкцію import доцільно застосовувати, коли передбачається використовувати весь вміст модуля, а інструкцію from — коли передбачається використовувати двітри функції із цього модуля.

Якщо використовується інструкція import, то для доступу до змінних (атрибутів) модуля з основної програми слід зазначати ім'я цього модуля, потім через крапку ім'я атрибута (<ім'я модуля>.<атрибут>). Таку дисципліну доступу називають **крапковою нотацією**.

Інструкція from не потребує вказувати ім'я модуля. Далі використовуватимемо переважно інструкцію import.

Поняття модуля та файла в мові Python майже ідентичні. Файли в мові Python мають зазвичай розширення .py. Та для імпортування файла розширення не вказується.

Інструкція імпортування модуля має таку структуру:

import <ім'я модуля>

За замовчуванням інтерпретатор Python здійснює пошук модуля в каталозі, де міститься сам Python, або в тому самому каталозі, де й файл, у який виконується імпортування.

Програмісти вважають «гарним тоном» додавати до своїх програм текст із поясненням сутності програми й розміщувати на початку модуля або після заголовка класу, методу.

Приклад 1.

Нехай для обчислення й виведення площі прямокутного трикутника нами або іншим програмістом розроблено модуль з іменем trukytn, вміст якого зображено на рис. 1.

```
class Plosha:                                  # клас Plosha
    def __init__(self, a1, a2, a3, a4):        # конструктор класу
        self.p1 = a1                           # змінна p1 набуває значення параметра a1
        self.p2 = a2                           # змінна p2 набуває значення параметра a2
        self.p3 = a3                           # змінна p3 набуває значення параметра a3
        self.p4 = a4                           # змінна a4 набуває значення параметра a4
        self.func1()                           # звернення до методу func1
    def func1(self):                           # метод func1
        s =(self.p2 * self.p3) / 2             # обчислення площі прямокутного трикутника
        print(self.p1, self.p4, "=", s)        # виведення значення площі
```

Рис. 1. Модуль trukytn для обчислення площі прямокутного трикутника

Скористаємося цим модулем для обчислення площ багатьох прямокутних трикутників із різними катетами. Зрозуміло, для цього необхідно розробити програму, яка є основною програмою, до якої імпортується цей модуль і реалізується уведення значень катетів трикутників. Вміст такої програми зображено на рис. 2.

```
import trukytn                          # імпортування модуля trukytn
for i in range(3):                      # циклічне уведення сторін трикутника
    a = int(input("увести a: "))        # уведення значення сторони a
    b = int(input("увести b: "))        # уведення значення сторони b
# звернення до класу Plosha модуля trukytn
    ob = trukytn.Plosha("площа трикутника №", a, b, i)
```

Рис. 2. Основна програма обчислення площ прямокутних трикутників

Як бачимо, у цій програмі циклічно вводяться катети трьох трикутників. Після введення значення кожного катета необхідно натиснути клавішу Enter.

Зауважимо на тому, як здійснюється звернення до класу Plosha модуля trukytn. У тілі циклу створюється об'єкт ob класу Plosha модуля trukytn. У процесі створення об'єкта автоматично викликається конструктор класу Plosha й уведені значення катетів, значення параметра циклу й рядок «площа трикутника N» передаються відповідним параметрам методу __init__.

Далі в цьому класі обчислюється й виводиться площа першого трикутника. Потім аналогічно виконується другий і третій цикли. На рис. 3 наведено один із можливих варіантів виконання програми.

```
увести a: 4
увести b: 5
площа трикутника №0 = 10.0
увести a: 6
увести b: 9
площа трикутника №1 = 27.0
увести a: 7
увести b: 6
площа трикутника №2 = 21.0
```

Рис. 3. Виконання програми обчислення площ прямокутних трикутників

В основну програму можна імпортувати не лише модулі, що містять класи, а й модулі, які містять один або більше методів.

Приклад 2.

Нехай, наприклад, існує модуль kilkist (рис. 4), за допомогою якого з клавіатури вводяться цілі числа, із яких формується одновимірний масив з іменем mas. У цьому масиві підраховується й виводиться на екран кількість чисел, які дорівнюють 5.

```
def func1():                                  # метод func1 модуля kilkist
  k = 0                                       # початкове значення кількості цифр
  mas = []                                    # порожній масив
  for i in rangr(4):                          # організація чотирьох циклів
      b = int(input("Увести елемент"))        # уведення елемента
      mas.append(b)                           # формування масиву
      if mas[i] == 5:                         # чи рівний елемент?
          k += 1                              # збільшення кількості цифр
  print("Таких чисел =", k)                   # виведення результату
```

Рис. 4. Модуль kilkist формування одновимірного масиву

Щоб скористатися цим модулем, слід скласти основну програму, яка містить лише дві інструкції, зображені на рис. 5.

```
import kilkist      # імпортування модуля
kilkist.func1()     # звернення до методу func1
```

Рис. 5. Програма імпортування модуля kilkist

Безумовно, модуль, що імпортується в основну програму, можна адаптувати до власних потреб. Наприклад, у наведеному модулі kilkist можна змінити довжину масиву, значення числа, яке підраховується. У результаті запуску основної програми можемо отримати, наприклад, такий варіант результату (рис. 6.).

```
Увести елемент 5
Увести елемент 74
Увести елемент 5
Увести елемент 33
Таких чисел = 2
```

Рис. 6. Виконання програми імпортування модуля kilkist

Із прикладами три-п'ять ви можете ознайомитися на сайті interactive.ranok.com.ua

Отже, програміст може конструювати свою програму з програм, розроблених іншими програмістами, а також зі стандартних модулів і бібліотек мови Python. Але для того щоб використовувати модулі інших програмістів, необхідно знати їх призначення та можливості. Для цього застосовують засоби документування.

Документувати програмний код можна різними способами. Найпростіший із них — це *звичайні коментарі*, які використовувалися в наведених раніше програмах. Інший спосіб — *створення спеціальних рядків документації*, які є текстом у потрійних лапках або в потрійних апострофах.

Запитання для перевірки знань

1. За допомогою яких інструкцій імпортуються модулі в програму?
2. Яка різниця між інструкціями import і from?
3. Яку структуру має інструкція import?
4. У якому каталозі здійснюється пошук модуля за замовчуванням?
5. Як здійснюється імпортування модулів, які містять лише методи?
6. Із якою метою використовуються засоби документування?

Завдання для самостійного виконання

1. Розробіть модуль обчислення об'єму кулі. Використайте цей модуль для обчислення трьох куль із різними радіусами.
2. Розробіть модуль обчислення середнього значення заданого масиву чисел 33, 8, 22, 17. Використайте цей модуль для обчислення середнього значення двох масивів із різними значеннями елементів.
3. Розробіть модуль обчислення об'єму конуса з відомим радіусом основи й висоти. Використайте цей модуль для обчислення об'єму двох різних конусів.
4. Розробіть модуль створення масиву з п'ятьма випадковими числами в діапазоні від 3 до 10. Використайте цей модуль для створення двох масивів із різними діапазонами чисел.

7.5. Опрацювання виняткових ситуацій

Чи доводилося вам стикатися з винятковими ситуаціями в програмуванні раніше? Наведіть приклади виняткових ситуацій.

Виняток — це подія, яка може виникнути під час виконання програми й яка може змінити подальший хід її виконання.

Винятки генеруються автоматично, коли, наприклад, у програмному коді виникає помилка (ділення на нуль, вихід індексу масиву за межі допустимого значення й ін.). У таких випадках програма припиняє своє виконання й на екрані висвітлюється діагностичне повідомлення: тип винятку, перелік рядків і функцій, які були активними в момент появи винятку.

Приклад 1.

Після спроби виконати інструкцію y = 23 / 0 отримаємо (рис. 1):

```
>>> y = 23 / 0
Traceback (most recent call last):
File "<pyshell#0>", line 1, in <module>
y = 23 / 0
ZeroDivisionError: division by zero
```

Рис. 1. Спроба виконати інструкцію y = 23 / 0

Тут ZeroDivisionError — це стандартний виняток (убудований у мову) помилки ділення на нуль, який входить до класу арифметичних помилок (до класу ArithmeticError).

У мові Python існує понад 50 класів стандартних винятків. Так, виняток ValueError може з'явитися, коли в методі index() фрагмент, що виконується, не входить у рядок:
>>>"принтер".index ("миша")

Але в мові Python існують засоби перехоплення винятків. Після їх опрацювання виконання програми може продовжитися до її завершення. Для цього призначена інструкція try.

Інструкція try має таку загальну структуру:

try :
    <блок, у якому перехоплюється виняток>
[except [<виняток 1> [as <об'єкт винятку>]]:
    <блок, який виконується в разі виникнення винятку 1>
    …
[except [<виняток N> [as <об'єкт>]]:
    <блок, який виконується в разі виникнення винятку N>]]
[else:
    <блок, який виконується, якщо виняток не виник>]
[finally:
    <блок, який виконується в будь-якому випадку>]

Інструкції, винятки яких необхідно перехоплювати, мають розміщуватися всередині блока try. У параметрі <виняток> блока except вказується клас винятку, що опрацьовується, наприклад, клас ZeroDivisionError.

Інструкції блока try виконуються в такому порядку.

Крок 1	Якщо в інструкціях блока try виникає виняток, управління передається найближчому блоку except.
Крок 2	Якщо виняток не відповідає класу цього блока, управління передається наступному блоку except.
Крок 3	Якщо жоден блок except не відповідає цьому винятку, то цей виняток передається для опрацювання на наступний, більш високий рівень.
Крок 4	Якщо в програмі взагалі не передбачено опрацювання цього винятку, то видається повідомлення про помилку і програма припиняє своє виконання.

Приклад 2.

На рис. 2 зображено найпростішу програму, в якій уводяться два цілих числа, і перше ділиться на друге. Операція ділення чисел міститься в блоці try, де виконується перехоплення та опрацювання винятку ділення на нуль.

```
class K18_16:                                  # клас K18_16
    def __init__(self, a1, a2):                # конструктор класу K18_16
        try:                                   # блок try
            self.z = a1 / a2                   # ділення уведених чисел
        except ZeroDivisionError:              # перехоплення ділення на 0
            print("ділення на 0")              # виведення повідомлення
        else:                                  # відсутнє ділення на 0
            print("результат =", self.z)       # виведення результату
        finally:                               # блок виконується обов'язково
            print("програма завершена")        # виведення повідомлення

x = int(input("увести число x: "))             # уведення числа x
y = int(input("увести число y: "))             # уведення числа y
```

Рис. 2. Програма з перехопленням винятку ділення на нуль

Як бачимо, у програмі уведене число x ділиться на число y. Якщо число y не дорівнює нулю, виконується інструкція print ("результат =", self.z), результат ділення виводиться на екран і після цього виводиться повідомлення програма завершена. На рис. 3 наведено один із можливих варіантів виконання.

```
увести число x: 31
увести число y: 4
результат = 7.75
програма завершена
```

Рис. 3. Варіант виконання програми

Якщо уведене число y дорівнює нулю, то ділення на нуль перехоплюється класом винятку ZeroDivisionError, виводиться повідомлення ділення на 0. Після цього виконання програми триває й видається повідомлення програма завершена.

Зміст варіанта виведення на екран у цьому разі може бути таким, як на рис. 4.

```
увести число x: 23
увести число y: 0
ділення на 0
програма завершена
```

Рис. 4. Варіант виконання програми

Одна інструкція except може опрацьовувати кілька винятків. Для цього їх слід перелічити через кому, наприклад:
Except (IndexError, ZeroDivisionError)

Отже, якщо в програмі передбачено опрацювання відповідних винятків, то вона продовжує виконуватися до повного завершення й видається повідомлення про помилку.

Приклад 3.

Для кращого розуміння сутності опрацювання винятків внесемо в програму (рис. 4), деякі зміни — вилучимо з неї блок try. Програма набуде такого змісту (рис. 5):

```
class K18_17:                              # клас K18_17
    def __init__(self, a1, a2):            # конструктор класу K18_17
        self.z = a1 / a2                   # ділення уведених чисел
        print("результат =", self.z)       # виведення результату

x = int(input("увести число x: "))         # уведення числа x
y = int(input("увести число y: "))         # уведення числа y
ob = K18_17(x, y)                          # екземпляр ob класу K18_17
```

Рис. 5. Програма без перехоплення винятку ділення на нуль

Після запуску програми й уведення другого числа, що дорівнює нулю, програма припинить своє виконання й на екран буде видано повідомлення про помилку (рис. 6):

```
увести число x: 7
увести число y: 0
Traceback (most resent call last):
    File "C:/Python34/prog_8.40.py", line 8, in <module>
        Ob = K18_17 (x, y)                 # екземпляр ob класу K18_16
    File "C:/Python34/prog_8.40.py", line 3, in __init__
        self.z = a1 / a2                   # ділення уведених чисел
ZeroDivisionError: division by zero
```

Рис. 6. Виконання програми без перехоплення винятку ділення на нуль

Винятки користувача створюються у вигляді класів, які наслідують один із класів убудованих винятків. Найчастіше для цього використовується клас **Exception**.

Для опрацювання винятків можна також використати ієрархічну структуру вкладеності, тобто один виняток можна вкласти в інший.

Уже розглядалися винятки, які генерувалися інтерпретатором, коли в програмі виявлялася помилка. Але винятки можуть генеруватися й самою програмою, можуть перехоплюватися або не перехоплюватися. Для перехоплення інструкція raise включається в блок try.

Якщо виняток, визначений програмою, не перехоплюється, він буде переданий опрацювачу винятків за замовчуванням. У результаті програма припинить своє виконання й на екран буде виведене стандартне повідомлення про помилку. Винятки можуть генеруватися також за допомогою інструкції assert, яка є умовною формою інструкції raise.

Пояснимо сутність генерування винятків користувачем на прикладі.

Приклад 4.

Припустимо, що на рахунок у банк покладено суму K і з банкомата ми намагаємося зняти суму x грн. Зрозуміло, що коли сума x не перевищує K, банкомат має видати таку суму й повідомити про остачу. Але ми можемо помилитися й увести більшу суму. У такому випадку банкомат не повинен видавати гроші й має повідомити про помилку. Програму, що реалізовує таку ситуацію, зображено на рис. 7.

```
class Kl8_18(Exception):                  # клас винятку Kl8_18
    def __init__(self,a1, a2):            # a1 набуває значення x, a2 - K
        Exception.__init__(self)          # конструктор класу Exception
        self.p1 = a1                      # p1 набуває значення a1
        self.p2 = a2                      # p2 набуває значеня a2

K = int(input("Сума вкладу: "))           # вводиться сума вкладу
try:                                      # блок try
    x = int(input("Яку суму зняти? "))    # вводиться сума для зняття
    if x > K:                             # сума перевищує суму вкладу?
        raise Kl8_18(x, K)                # звернення до класу Kl8_18
except Kl8_18 as t1:                      # перехоплення винятку
                                          # виведення значення p1
                                            класу Kl8_18 з форматуванням

    print('Зняти суму {0} не можна\
          '.format(t1.p1))
    print("На рахунку залишається ", K) # виведення суми вкладу
else:                                     # виконується, якщо сума доступна
    s = K - x                             # обчислюється залишок вкладу
    print("Знято ", x, "Залишок ", s)     # повідомлення про викону операцію
finally:                                  # виконується обов'язково
    print("Програма завершена")           # повідомлення про завершення
```

Рис. 7. Код програми з генеруванням винятку в програмі

У програмі створено власний тип винятку, який має назву Kl8_18. Він містить два поля: x і K, у яких зберігається сума, яку бажано зняти, і сума вкладу. В інструкції except указано клас помилки, котрий буде збережений як змінна t1, у якій міститься відповідний клас винятку. Усередині інструкції except використовується поле об'єкта для виведення необхідного повідомлення.

Результат виконання програми для випадку, якщо сума для зняття не перевищує суму вкладу, наведено на рис. 8, а якщо перевищує суму вкладу — на рис. 9.

```
Сума вкладу: 670
Яку суму зняти? 450
Знято 450 Залишок 220
Програма завершена
```

Рис. 8. Виконання програми для зняття суми, що не перевищує суму вкладу

```
Сума вкладу: 3000
Яку суму зняти? 3600
Зняти суму 3600 не можна
На рахунку залишається 3000
Програма завершена
```

Рис. 9. Виконання програми для зняття суми, яка перевищує суму вкладу

Запитання для перевірки знань

1. Що називають винятком у програмі?
2. Які стандартні винятки можуть виникати?
3. Що повідомляється користувачу, коли виникає виняток?
4. За допомогою якої інструкції перехоплюються винятки?
5. Яку структуру має блок try?
6. Для чого призначена інструкція raise?

Завдання для самостійного виконання

1. Генеруються два цілих випадкових числа в діапазоні 0–2. Перше число ділиться на друге. Розробіть програму моделювання цього процесу, виконайте її кілька разів і проаналізуйте результат.
2. Розробіть програму уведення двох чисел та їх додавання, якщо перше число більше другого, інакше — віднімання другого від першого. Зробіть помилку в записі оператора умовного переходу (виняток класу SyntaxError — синтаксична помилка). Перехопіть, опрацюйте помилку й завершіть програму.
3. Створіть програму множення значень двох змінних, одна з яких не визначена. Використайте виняток NameError — спроба звернення до ідентифікатора до його визначення для перехоплення, опрацювання цієї події і нормального завершення програми.
4. Назви областей України записано в алфавітному порядку. Розробіть програму створення списку з п'яти перших областей цього списку, та звернення до восьмої, відсутньої у списку області. Використайте виняток IndexError, указаний індекс не існує в послідовності для перехоплення винятку, опрацювання та нормального завершення програми.
5. Учитель звертається до учня з проханням назвати дві найменші одиниці вимірювання інформації. Якщо учень дає відповідь «біт» або «байт», учитель каже «правильно». Розробіть програму моделювання цього процесу з опрацюванням винятку введення непередбаченої відповіді. У будь-якому випадку програма має завершити свою роботу.

8. Основи графічного інтерфейсу користувача

8.1. Загальний порядок створення графічного інтерфейсу

До цих пір програми розроблялися в консольному режимі. Які, на вашу думку, дефекти має такий режим виконання програм?

У мовах програмування використання графічного інтерфейсу робить зручнішою роботу з програмою. Графічні об'єкти розміщуються на екрані, деякі з них потребують зовнішньої дії, наприклад клацання кнопки миші. Далі програма виконує відповідну передбачувану дію. Якщо клацнути іншу кнопку — програма може виконувати іншу запрограмовану дію.

Мова Python забезпечує роботу з різними графічними бібліотеками. За допомогою таких бібліотек можна створювати програми з розвинутим графічним інтерфейсом користувача. Достатньо потужні графічні можливості має модуль tkinter, який і розглядатиметься далі.

Модуль tkinter входить до стандартного дистрибутива Python. Основним призначенням цього модуля є створення графічних інтерфейсів для програм, а завдяки наявності об'єкта полотно (Canvas) його можна застосовувати й для малювання. За допомогою модуля tkinter можна створювати також графічні функції. Але для побудови графіків потужніші можливості в модуля matlotlib (який тут не розглядається).

Бібліотека tkinter має стандартний набір об'єктів (їх ще називають віджетами, або компонентами), із яких, власне, і створюється графічний інтерфейс. Прикладами віджетів є кнопка, смуга прокручування, прапорець, перемикач та інші.

Кожному об'єкту в бібліотеці відповідає свій клас. Наприклад, об'єкту кнопка відповідає клас Button, а об'єкту перемикач — клас Radiobutton. Базовим класом бібліотеки є клас Tk, за допомогою якого створюється головне вікно.

Усі віджети в програмі створюються за однаковим принципом — шляхом виклику конструктора відповідного класу, який має таку загальну структуру:

<ім'я змінної> = <назва класу> [(параметри)]

Наприклад, створити кнопку можна за допомогою інструкції: but1 = Button(). Першим параметром у цьому конструкторі вказується ім'я батьківського віджета, у якому буде розміщено цей об'єкт. Якщо цей параметр не вказано, то об'єкт буде розміщено в головному вікні. Далі можуть міститися інші параметри, які визначають його конфігурацію, наприклад, назву — text, колір — bg, шрифт — font та інші.

Щоб скористатися можливостями графічного інтерфейсу, слід передбачити розміщення у вікні необхідних графічних об'єктів, а також ті дії, які мають виконуватися в процесі взаємодії користувача з цими об'єктами. Безумовно, це збільшує обсяг програмного коду й ускладнює його структуру, але суттєво спрощує користувачеві взаємодію з програмою. А інколи без використання графічних об'єктів просто не можна обійтися.

Наприклад, неможливо уявити роботу банкомата або платіжного термінала без висвітлення на екрані різних компонентів, за допомогою яких практично будь-який користувач може взаємодіяти з програмами, які ці пристрої обслуговують. Використовуючи графічні об'єкти, користувачі навіть можуть і не знати про сутність програми, із якою він взаємодіє. Таким прикладом є програма управління сучасною пральною машиною.

Для створення графічного інтерфейсу користувача необхідно знати перелік об'єктів бібліотеки tkinter, їх призначення й можливості, порядок розміщення об'єктів у вікні та події, які можуть виникати під час взаємодії з ними. Саме в такій послідовності й розроблено цей підрозділ.

Опишемо *алгоритм створення графічного інтерфейсу* мовою Python.

Крок 1	Імпортуйте модуль tkinter і створіть головне вікно. Як правило, для імпортування модуля використовують інструкцію from tkinter import *. Далі створюється головне вікно. Пригадаємо, що воно міститься в класі Tk. Змінна, яку пов'язують з об'єктом цього вікна, зазвичай має ім'я root. Найпростіша інструкція для створення головного вікна має таку структуру: root = Tk().
Крок 2	Створіть у програмі віджети й визначте їх властивості. Пригадаємо, що кожен віджет має свій клас. Будь-який віджет повинен бути пов'язаний із відповідним іменем змінної. Наприклад, кнопку Button будемо пов'язувати зі змінною but за допомогою інструкції: but=Button(root). Тут аргумент root означає, що кнопка буде розміщена в головному вікні. За замовчуванням кнопку також буде розміщено в головному вікні. Але якщо кнопку необхідно розмістити в іншому вікні, то слід указати його ім'я. Будь-який віджет має властивості. Наприклад, кнопка може мати розмір, колір, напис. Конкретні значення властивостей віджета вказуються як значення відповідних параметрів після назви вікна (наприклад, після root) або встановлюються після їх створення. Наприклад, для кнопки, якій присвоєно ім'я but, можна встановити властивість **напис** за допомогою інструкції: but=Button (root, text="Виконати").
Крок 3	Визначте події та порядок їх опрацювання. Для кожного компонента можна передбачити деякі події. Наприклад, кнопка може бути натиснута або змінено її розмір. У програмі необхідно передбачити опрацювання кожної події, наприклад, які дії слід виконати під час натискання кнопки. Опрацювання подій часто реалізується у вигляді функцій, які викликаються під час виникнення відповідної події. Функції у цьому випадку мають таку загальну структуру: def <назва функції> (<назва події>): <тіло функції> Функція зазвичай створюється на початку програмного коду. Щоб можна було викликати функцію опрацювання події, необхідно зв'язати цю функцію із самою подією. Зв'язування реалізується за допомогою методу bind. Наприклад, зв'язати подію натиснення кнопки Button_1 із функцією func_1 можна за допомогою інструкції: but.bind ("Button_1".<func_1>).
Крок 4	У головному вікні розміщуються віджети, вони можуть бути навіть у строго визначених координатах вікна. Щоб розмістити віджети без визначення координат, використайте метод pack(). Наприклад, розмістити кнопку у вікні можна за допомогою інструкції: but.pack().
Крок 5	Головне вікно відображається на екрані за допомогою методу mainloop(). Виконайте інструкцію root.mainloop(), яка має бути останньою в програмі.

Приклад 1.

На рис. 1 зображено код програми, за допомогою якої відображається головне вікно з двома кнопками. Одна з них має напис Множення, а друга — Додавання. Якщо натиснути кнопку Множення, то викличеться функція func2, уведуться й помножаться два числа. Якщо натиснути кнопку Додавання, то викличеться функція func1, уведуться й додадуться два інших числа.

```
from tkinter import *                  # імпортування модуля tkinter

def func1(eve):                        # функція func1
    a = int(input("Увести a: "))       # інструкції
    b = int(input("Увести b: "))       # тіла
    print("a+b = ", a + b)             # функції func1

def func2(fff):                        # функція func2
    c = int(input("Увести c: "))       # інструкції
    d = int(input("Увести d: "))       # тіла
    print("c*d = ", c * d)             # функції func2

root = Tk()                            # створення головного вікна
but1 = Button(root)                    # створення кнопки but1 у вікні
but1["text"] = "Додавання"             # напис на кнопці but1
but1.bind("<Button>", func1)           # зв'язування кнопки but1 із функцією func1
but2 = Button(root)                    # створення кнопки but2 у вікні
but2["text"] = "Множення"              # напис на кнопці but2
but2.bind("<Button>", func2)           # зв'язування кнопки but2 із функцією func2
but2.pack()                            # розміщення у вікні кнопки but2
but1.pack()                            # розміщення у вікні кнопки but1
root.mainloop()                        # відображення на екрані вікна
```

Рис. 1. Код програми відображення вікна з двома кнопками

Після запуску програми на виконання на екрані з'явиться вікно з двома кнопками (рис. 2) та вікно, у яке необхідно увести дані для опрацювання. Вікно для уведення даних можна перемістити в інше місце екрана так, щоб ці вікна не перетиналися.

Натиснемо спочатку, наприклад, копку Додавання й уведемо два числа — результат додавання чисел відобразиться на екрані. Потім натиснемо кнопку Множення й уведемо два числа — з'явиться результат множення. Ще раз натиснемо кнопку Додавання й уведемо два числа. Можливий результат описаних взаємодій із програмою наведено на рис. 3.

Якщо у вікні (див. рис. 2) натиснути ліву верхню кнопку, то відкриється меню, зображене на рис. 4.

За допомогою команд цього меню можна перемісти вікно на інше місце екрана або виконати інші операції.

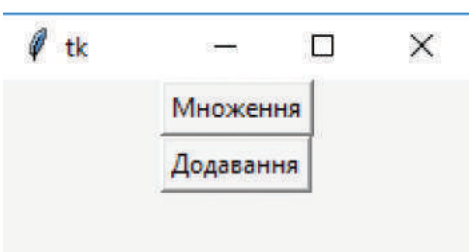


Рис. 2. Вікно з двома кнопками

```
Увести a: 8
Увести b: 45
a+b =  53
Увести c: 7
Увести d: 9
c*d =  63
Увести a: 33
Увести b: 9
a+b =  42
```

Рис. 3. Результат взаємодій із програмою

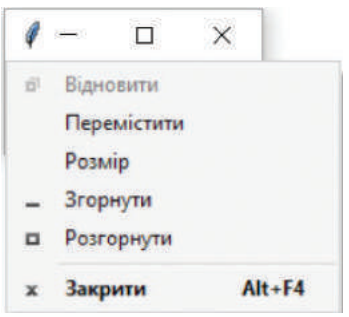


Рис. 4. Системне меню

Створювати графічний інтерфейс користувача можна й на основі об'єктно-орієнтованого підходу, обов'язковою складовою якого є клас. Такий підхід у деяких випадках є ефективнішим за метод процедурного програмування.

Приклад 2.

На рис. 5 зображено код програми, яка реалізує ті самі функції, що й програма, яку зображено на рис. 1. Але вона побудована з використанням класу.

Принципова різниця між цими програмами полягає лише в тому, що числа в другій програмі не вводяться за допомогою клавіатури.

```
from tkinter import *                          # імпортування модуля tkinter

class Kl9_04:                                  # клас Kl9_02
    def __init__(self, a, b, c, d):           # конструктор класу Kl9_02
        self.s = a + b                         # складання чисел a і b
        self.p = c * d                         # множення чисел c і d
        self.but1 = Button(root)               # створення кнопки but1
        self.but1["text"] = "Додавання"        # напис на кнопці but1
        self.but1.pack()                       # розміщення у вікні кнопки but1
        self.but2 = Button(root)               # створення кнопки but2
        self.but2["text"] = "Множення"         # напис на кнопці but2
        self.but2.pack()                       # розміщення у вікні кнопки but2
 # зв'язування кнопок but1 і but2 відповідно з методами func1 і func2
        self.but1.bind("<Button-1>", self.func1)
        self.but2.bind("<Button-1>", self.func2)

    def func1(self, rkt):                      # метод func1
        print("a+b = ", self.s)                # виведення суми чисел
    def func2(self, rkt):                      # метод func2
        print("c*d = ", self.p)                # виведення добутку чисел

root = Tk()                                    # створення головного вікна
obj = Kl9_04(3, 9, 6, 7)                       # об'єкт класу Kl9_02
root.mainloop()                                # відображення на екрані вікна
```

Рис. 5. Код програми на основі ООП будови графічного інтерфейсу

На рис. 6 наведено можливий варіант виконання програми.

```
c*d =  42
a+b =  12
c*d =  42
c*d =  42
a+b =  12
```

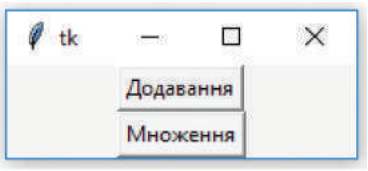


Рис. 6. Виконання програми на основі ООП будови графічного інтерфейсу

Запитання для перевірки знань

1. Для чого призначений модуль tkinter?
2. Які основні об'єкти містить модуль tkinter?
3. За допомогою якої інструкції віджети розміщуються у вікні?
4. Яку загальну структуру має функція опрацювання події?
5. Які переваги й дефекти має графічний інтерфейс користувача?
6. Поясніть загальний порядок створення графічного інтерфейсу користувача.
7. Поясніть особливості створення графічного інтерфейсу користувача на основі об'єктно-орієнтованого підходу.

Завдання для самостійного виконання

1. Розробіть програму, за допомогою якої у головному вікні розміщується кнопка та здійснюється введення з клавіатури трьох чисел. Після натиснення кнопки обчислюється середнє арифметичне уведених чисел.
2. Розробіть програму, за допомогою якої у головному вікні розміщується кнопка, після натиснення якої генеруються два цілі випадкові числа, множаться й результат виводиться на екран.
3. Розробіть програму, за допомогою якої у головному вікні розміщуються дві кнопки з назвами Увести та Ділення. Після натиснення першої кнопки два числа уводяться з клавіатури, а після натиснення другої перше число ділиться на друге.
4. Розробіть програму, за допомогою якої у головному вікні розміщуються три кнопки. Після натиснення першої кнопки уводиться ціле число, після натиснення другої — генерується випадкове число, а після натиснення третьої — вони складаються.
5. Розробіть програму на основі об'єктно-орієнтованого підходу, за допомогою якої у головному вікні розміщуються кнопки Менше і Більше. Після натиснення першої кнопки з клавіатури уводяться два числа й повідомляється, яке з них менше. Після натиснення другої кнопки з клавіатури уводяться два інших числа й повідомляється, яке з них більше.

8.2. Графічні об'єкти і їх властивості

Під час роботи з ОС Windows використовують стандартні елементи, наприклад, меню, смугу прокручування й ін. Бібліотека мови Python також має необхідний перелік графічних елементів, на основі яких створюється графічний інтерфейс користувача. Спробуйте визначити перелік таких елементів.

Графічні об'єкти бібліотеки tkinter можна розподілити на дві групи: найчастіше вживані, які назвемо *основними*, і ті, які використовуються рідше (назвемо їх *додатковими*).

8.2.1. Основні графічні об'єкти

Ознайомимося з основними графічними об'єктами бібліотеки tkinter (рис. 1).

- Кнопки зазвичай використовуються для управління програмою. Після натиснення кнопки може опрацьовуватися та чи інша подія, передбачена в коді програми. Кнопці відповідає клас Button.

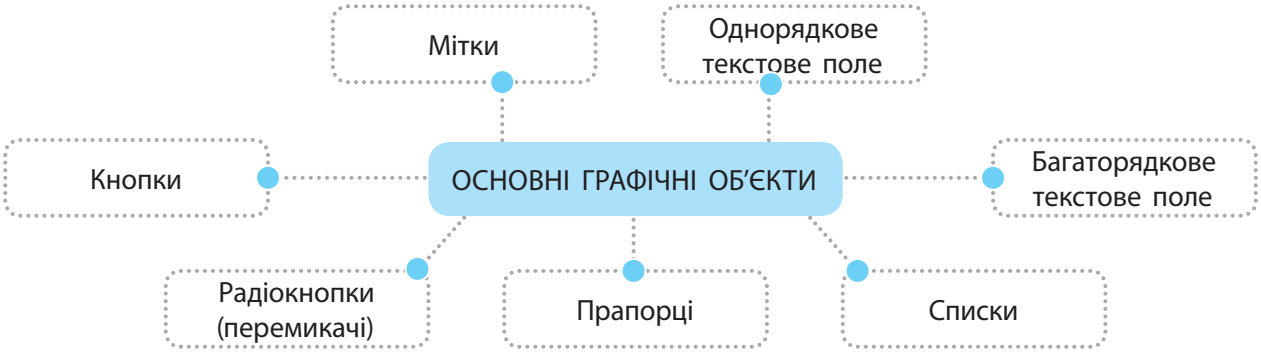


Рис. 1. Об'єкти бібліотеки tkinter

Загальна структура конструктора створення кнопки така:

<змінна> = Button [(<параметр_1>,<параметр_2>,<параметр_3,…>)],

де параметр_1 (якщо він указаний) називають батьківським. Цей параметр визначає посилання на об'єкт, у якому створюється кнопка.

Наприклад, якщо кнопка створена в головному вікні, то цей параметр має значення root. За замовчуванням кнопка також створюється в головному вікні.

Інші параметри визначають властивості кнопки та їх значення. Ці параметри оголошуються за такою структурою:

<властивість=значення>.

Приклад 1.

На рис. 2 зображено код програми, за допомогою якої у головному вікні відображаються дві кнопки різних розмірів, які мають різний колір фону й назви.

```
from tkinter import *                          # імпортування модуля tkinter
root = Tk()                                    # створення головного вікна
but1 = Button(root,                            # створення кнопки but1 у вікні
              text = "Це кнопка 1",            # напис на кнопці 1
              width = 15, height = 2,          # ширина і висота кнопки but1
              bg = "blue", fg = "white")       # колір фону кнопки but1 і напису
but2 = Button(root,                            # створення кнопки but2 у вікні
              text = "Це кнопка 2",            # напис на кнопці 2
              width = 30, height = 3,          # ширина і висота кнопки but2
              bg = "yellow", fg = "red")       # колір фону кнопки but2 і напису
but1.pack()                                    # розміщення у вікні кнопки but1
but2.pack()                                    # розміщення у вікні кнопки but2
root.mainloop()                                # відображення на екрані вікна
```

Рис. 2. Код відображення двох кнопок із різними значеннями властивостей

Тут bg — скорочення від background (фон), а fg — скорочення від foreground (передній план). Ширина й висота вимірюються кількістю знакових місць.

Результат виконання програми подано на рис. 3.

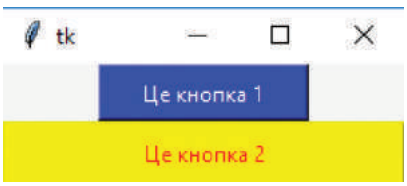


Рис. 3. Виконання програми

Приклад 2.
За допомогою інструкції:

lab = Label(root, text="виконана функція func1", font="Arial 12")

у головне вікно буде виведено 12 кеглем шрифту Arial повідомлення «виконана функція func1».

Приклад 3.
Створити однорядкове текстове поле можна за допомогою такої інструкції:

ent = Entry(root, width=18, bd=5)

тут ent — ім'я змінної, із яким пов'язується текстове поле, bd — скорочення від borderwidth (ширина межі).

Приклад 4.
Створити багаторядкове текстове поле можна за допомогою інструкції:

tex = Text(root, width=30, font="Arial 12", wrap=Word)

Тут властивість wrap визначає, що слова не будуть розриватися в процесі перенесення тексту на новий рядок.

За допомогою інструкції **var = IntVar()** створюється об'єкт **var**, який виконує роль однойменної змінної. За допомогою методу **set()** цій змінній можна надати початкове значення. Наприклад, за допомогою інструкції **var.set(1)** об'єкт-змінна **var** набуде значення **1**.

- Мітки використовуються здебільшого для інформування користувача про результати виконаних дій програмою, або про ті дії, які має виконати користувач.
  Мітки можуть містити один або кілька рядків. Для їх створення викликають клас Label.

Загальна структура інструкції для створення мітки в головному вікні така:

<змінна> = Label(root, <параметр_1 >, <параметр_2>,…)

Змінна може мати будь-яке ім'я. Далі будемо використовувати ім'я lab. Параметри мають таку саму структуру, як і параметри кнопки (приклад 2).

- Однорядкове текстове поле використовується для введення тексту користувачем в один рядок. Для його створення викликається клас Entry (приклад 3).
- Багаторядкове текстове поле також призначене для введення тексту користувачем. Для його створення викликається клас Text (приклад 4).
- Радіокнопки (перемикачі) призначені для вибору одного з кількох запропонованих режимів (варіантів). Таким чином, не може бути однієї радіокнопки, лише кілька. У кожен окремий момент увімкнутою може бути лише одна кнопка.

Відзначимо, що в бібліотеці tkinter є класи, об'єкти яких виконують роль змінних для збереження значень про стан віджетів. Якщо змінити значення такої змінної, то зміниться і властивість віджета, а зміна властивості віджета викличе зміну значення відповідної змінної.

До таких класів належать:
StringVar() — для опрацювання рядків;
IntVar() — для опрацювання цілих чисел;
DoubleVar() — для опрацювання дійсних чисел;
BoolenVar() — для опрацювання булевих значень.

Для створення радіокнопок викликається клас Radiobutton. Створити дві радіокнопки можна, наприклад, за допомогою таких інструкцій:

```
var = IntVar(0)
var.set(1)
rad1 = Radiobutton(root, text='Перша', variable=var, value=1)
rad2 = Radiobutton(root, text='Друга',variable=var, value=2)
```

За допомогою цих інструкцій створюється об'єкт var класу IntVar, який виконує роль змінної.

За допомогою методу set() початкове значення змінної установлюється як таке, що дорівнює 1.

Властивість variable зв'язує змінну з радіокнопкою, а властивість value визначає значення, яке буде передано змінній, якщо ця кнопка буде ввімкнута. Ці дві кнопки належать одній групі, про що свідчить однакове значення властивості variable.

- Прапорці, як і радіокнопки, можуть міститися в різній кількості. Одночасно можна увімкнути кілька прапорців, а можна не вмикати жодного. Для створення прапорців

викликається клас Checkbutton. Кожен прапорець повинен мати власну змінну. Створити два прапорці можна за допомогою певних інструкцій (приклад 5).

- **Списки** — це об'єкти (поля), у яких користувач може вивести необхідні дані та вибрати з них один або кілька пунктів залежно від значення параметра selectmode. Для створення списку викликається клас Listbox.
Розглянемо такий фрагмент інструкцій:

```
a = ['кнопка', 'мітка', 'прапорець', 'перемикач']
lis = Listbox(root, selectmode=SINGLE, height=3)
for i in a:
    lis.insert(END, i)
```

За допомогою першої інструкції оголошується тип даних список з іменем a. Він містить чотири рядки. За допомогою другої інструкції створюється об'єкт-список, який пов'язаний зі змінною lis. Після створення об'єкта-списку він є порожнім. Заповнюється список значеннями за допомогою методу insert в операторі циклу. Значення SINGLE параметра selectmode створює можливість вибрати з чотирьох виведених у список рядків лише один.

Приклад 5.
Створити два прапорці можна, наприклад, за допомогою таких інструкцій:

```
a1 = IntVar()
a2 = IntVar()
ch1 = Checkbutton(root,
text="Перший", variable=a1,
onvalue=1, offvalue=0)
ch2 = Checkbutton(root, text =
"Другий", variable=a2, onvalue=2,
offvalue=0)
```

Приклад 6.
Розробимо програму, за допомогою якої в головному вікні відображаються такі об'єкти: два прапорці (з назвами, відповідно, Перший і Другий), кнопка з назвою Стан прапорців і об'єкт-список, у який виводиться стан прапорців. У цьому списку можна вибрати один рядок. Програму, що реалізує це завдання, зображено на рис. 4.

```
from tkinter import * # імпортування модуля tkinter
root = Tk()           # створення головного вікна
var1 = StringVar()    # змінна для збереження значення першого прапорця
var2 = StringVar()    # змінна для збереження значення другого прапорця
# створення першого прапорця з відповідними значеннями параметрів
prap1 = Checkbutton(root, text = "Перший", variable = var1,
                    onvalue="перший увімкнутий", offvalue="перший вимкнутий")
# створення другого прапорця з відповідними значеннями параметрів
prap2 = Checkbutton(root, text = "Другий", variable = var2,
                    onvalue="другий увімкнутий", offvalue="другий вимкнутий")
lis = Listbox(root, bg="yellow", height = 3) # створення об'єкта-списку
def func1(ven):                  # функція func1
    v1 = var1.get()              # змінна v1 набуває значення першого прапорця
    v2 = var2.get()              # змінна v2 набуває значення другого прапорця
    l = [v1, v2]                 # створюється список зі значень змінних
    lis.delete(0,1)              # очищення попередніх значень об'єкта lis
    for i in l:                  # циклічне наповнення об'єкта lis
        lis.insert(END, i)       # значення списку l вставляються в об'єкт lis
but = Button(root, text="Стан прапорців")  # створення кнопи з написом
but.bind('<Button-1>', func1)    # зв'язування кнопки but з функцією
prap1.deselect()                 # прапорець prap1 за замовчуванням знятий
prap2.deselect()                 # прапорець prap1 за замовчуванням знятий
prap1.pack()                     # розміщення у вікні прапорця prap1
prap2.pack()                     # розміщення у вікні прапорця prap2
but.pack()                       # розміщення у вікні кнопки but
lis.pack()                       # розміщення у вікні списку
root.mainloop()                  # відображення вікна на екрані
```

Рис. 4. Код програми зміни й відображення стану прапорців

Після запуску програми відкриється вікно з об'єктами (рис. 5). Увімкнемо, наприклад, другий прапорець і натиснемо кнопку Стан прапорців. Отримаємо результат, який зображено на рис. 6.

У функції func1 за допомогою методу get() стан першого прапорця присвоюється змінній z1, а другого — змінній z2.

За допомогою інструкції l = [z1, z2] зі значень цих змінних створюється тип список l. Після очищення об'єкта lis він заповнюється відповідними значеннями за допомогою оператора циклу.

Стан першого прапорця запам'ятовується в змінній var1, а другого — у змінній var2.

Після натиснення кнопки Стан прапорців здійснюється звернення до функції func1.

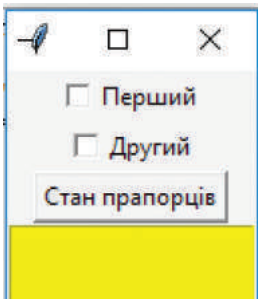


Рис. 5. Стан об'єктів вікна після запуску програми

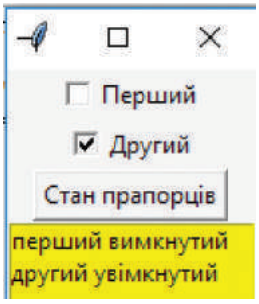


Рис. 6. Стан об'єктів після вмикання прапорця і натиснення кнопки but

8.2.2. Додаткові графічні об'єкти

Ознайомимося з додатковими графічними об'єктами бібліотеки tkinter (рис. 7).

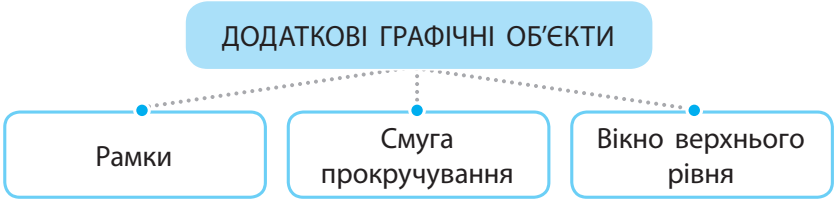


Рис. 7. Об'єкти бібліотеки tkinter

- Рамки (фрейми) використовуються для об'єднання у групи об'єктів, які розміщено у вікні. У вікні може бути створено кілька рамок. Для створення рамки викликається клас Frame.

Загальна структура інструкції для створення рамки в головному вікні така:

<змінна> = Frame(root, <параметр_1>,<параметр_2>,…)

Наприклад, рамку можна створити за допомогою такої інструкції:

fr1 = Frame(root, width=300, height=100, bg="darkblue", bd=20)

Тут bg — скорочення від boderwidth означає відстань від кромки рамки до віджетів у ній (якщо вони є).

Американська розробниця програмного забезпечення Радія Джой Перлман стала першою, хто почав навчати програмуванню дітей молодшого віку. Вона розробила дитячу версію навчальної робототехнічної мови LOGO (названу TORTIS).

Приклад 7.

На рис. 8 зображено код програми, за допомогою якого в головному вікні створюються дві рамки. У першій із них розміщено дві кнопки з написами Увімкнути й Вимкнути, у другій — одна кнопка з написом Завершити роботу. Результат виконання програми подано на рис. 9.

```
from tkinter import *                          # імпортування модуля tkinter
root = Tk()                                    # створення головного вікна
fr1 = Frame(root, bg='yellow', bd=20)          # створення першої рамки
fr2 = Frame(root, bg='magenta', bd=40)         # створення другої рамки
but1 = Button(fr1,text='Увімкнути')            # кнопка but1 у першій рамці
but2 = Button(fr1,text='Вимкнути')             # кнопка but2 у першій рамці
but3 = Button(fr2, text='Завершити роботу') # кнопка but3 у другій рамці
fr1.pack()                                     # розміщення у вікні першої рамки
fr2.pack()                                     # розміщення у вікні другої рамки
but1.pack()                                    # розміщення кнопки but1 у вікні
but2.pack()                                    # розміщення кнопки but2 у вікні
but3.pack()                                    # розміщення кнопки but3 у вікні
root.mainloop()                                # відображення на екрані вікна
```

Рис. 8. Код відображення двох рамок і трьох кнопок

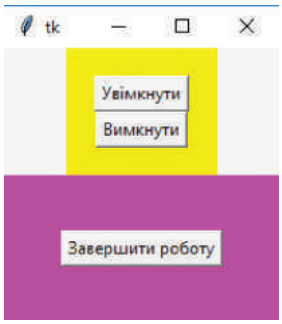


Рис. 9. Виконання програми

- Смуга прокручування використовується для прокручування вмісту іншого віджета, наприклад, списку або текстового поля.

Прокручування може бути як по горизонталі, так і по вертикалі. Перед створенням смуги прокручування слід визначити об'єкт (список або текстове поле), для якого вона створюється. Для створення смуги прокручування викликається клас Scrollbar. Наприклад, створити вертикальну смугу прокручування для текстового поля (попередньо створеного) можна за допомогою інструкції:

```
sc = Scrollbar(root, command=tx.yview)
```

Тут tx — змінна текстового поля, yview — вертикальне розташування смуги прокручування, параметр command зв'язує вертикальну смугу прокручування з текстовим полем tx.

- Вікно верхнього рівня є дочірнім головного вікна й найчастіше застосовується для групування об'єктів за призначенням. Наприклад, в одному вікні можуть розміщуватися перемикачі, у другому — прапорці.

На екрані можна відобразити кілька таких вікон. Закриття головного вікна призведе й до закриття дочірніх, а закриття дочірніх не потягне закриття головного вікна. Для створення вікна верхнього рівня викликається клас Toplevel.

Створити дочірнє вікно можна, наприклад, за допомогою такої інструкції:

```
top = Toplevel(root, relief=SUNKEN, width=200, height=100,
bd=15, bg="lightblue")
```

а визначити заголовок вікна можна за допомогою методу title, наприклад:

```
top.title(«Перше додаткове вікно»)
```

Параметр relief визначає рельєф вікна.

Марісса Енн Маєр відома тим, що в 1999 році стала першою жінкою-інженером «Google» і тривалий час посідала керівні посади в компанії. У 2012–2018 роках Маєр була президенткою і головною виконавчою директоркою компанії «Yahoo!». Зараз працює над новим проектом.

Приклад 8.

На рис. 10 зображено програмний код, за допомогою якого створюється головне вікно та два дочірніх вікна: Вікно прапорців і Вікно перемикачів. У головному вікні розташована кнопка Start, у вікні перемикачів — два перемикачі: Тип і Ім'я, а у вікні прапорців — два прапорці: Без заливки і Градієнтна заливка.

```
from tkinter import *                    # імпортування модуля tkinter
root = Tk()                              # створення головного вікна
top1 = Toplevel(root, bd=120, bg="yellow")# перше вікно верхнього рівня
top1.title("Вікно прапорців")            # заголовок першого дочірнього вікна
top1.minsize(width=200, height=60)       # мінімальний розмір вікна top1
top2 = Toplevel(root, bd=120, bg="lightblue") # друге вікно верхнього рівня
top2.title("Вікно перемикачів")          # заголовок другого дочірнього вікна
top2.minsize(width=200, height=60)       # мінімальний розмір вікна top2
a1 = StringVar()          # змінна для збереження значення першого прапорця
a2 = StringVar()          # змінна для збереження значення другого прапорця
# створення першого прапорця з відповідними значеннями параметрів
prap1 = Checkbutton(top1, text="Тип", variable=a1)
# створення другого прапорця з відповідними значеннями параметрів
prap2 = Checkbutton(top1, text = "Ім'я", variable=a2)
a3 = IntVar()             # змінна для збереження значення перемикача
# створення перемикачів, зв'язаних із змінною a3, у другому дочірньому вікні
rad1=Radiobutton(top2,text="    Без заливки   ", variable=a3, value=1)
rad2=Radiobutton(top2,text="Градієнтна заливка", variable=a3, value=2)
# створення у головному вікні кнопки з написом Srart
but = Button(root, text='Start', width=30, height=5, bg='white', fg='red')
prap1.deselect() # очищення прапорця prap1
prap2.deselect() # очищення прапорця prap2
but.pack()       # розміщення у головному вікні кнопки
prap1.pack()     # розміщення у першому дочірньому вікні першого прапорця
prap2.pack()     # розміщення у першому дочірньому вікні другого прапорця
rad1.pack()      # розміщення у другому дочірньому вікні першого перемикача
rad2.pack()      # розміщення у другому дочірньому вікні другого перемикача
root.mainloop()  # відображення на екрані вікна
```

Рис. 10. Код відображення різних типів вікон

Після виконання програми на екрані відобразяться вікна (рис. 11).

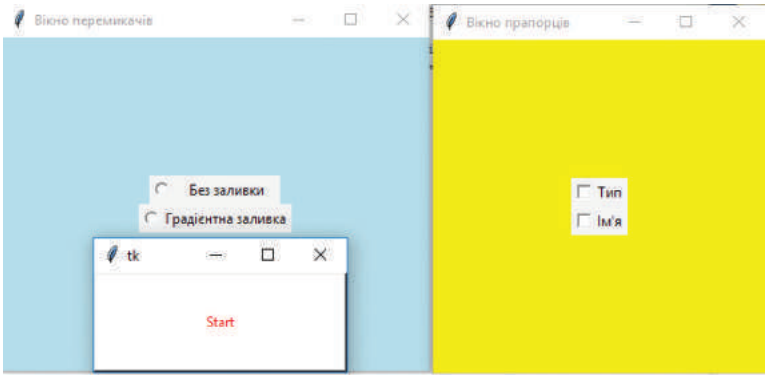


Рис. 11. Вікна перемикачів і прапорців

Запитання для перевірки знань

1. Яке призначення текстового поля, радіокнопки?
2. Поясніть призначення параметрів мітки.
3. Для чого призначені списки, рамки?
4. Наведіть приклад інструкції створення радіокнопки.
5. Як створюються прапорці?
6. Яку загальну структуру має конструктор створення кнопки?
7. Як створюються рамки?
8. Наведіть приклад інструкції створення смуги прокручування.
9. Які вікна належать до верхнього рівня?

Завдання для самостійного виконання

1. Розробіть програму, за допомогою якої у головному вікні розміщується кнопка з назвою Виконати і мітка з повідомленням Завершено.
2. Розробіть програму, за допомогою якої у головному вікні розміщуються однорядкове, багаторядкове текстові поля і дві радіокнопки.
3. Створіть програму, за допомогою якої для багаторядкового текстового поля створюється вертикальна смуга прокручування.
4. Розробіть програму, за допомогою якої у головному вікні розміщується кнопка, два прапорці та об'єкт-список. Після натиснення кнопки в об'єкт-список виводяться рядки: Черкаси, Чернігів, Чернівці.
5. Розробіть програму, за допомогою якої створюються два дочірніх вікна. В одному з них розміщуються дві радіокнопки, а в другому — два прапорці.

8.3. Опрацювання подій

Із подіями нам неодноразово доводилося стикатися в процесі роботи з ОС. Наприклад, після натиснення кнопки Друк відкритий файл виводиться на принтер, а після виконання підпункту Save зберігається. У мові Python також існує значна кількість подій і способів їх опрацювання.

Раніше вже було описано склад, призначення й технологію розміщення об'єктів бібліотеки tkinter у вікнах. Перейдемо до розгляду методики використання об'єктів графічного інтерфейсу для управління програмою. Описаний склад об'єктів графічного інтерфейсу мови Python фактично є стандартом для багатьох мов ООП. Разом із тим, порядок їх розміщення у вікнах і методика використання в різних мовах суттєво відрізняється.

Зовнішнє діяння на віджет називають подією. У мові Python подій досить багато. Тут розглядаються лише ті, які застосовуються найчастіше.

Події можна розподілити на три основні групи.

- **Події, що виникають у результаті діяння на мишу.** Назви подій цього типу беруться в лапки і знаки <>. Наведемо деякі події:

  "<Button-1>" — клацання лівою кнопкою миші;
  "<Button-3>" — клацання правою кнопкою миші;
  "<Motion>" — рух миші.

У програмах користувача з використанням графічного (віконного) інтерфейсу на екрані мають висвітлюватися віджети, передбачені в програмі. Програма має зупинити своє виконання й очікувати зовнішнього діяння на віджети. Подальша реакція програми залежить від того, на який віджет і яке виконане діяння.

Приклад 1.
Натиснення клавіші Enter позначається так: <Return>, а клавіші пробіл — <space>.
Натиснення клавіш Ctrl+Shift позначається так:
"<Control-Shift>",
а тиснення клавіш Ctrl+z так:
"<Control-z>".

- Події, що виникають у результаті діяння на клавіші. Клавіші букв записуються в лапках, наприклад: "W". Для неалфавітних клавіш існують спеціальні зарезервовані слова. Для клавіш, які натискуються одночасно, також існують спеціальні позначення (приклад 1).
- Події, що виникають у результаті зміни властивостей інших об'єктів. Слід чітко усвідомити, що для опрацювання тієї чи іншої події попередньо необхідно зв'язати віджет, над яким виконується діяння, саму подію і функцію, яка повинна опрацьовувати цю подію. Їх зв'язування реалізується спеціальними методами, наприклад, методом bind, який нами вже використовувався в програмі.

Розглянемо опрацювання подій на прикладах.

Приклад 2.

Нам відомі назви футбольних команд — чемпіонів України в 2010–2016 роках. Розробимо програму, за допомогою якої після уведення назви команди виводяться роки, у які вона була чемпіоном країни. Якщо команда не була чемпіоном у цей період, має висвітлюватися відповідне повідомлення.

Для реалізації цього завдання використаємо однорядкове текстове поле для введення назви команди, багаторядкове текстове поле для відображення років, кнопку та подію "<Button>".

Відповідний програмний код зображено на рис. 1.

```
from tkinter import *  # імпортування модуля tkinter
root = Tk()            # створення головного вікна
def func1(ven):        # функція func1
    s = ent.get()      # змінна s набуває вмісту однорядкового текст. поля
    if s == "Динамо":  # якщо введена назва Динамо
        tex.delete(1.0, END)                    # видалення тексту із текстового поля
# у текстове поле вставляються ті роки, коли Динамо було чемпіоном
        tex.insert(END, "Чемпіон України у\
                   2014-2015, 2015-2016 роках")
    elif s == "Шахтар":                         # Якщо введена назва Шахтар
        tex.delete(1.0, END)                    # видалення тексту із текстового поля
# у текстове поле вставляються ті роки, коли Шахтар був чемпіоном
        tex.insert(END, "Чемпіон Украни у\
                   2010-2011, 2011-2012, 2012-2013, 2013-2014 роках")
    else:                                       # якщо уведена інша назва команди
        tex.delete(1.0, END)                    # видалення тексту із текстового поля
        tex.insert(END, "Ця команда не була чемпіоном")
ent = Entry(root, width=10, bg='yellow')      # створ. однорядового текст. поля
but = Button(root, text ="Виконати", bg='magenta')      # створення кнопки
# створення багаторядкового текстового поля
tex = Text(root, width=20, bg='lightblue', height=4, font="12", wrap=WORD)
ent.grid(row=0, column=0, padx=2, pady=10)    # розміщення одн. текст. поля
but.grid(row=0, column=1, padx=12, pady=10)   # розміщення у вікні кнопки
tex.grid(row=0, column=2, padx=20, pady=10)   # розміщення багат. текст. поля
but.bind("<Button-1>", func1)      # зв'язування події натиснення кнопки з func1
root.mainloop()                    # відображення на екрані вікна
```

Рис. 1. Код із використанням текстових полів і кнопки

У програмі використано метод grid, за допомогою якого віджети розміщуються не довільно, а за значеннями параметрів padx і pady. Їх значення вказують кількість пікселів від віджета до краю рамки (або комірки) відповідно по осях x і y.

Метод delete() очищує багаторядкове текстове поле. Значення 1.0 і END у методі означають, що очищення слід виконати, починаючи з першого рядка першого символу (нумерація символів починається з нуля) і до кінця.

Метод insert() заповнює це поле значенням змінної s. Ця змінна за допомогою методу set () отримує значення, яке введене в однорядкове текстове поле. Після введення назви команди слід натиснути кнопку Виконати. Можливий варіант виконання програми наведено на рис. 2.

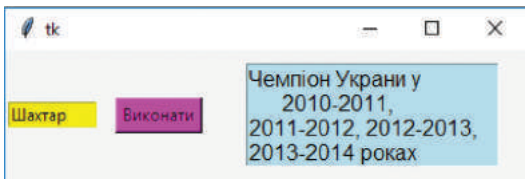


Рис. 2. Виконання програми із використанням текстових полів і кнопки

Приклад 3.

Банкомат функціонує в спрощеному режимі: після введення ПІН-коду він може повідомити лише суму вкладу (баланс). Видачу коштів він не реалізує. Розробимо програму моделювання такого режиму роботи банкомата.

Програму моделювання такого режиму роботи банкомата можна розробити за допомогою різних об'єктів, подій і методів. Один із варіантів програми зображено на рис. 3. Після запуску програми в головному вікні висвітиться мітка lb1 із написом ПІН-код і порожній однорядковий текстовий рядок ent для введення ПІН-коду. Після введення ПІН-коду слід натиснути кнопку Enter. Якщо уведено правильний ПІН-код (1234), управління буде передано функції func1, у результаті чого з головного вікна будуть видалені мітка й текстове поле, пов'язані, відповідно, зі змінними lb1 і ent, і з'явиться кнопка but із написом Баланс.

Програма зупиняє виконання й очікує натиснення цієї кнопки. Після її натискання в головному вікні з'явиться мітка lb2 із повідомленням: На Вашому рахунку 2000 грн. Якщо уведено неправильний ПІН-код, програма видає повідомлення: Неправильний код, Завершено!

```
from tkinter import *  # імпортування модуля tkinter
root = Tk()            # створення головного вікна
def func1(ven):        # функція func1
    s = ent.get()      # змінна s набуває вмісту однорядкового текстового поля
    if s == "1234":    # якщо уведено правильний ПІН-код
        lb1.destroy()  # видалення із вікна мітки
        ent.destroy()  # видалення із вікна однорядкового текстового поля
        but = Button(root, bg='red', text="Баланс") # створення кнопки
        but.pack()                                  # розміщення у вікні кнопки but
        but.bind("<Button-1>", func2)               # зв'язування події Button-1 і func2
    else:                                           # якщо неправильний ПІН-код
        lb1.configure(text="Неправильний код.  ЗАВЕРШЕНО!")
def func2(ven):                                     # функція func2
    lb2 = Label(root, width=40, bg='yellow', text="На Вашому рахунку 2000 грн")
    lb2.pack()                                      # розміщення у вікні мітки lb2
lb1 = Label(root, bg='red',text = "ПІН-код", font="Arial 10") # створення lb1
lb1.grid(padx=30,pady=1)                            # розміщення мітки lb1 у вікні
ent = Entry(root, width=20, bg="yellow")            # створення одн. текст. поля
ent.grid(row=0, column=1, padx=5, pady=20)          # розміщення одн. текст. поля
ent.bind('<Return>', func1)                         # зв'язування події Enter і func1
root.mainloop()                                     # відображення на екрані вікна
```

Рис. 3. Код програми моделювання спрощеного режиму роботи банкомата

Отже, принципова особливість програми в тому, що об'єкти створюються як у головній програмі, так і у функції func2.

Один із варіантів результату виконання програми наведено на рис. 4.

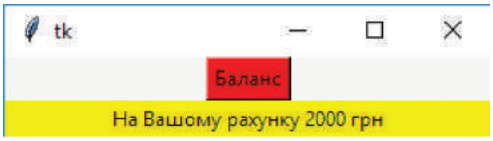


Рис. 4. Виконання програми моделювання спрощеного режиму роботи банкомата

Запитання для перевірки знань

1. Що в програмах називають подією?
2. На які групи поділяються події у мові Python?
3. Як позначається подія клацання лівою кнопкою миші?
4. Які дії можна виконати в програмі для опрацювання події?
5. Яке призначення має метод grid()?
6. Для чого застосовується метод set()?

Завдання для самостійного виконання

1. Зріст учня Сашка становить 178 см, а зріст учниці Михайлини — 166 см. Розробіть програму, яка видає зріст дитини за її ім'ям.
2. Розробіть програму, за допомогою якої за назвою одного з чотирьох лікарських препаратів видається його ціна.
3. У продовольчий магазин доставлені 3 назви товарів. Для кожного з них відома вага товару й загальна його сума. Розробіть програму, за допомогою якої за назвою товару повідомляється вартість його 1 кг.
4. У банкомат може вводитися ПІН-код картки абонента. Для кожного абонента відома загальна сума вкладу й сума останнього надходження. Розробіть програму, за допомогою якої у випадку правильного уведення ПІН-коду повідомляється загальна сума вкладу й сума останнього надходження. Якщо уведено неправильний ПІН-код, видається повідомлення Неправильний ПІН-код. Обмежитися чотирма відомими ПІН-кодами.
5. У протоколі Всеукраїнської олімпіади з інформатики після прізвища учня чи учениці вказано кількість набраних балів і назву навчального закладу. Розробіть програму, за допомогою якої можна за певним прізвищем отримати вказані в протоколі дані (увести дані для чотирьох учнів).

8.4. Меню

Поняття меню в мові програмування Python фактично не відрізняється від аналогічного в операційній системі. Але Python дає змогу користувачу створювати й використовувати меню. Які пункти меню, на вашу думку, доцільно створювати?

Меню є об'єктом бібліотеки tkinter. Для створення меню викликається клас Menu. Меню може містити кілька пунктів, наприклад, File, Edit тощо. Кожен пункт меню може містити список, що розкривається, тобто кілька підпунктів (команд). Наприклад, пункт меню File може містити підпункти Open, Save й ін.

Для розміщення в головному меню пункту меню використовується метод add_cascade. Наприклад, щоб розмістити пункт меню File у головному вікні, слід виконати команду:

<змінна-посилання на головне вікно>.add_cascade(label=»File», menu=<змінна-посилання на основне меню>)

Для додавання підпункту (команди) до пункту меню використовується метод add_command(). Наприклад, щоб розмістити в пункті меню File команду Close, слід виконати команду:

<змінна-посилання на основне меню>.add_command(label=«Close»)

Приклад 1.

Розробимо програму, за допомогою якої у головному вікні створюються пункти меню File, Edit і Help. Пункт File має містити команди New File, Save і Close, пункт Edit — команди Gut і Copy, а пункт Help — команди IDLE Help і Python Docs. Програму створення такого меню зображено на рис. 1.

```
from tkinter import *    # імпортування модуля tkinter
root = Tk()              # створення головного вікна

m = Menu(root)           # створення об'єкта Menu у головному вікні
root.config(menu=m)      # вікно конфігурується з указаним меню

mm1 = Menu(m)            # створення пункту меню File в основному меню
m.add_cascade(label ="File", menu = mm1) # розміщення пункту  File у  меню
mm1.add_command(label="New File       Ctrl+N") # пункт New File у меню File
mm1.add_command(label="Open           Ctrl+O") # пункт Open у меню File
mm1.add_command(label="Close          Alt+F4") # пункт Close у меню File

mm2 = Menu(m)            # створення пункту меню Edit в основному меню
m.add_cascade(label="Edit", menu=mm2)       # розміщення пункту Edit у меню
mm2.add_command(label="Gut          Ctrl+X") # пункт Gut у меню Edit
mm2.add_command(label="Copy         Ctrl+C") # пункт Copy у меню Edit

mm3 = Menu(m)             # створення меню Open у пункті File
m.add_cascade(label="Help", menu= mm3)      # розміщення пункту Help у меню
vmm3.add_command(label="IDLE Help ")        # пункт IDLE Help у меню Help
vmm3.add_command(label="Python Docs   F1") # пункт IDLE Help у меню Help
root.mainloop ()                            # відображення на екрані вікна
```

Рис. 1. Код програми створення меню

Поточний варіант виконання програми наведено на рис. 2:

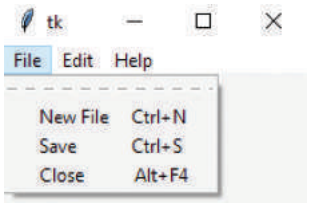


Рис. 2. Виконання програми створення меню

Мова Python дозволяє створювати вкладені меню, тобто підпункт може містити власні підпункти. Для цього створюється ще одне меню, яке за допомогою методу add_cascade зв'язується з відповідними підпунктами (командами) меню.

Зазвичай підпункти меню зв'язуються з відповідною функцією, яка виконує ту чи іншу дію. Зв'язування функцій із командами меню виконується за допомогою параметра command методу add_command. Функція викликається під час клацання лівою кнопкою миші по відповідному підпункту меню.

Приклад 2.

На рис. 3 зображено програму, за допомогою якої виконуються такі дії. Після запуску програми на екрані з'явиться вікно з пунктами меню File і Help. Якщо після цього у відкритому меню File клацнути пункт New, то відбудеться звернення до функції func1. У результаті на екрані відкриється ще одне порожнє вікно.

Якщо клацнути пункт Exit, програма припинить виконання й вікно закриється. Клацання пункту About у відкритому пункті Help призведе до виклику функції func3, у результаті чого в мітці lab з'явиться відповідний текст.

```
from tkinter import *      # імпортування модуля tkinter
root = Tk()                # створення головного вікна
def func1():               # функція викликається при натисненні пункту New
    win = Toplevel(root)   # створення нового вікна
def func2():               # функція викликається при натисненні пункту Exit
    root.destroy()         # завершується програма і закривається вікно
def func3():               # функція викликається при натисненні пункту About
    # властивість text мітки lab набуває текстового значення
    lab = Label(root, text="Метод add_cascade додає новий новий пункт")
    lab.pack()             # розміщення у вікні мітки lab
m = Menu(root)             # створення об'єкту Menu у головному вікні
root.config(menu=m)        # вікно конфігурується з вказаним меню

mm1 = Menu(m)              # створення пункту меню File в основному меню
m.add_cascade(label="File", menu=mm1)   # розміщення пункту  File в  меню
mm1.add_command(label="New" )           # додавання пункту New у меню File
mm1.add_command(label="Exit")           # додавання пункту Exit у меню File

mm2 = Menu(m)              # створення пункту меню Help в основному меню
m.add_cascade(label="Help", menu=mm2)   # розміщення пункту Help в меню
mm2.add_command(label="IDLE Help")      # додавання пункту IDLE Help у меню Help
mm2.add_command(label="About")          # додавання пункту About у меню Help

mm1.add_command(label="New", command=func1)        # зв'язування New з func1
mm1.add_command(label="Exit", command=func2)       # зв'язування Exit з func2
mm2.add_command(label="About", command=func3)      # зв'язування About з func3
root.mainloop()                                    # відображення на екрані вікна
```

Рис. 3. Код програми подій натиснення команд меню

На рис. 4 наведено результат виконання програми в результаті натиснення пункту About у відкритому пункті Help.

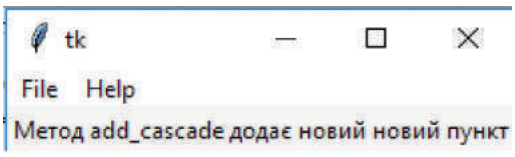


Рис. 4. Виконання програми подій натиснення команд меню

Запитання для перевірки знань

1. Який метод використовується для створення меню?
2. Для чого призначений метод add_command()?
3. Наведіть приклад команди створення пункту меню Edit.
4. Поясніть порядок створення вкладеного меню.
5. Як зв'язуються підпункти меню з відповідними функціями?

Завдання для самостійного виконання

1. Розробіть програму, за допомогою якої у головному вікні створюється меню Help, що містить підпункти Adoub IDLE, IDLE HELP, Python Docs і Turtle Demo.
2. Розробіть програму, за допомогою якої у головному вікні створюється меню Options, що містить підпункти Configure IDLE та Configure Extensions, а також пункт меню Windows.
3. Розробіть програму, за допомогою якої у головному вікні створюється меню з пунктом Edit і його підпунктами Gut і Copy, а також пункт меню Debug із підпунктом Configure IDLE.
4. Розробіть програму, за допомогою якої створюється меню з пунктами File і Help. Пункт File містить підпункти New і Exit. Виконання команди New викликає відкриття ще одного вікна, а команди Exit — закриття вікна й програми.
5. Розробіть програму, за допомогою якої створюється меню з пунктами File і Edit. Пункт Edit містить підпункт Undo, а пункт File — підпункти Exit і Recent Files, який, у свою чергу, містить підпункти doc і txt.

8.5. Діалогові вікна

Пригадайте, для чого застосовувалися діалогові вікна в операційній системі Windows. Назвіть приклади діалогових вікон, із якими вам доводилося працювати.

Існують різні типи діалогових вікон. Один із них — це вікна для відкриття та збереження файлів. Для їх створення слід, окрім модуля tkinter, імпортувати модуль tkinter.filedialog. Вікно для відкриття файлів створюють за допомогою методу askopenfilename(), а для збереження файлів — за допомогою методу asksaveasfilename().

Діалогові вікна призначені для виведення повідомлень користувачу й отримання від нього відповідей та для керування об'єктами.

Найпростіший код створення вікна для відкриття й вікна для збереження файлів зображено на рис. 1.

```
from tkinter import*                  # імпортування модуля tkinter
from tkinter.filedialog import*       # імпортування модуля kinter.filedialog
root = Tk()                           # створення головного вікна

opn = askopenfilename()               # вікно відкриття файлу
sav = asksaveasfilename()             # вікно збереження файлу

root.mainloop()                       # відображення на екрані вікна
```

Рис. 1. Код створення вікон для відкриття та збереження файлів

Об'єкт opn відкриває вікно для відкриття файлів, а об'єкт sav — вікно для збереження файлів. На екрані одразу з'явиться вікно для відкриття файлів, а після його закриття — вікно для збереження файлів. Більше жодних дій за допомогою цієї програми не виконується, її наведено з метою пояснення сутності методів askopenfilename() і asksaveasfilename() та демонстрування зовнішнього вигляду цього типу вікон.

Приклад 1.

Розробимо програму, за допомогою якої у головному вікні створюється багаторядкове текстове поле. У нього можна ввести невеликий текстовий файл, потім відкрити вікно для збереження файлів, указати ім'я та місце збереження файла та зберегти його.

У текстове поле можна також завантажити невеликий текстовий файл, для чого слід відкрити вікно для відкриття файлів. Програму, що реалізує цей сценарій, зображено на рис. 2.

```
from tkinter import *                  # імпортування модуля tkinter
from tkinter.filedialog import *       # імпортування модуля tkinter.filedialog
import fileinput                       # імпортування модуля fileinput

def func1():                           # функція відкриття файлу
    opn = askopenfilename()            # вікно відкриття файлів
    for l in fileinput.input(opn):     # цикл зчитування рядків із файлу
        tex.insert(END,l)              # вставлення рядку у текстове поле

def func2():                           # функція збереження файлу
    sav = asksaveasfilename()          # вікно збереження файлів
    lat = tex.get(1.0,END)             # зчитування вмісту текстового поля
    nk = open(sav,"w")                 # відкриття файлу для запису
    nk.write(lat)                      # запис у файл вмісту текстового поля
    nk.close()                         # закриття вікна

root = Tk()                            # створення головного вікна

m = Menu(root)                         # створення об'єкту Menu у головному вікні
root.config(menu=m)                    # вікно конфігурується з меню

mm1 = Menu(m)                          # створення пункту меню File в основному меню
m.add_cascade(label="File", menu=mm1)  # розміщення пункту File в меню
# створення пункту Open... і зв'язування його з функцією func1
mm1.add_command(label="Open...", command = func1)
# створення пункту Save... і зв'язування його з функцією func2
mm1.add_command(label="Save...", command = func2)

tex = Text(root,width=50,height=20,font="12") # створення текстового поля
tex.pack()                             # розміщення у вікні текстового поля

root.mainloop()                        # відображення на екрані вікна
```

Рис. 2. Код програми завантаження та збереження файлів

Після запуску програми відкриється головне вікно з багаторядковим текстовим полем. Уведемо в це поле, наприклад, такий текст: «Діалогові вікна призначені для виведення повідомлень користувачу, отримання від нього відповідей, а також для управління об'єктами». Після цього відкриємо пункт меню File і виконаємо команду Save…

Відкриється стандартне вікно для збереження файла, вибираємо папку (наприклад Python3.4), уводимо ім'я файла й підтверджуємо Зберегти.

Головне вікно з текстом залишиться на екрані. Закриємо головне вікно й ще раз виконаємо програму. У головному меню відкриємо пункт меню File і виконаємо команду Open…. Відкриється вікно Відкриття файлу, у якому знайдемо ім'я файла Proba1 і відкриємо його. Відкриється головне вікно з уведеним текстом (рис. 3).

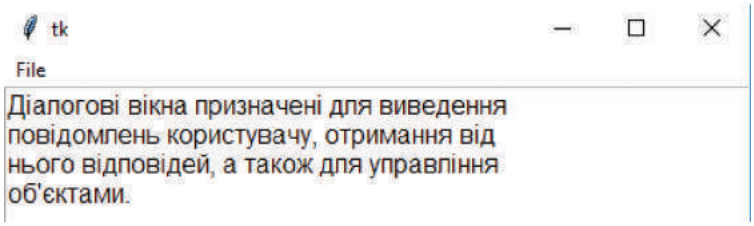


Рис. 3. Головне вікно після відкриття файла

Розглянемо ще один тип діалогових вікон, які призначені для попередження користувача про можливі наслідки подальшого виконання програми й отримання від нього відповіді типу «Так» або «Ні». Такий тип діалогових вікон описано в модулі tkinter.messagebox. Тому на початку програми необхідно імпортувати цей модуль.

Приклад 2.

Приклад програми створення таких вікон зображено на рис. 4.

```
from tkinter import *                # імпортування модуля tkinter
from tkinter.messagebox import *     # імпортування модуля

tkinter.messagebox
def func1():                         # функція закриття головного вікна
    if askyesnc("Exit", "Закрити головне вікно?"):   # якщо натиснуто Так
        root.destroy()               # закриття головного вікна

def func2():                 # функція виведення повідомлення користувачу
    showinfo("Editor", "Виконано успішно") # повідомлення

root = Tk()                  # створення головного вікна
m = Menu(root)               # створення об'єкта Menu у головному вікні
root.config(menu=m)          # вікно конфігурується з меню

mm1 = Menu(m)                # створення пункту меню File в основному меню
m.add_cascade(label="File", menu=mm1)   # розміщення пункту File у меню
# створення пункту Exit і зв'язування його з функцією func1
mm1.add_command(label="Exit", command=func1)

mm2 = Menu(m)                 # створення пункту меню Help в основному меню
m.add_cascade(label="Help", menu=mm2) # розміщення пункту Help у меню
# створення пункту About і зв'язування його з функцією func2
mm2.add_command(label="About", command=func2)
root.mainloop()               # відображення на екрані вікна
```

Рис. 4. Код програми створення вікон із попередженням

Після запуску програми відкриється головне вікно з пунктами меню File і Help. Пункт меню File містить пункт Exit, а пункт Help — пункт About.

Якщо виконати пункт About, з'явиться вікно з назвою Editor, повідомленням Виконано успішно і кнопкою OK. Натиснення на цю кнопку викличе закриття вікна Editor. Головне вікно залишиться на екрані.

Якщо відкрити пункт меню Help і виконати команду Exit, з'явиться вікно Exit (рис. 5) із запитанням Закрити головне вікно? і кнопками Так і Ні.

Натиснення на кнопку Так приведе до закриття головного вікна та вікна Exit, а на кнопку Ні приведе лише до закриття вікна Exit.

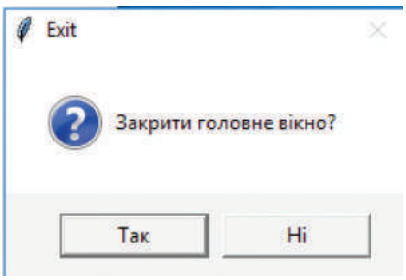


Рис. 5. Діалогове вікно з повідомленням

Запитання для перевірки знань

1. Яке призначення діалогових вікон?
2. За допомогою якого методу створюється вікно відкриття файлів?
3. Для чого призначено метод askcaveasfilename()?
4. Опишіть порядок завантаження файлів у текстове поле.
5. Поясніть порядок створення діалогових вікон із повідомленнями.

Завдання для самостійного виконання

1. Розробіть програму створення вікна лише для відкриття файлів.
2. Розробіть програму, за допомогою якої в багаторядкове текстове поле вводиться текст:

   Географічні координати Києва:

   Широта 50°27"16» пн. ш.

   Довгота 30°31"25» сх. д.

   Висота над рівнем моря 187 м

   Цей текст зберігається у файлі за допомогою вікна для збереження файлів.
3. Створіть програму, за допомогою якої у багаторядкове текстове поле з клавіатури уводиться текст, відкриваються вікна для відкриття та збереження файлів. Використовуючи елементи цього вікна, збережіть файл, а потім відкрийте його.
4. За допомогою текстового редактора створіть файл. Розробіть програму завантаження цього файла в багаторядкове текстове поле.
5. Розробіть програму створення в головному вікні меню з пунктами File і Edit. Меню File містить підпункти Open… і Exit. Після виконання команди Exit має відкритися вікно з назвою Exit і повідомленням Дійсно завершити? Натиснення на кнопку Так повинно закрити програми й вікна.

8.6. Графічні примітиви об'єкта Canvas

Графічні об'єкти будь-якої складності завжди створюються з найпростіших геометричних елементів (примітивів). Який, на вашу думку, необхідно мати перелік примітивів для побудови графічних об'єктів?

Об'єкт Canvas (полотно) — це віджет бібліотеки tkinter, на якому можуть бути розміщені інші віджети: геометричні фігури, малюнки, фотозображення тощо.

Властивості об'єктів, які розміщено на полотні, можна змінювати та переміщувати. Створення полотна в головному вікні виконується за допомогою інструкції:

<змінна> = Canvas(root, параметри)

Параметрами можуть бути розміри полотна (у пікселях), колір заливки й ін. Canvas має систему координат, початок яких розташовується в лівому верхньому куті полотна. Першою завжди вказується координата x, а другою — координата y.

Геометричні фігури, що розміщуються на полотні, називають **графічними примітивами**.

Для кожного примітива існує відповідний метод створення:

Приклад 1.
Інструкцією xol = Canvas (root, width=350, height=450) створюється полотно шириною 350 та висотою 450 пікселів із посиланням на змінну xol.

Метод	Структура
Метод створення ліній — **create_line()**	Має таку загальну структуру: <змінна-посилання на полотно>.create_line(кординати початку і кінця лінії [,інші параметри])
Метод створення прямокутника — **create_rectangle()**	Загальна структура інструкції створення прямокутника така: <змінна посилання на полотно>.create_rectangle(координати верхнього лівого кута, координати правого нижнього кута [,інші параметри])
Метод створення багатокутника — **create_polygon()**	Структура інструкції створення багатокутника така: <змінна посилання на полотно>.create_polygon(координати вершин [,інші параметри])
Метод створення еліпса — **create_oval()**	Загальна структура інструкції створення еліпса така: <змінна посилання на полотно>.create_oval(координати верхнього лівого кута й координати правого нижнього кута неіснуючого прямокутника, у який буде вписано еліпс [,інші параметри])
Метод створення дуги, сектора, сегмента — **create_arc()**	Значення CHORD параметра style у методі означає, що буде створено сегмент, значення ARC — дуга, а за замовчуванням цього параметра створюється сектор. За допомогою параметрів start і extent можна задати кут фігури
Метод створення текстового напису — **create_text()**	Інструкція його створення має таку загальну структуру: <змінна посилання на полотно>.create_text(координати напису, text=«напис»)

У мові Python існують методи, які допомагають у процесі виконання програми змінювати властивості об'єктів, розташованих на полотні, переміщувати їх на полотні й виконувати деякі інші операції. Але для того щоб застосувати ці методи, попередньо об'єктам необхідно надати імена (ідентифікатори).

Присвоїти об'єкту ідентифікатор можна за допомогою такої інструкції:

<ідентифікатор>=<ім'я полотна>.create_<назва об'єкта>(параметри)

Наприклад, присвоїти об'єкту oval, що розміщується на полотні з іменем cnv, ім'я kolo можна так:

kolo = cnv.create_oval([160,10], [120, 60], fill="yellow").

Приклад 2.

На рис. 1 зображено код програми, за допомогою якого розміщуються основні примітиви з назвами.

```
from tkinter import *                                    # імпортування модуля tkinter
root = Tk()                                              # створення головного вікна
cnv = Canvas(root, width=300, height=280)                # створення полотна
cnv.pack()                                               # розміщення у вікні полотна
root.title("Графічні примітиви")                         # заголовок вікна
cnv.create_line([[100,120], [200,140]], fill="black",
width=4, arrow=LAST)                                     # створення лінії
cnv.create_text(170, 120, text="Лінія")                  # напис Лінія
# створення прямокутника із заливкою
cnv.create_rectangle(20,50,100,100, fill="cyan", outline="blue")
cnv.create_text(60, 40, text="Прямокутник")              # напис Прямокутник
# створення прямокутника з написом усередині прямокутника
cnv.create_rectangle(150, 70, 270,100, outline ="magenta")
cnv.create_text(200, 80, text="Прямокутник")             # напис Прямокутник
# створення трикутника з відомими вершинами
cnv.create_polygon([20,120], [200,150], [80,200], fill="orange")
cnv.create_text(100, 160, text = "Трикутник")
cnv.create_oval([160,10], [200,60], fill="gray80")       # створення еліпса
cnv.create_text(230, 40, text="Еліпс")                   # напис Еліпс
cnv.create_arc([200,200], [250,300], start=0, extent=135,
               style = CHORD, fill="green")              # створення хорди
cnv.create_text(200, 240, text = "Хорда")                # напис Хорда
cnv.create_arc([5,220], [120,270], start=0, extent=90,
               outline="red", style = ARC, width=3)      # створення дуги
cnv.create_text(80, 240, text="Дуга")                    # напис Дуга
root.mainloop()                                          # відображення на екрані вікна
```

Рис. 1. Код програми відображення на полотні графічних примітивів

Результат виконання програми зображено на рис. 2.

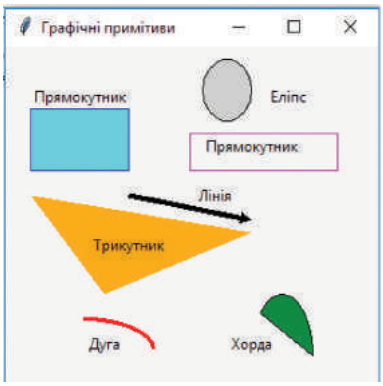


Рис. 2. Графічні примітиви на полотні

Приклад 3.

На рис. 3 зображено код програми, за допомогою якої на полотні відображається коло, квадрат і дві кнопки Button. Після натиснення однієї з кнопок коло кожного разу переміщується вниз на 40 пікселів, а після натиснення другої змінюється товщина рамки й колір заливки прямокутника.

```
from tkinter import *                              # імпортування модуля tkinter
root = Tk()                                        # створення головного вікна
cnv = Canvas(root, width=300, height=280)          # створення полотна
cnv.pack()                                         # розміщення у вікні полотна
root.title("Використання тегів")                   # заголовок вікна

def func1(en):                                      # функція переміщення кола
    cnv.move(kolo, 0, 40)

def func2(en):                                     # функція зміни кольору кв.
    cnv.itemconfig(kvadrat, fill="red", width=5)

kvadrat=cnv.create_rectangle(20, 10, 70, 60, fill="cyan",
                             outline="blue")              # квадрат
kolo=cnv.create_oval([160,10], [210,60], fill="yellow")  # коло

but1 = Button(root)                                # створення кнопки but1 у вікні
but1["text"] = "Переміщення кола"                  # назва кнопки but1
but1.bind("<Button>", func1)                       # зв'язування кнопки but1 з func1

but2 = Button(root)                                # створення кнопки but2 у вікні
but2["text"] = "Зміна кольору квадрата"            # назва кнопки but2
but2.bind("<Button>", func2)                       # зв'язування кнопки but2 з func2

but1.pack()                                        # розміщення кнопки but1 у вікні
but2.pack()                                        # розміщення кнопки but2 у вікні
root.mainloop()                                    # відображення на екрані вікна
```

Рис. 3. Код програми переміщення та зміни властивостей об'єкта

Після запуску програми отримаємо результат, який наведено на рис. 4. Якщо п'ять разів натиснути кнопку Переміщення кола й один раз кнопку Зміна кольору прямокутника, отримаємо такий результат, який подано на рис. 5.

Рис. 4. Розміщення кола та прямокутника одразу після виконання програми

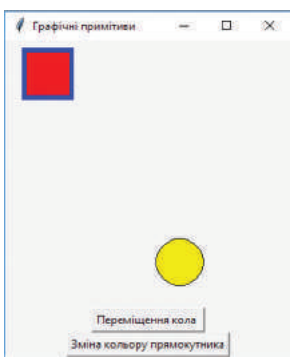


Рис. 5. Розміщення кола та прямокутника після натиснення кнопок

Кожен об'єкт має унікальні ідентифікатори, тому властивості кожного з них можна змінювати незалежно один від одного. Але інколи в процесі виконання програми доцільно одночасно змінити ту саму властивість кількох об'єктів. У мові Python це можна зробити за допомогою тегів, які мають ім'я tag. Для цього той самий тег присвоюється різним об'єктам. Звернення до такого тега дає змогу змінити властивості всіх об'єктів, у яких вказано цей тег.

Розглянемо два методи для роботи з об'єктами: move() та itemconfig.

- Метод move() призначений для переміщення об'єктів по осях X і Y на вказану кількість пікселів. Загальна структура інструкції для застосування цього методу така:

<ідентифікатор полотна>.move(<ідентифікатор об'єкта>, X, Y)

Наприклад, у результаті виконання інструкції cnv.move(kolo, 0, 40) об'єкт з іменем kolo переміститься вниз на 40 пікселів.

- Метод itemconfig() змінює властивості, перелічені в методі. Структура інструкції використання цього методу трохи відрізняється від інструкції застосування методу move(). Наприклад, за допомогою інструкції cnv.itemconfig(kvadrat, fill=«red», width=5) об'єкт з іменем kvadrat набуде кольору red із товщиною рамки 5 пікселів.

Природно, що зміни над об'єктами мають виникати під час зовнішнього впливу, наприклад, після натиснення кнопки миші, клацання кнопки Button тощо. Після виконання таких дій управління повинно передаватися відповідній функції для опрацювання цієї події.

За допомогою методу delete() можна видалити з полотна об'єкт зі вказаним ідентифікатором або тегом. Якщо в методі вказати значення all, із полотна буде видалено всі об'єкти. Наприклад, за допомогою інструкції cnv.delete(all) із полотна з іменем cnv буде видалено всі об'єкти.

Запитання для перевірки знань

1. Для чого призначено об'єкт Canvas?
2. За допомогою якої інструкції створюється полотно в головному вікні?
3. Які параметри може мати об'єкт Canvas?
4. Для чого призначений метод delete()?
5. Яке призначення методу create_rectangle()?
6. Наведіть інструкцію створення еліпса.
7. Із якою метою об'єктам на полотні присвоюються ідентифікатори?
8. Що називають тегами; яке їх призначення?

Завдання для самостійного виконання

1. Розробіть програму, за допомогою якої на полотні створюється трикутник з описаним навколо нього колом.
2. Розробіть програму, за допомогою якої на полотні створюється піраміда, основою котрої є квадрат.
3. Розробіть програму, за допомогою якої на полотні створюється футбольне поле з усіма лініями на ньому й воротами.
4. Розробіть програму, за допомогою якої на полотні створюється баскетбольний майданчик з усіма його елементами.
5. Розробіть програму, за допомогою якої на полотні розміщується прямокутник, правильний трикутник і дві кнопки. Після натиснення першої кнопки переміщується прямокутник, а після натиснення другої — обидві фігури.
6. Розробіть програму, за допомогою якої на футбольному полі після натиснення кнопки м'яч влітає у ворота.

Виконайте тестове завдання до розділу 2 з автоматичною перевіркою результату на сайті interactive.ranok.com.ua

Розділ 2. СУЧАСНІ ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ

2.1. Сучасні інформаційні технології та системи. Людина в інформаційному суспільстві

Уявіть, ви живете півстоліття тому. Вам необхідно: повідомити рідних, що сьогодні ви затримуєтеся в школі, купити в магазині цукерки для бабусі, на завтра написати реферат про Алана Тюрінга, на вихідні здійснити незаплановану подорож у сусіднє місто. Які ваші дії?

Сьогодні все наше повсякденне життя нерозривно пов'язане з використанням інформаційних технологій (ІТ).

Інформаційні технології — це сукупність процесів, що використовує засоби та методи пошуку, збирання, накопичення, зберігання, опрацювання та передавання первинної інформації для отримання інформації про стан об'єкта, процесу або явища за допомогою засобів обчислювальної та комунікаційної техніки.

Стрімкий розвиток ІТ та поява соціальних мереж і сервісів обміну інформацією зробили революцію у сфері міжособистісних комунікацій, знищили просторові й часові межі в контактах між людьми, дали змогу формувати й підтримувати нові економічні моделі. У 2011 році журнал Time назвав економічну модель «спільного споживання» (sharing economy — економіка спільної участі) у числі тих, що можуть змінити світ споживання. Головна ідея цієї моделі — об'єднати масовий попит і розрізнену пропозицію, використовуючи принципи бартеру й оренди. Концепція проявляється як у різноманітних галузях цифрового бізнесу, так і в людській діяльності загалом.

Сьогодні на моделі «спільного споживання» побудовано всесвітньо відомі глобальні онлайн-сервіси (рис. 1):

- Uber (мобільний застосунок для пошуку, виклику та оплати таксі або приватних водіїв; існує з 2009 року);
- BlaBlaCar (платформа для спільних поїздок автомобілем; запущена в 2004 році);
- Airbnb (онлайн-сервіс із розміщення, пошуку та короткострокової оренди житла по всьому світі; започаткований у 2008 році).

Існують також TaskRabbit, Sorted (дрібні побутові послуги), Shareyourmeal (можливість купити/продати надлишок домашньої їжі).

Ми легко спілкуємося з друзями, обмінюємося «Сторіс» в Інстаграмі, «лайкаємо» сторінки незнайомих нам людей у Фейсбуці, шукаємо в Google значення якогось терміна, мандруємо, не виходячи з дому, кращими музеями світу, замовляємо через Інтернет квитки на потяг або концерт.

Рис. 1. Логотипи онлайн-сервісів, які використовують модель «спільного споживання»

Людину або будь-яку істоту можна вважати біологічною інформаційною системою (ІС), суспільство — соціальною ІС, а комп'ютер — технічною.

Карен Спарк Джон — британська дослідниця, яка розробила технології пошуку, які дозволили користувачам працювати з комп'ютерами, використовуючи слова замість рівнянь та кодів.

Вплив ІТ стає дійсно всеосяжним: вони в той чи інший спосіб змінюють традиційні форми людської діяльності: ведення бізнесу, виробництво, освіту, культурне життя, та приводять до появи нових. Комп'ютерна грамотність стала невід'ємною частиною повсякденного життя, обов'язковою умовою працевлаштування й успішної кар'єри.

Упровадження ІТ впливає на характер праці на виробництві, знижує число працюючих у цій галузі, змінює саму організацію й технологію виробництва. Зазнає змін сама процедура обговорення та прийняття виробничих рішень.

Термін «інформаційні технології» нерозривно пов'язаний із терміном «інформаційна система» (ІС).

Інформаційна система — сукупність організаційних і технічних засобів для збереження та опрацювання інформації з метою забезпечення інформаційних потреб користувачів.

Усе частіше технічні ІС розробляються для задоволення інформаційних потреб у межах конкретної галузі людської діяльності. Комп'ютер як технічна ІС є програмно-апаратним комплексом для зберігання та опрацювання даних і забезпечення зручного інтерфейсу.

Згадаємо основні складові та призначення сучасної ІС та її призначення (рис. 2).

Рис. 2. Типовий склад технічної ІС

Розробляють ІС для задоволення інформаційних потреб у межах певної галузі людської діяльності. Згадаємо, як часто ми користуємося послугами геосервісів — геоінформаційними системами на кшталт Google Maps. ІС стають важливою та невід'ємною складовою сучасного суспільства. Розширення їх можливостей сприяє підвищенню рівня освіти, науково-технічному та культурному розвитку, і відповідно зростанню ролі кваліфікації, професіоналізму і здатності до творчості.

Людство пережило кілька етапів суспільного розвитку (рис. 3). Важливим є й постіндустріальний етап, коли понад 50 % працездатного населення зайняте у сфері послуг. Поступово людство переходить до інформаційного етапу — коли понад 50 % населення зайнято у сфері інформаційних послуг.

Немає випускника чи випускниці, які б під час вступу до вишів не використовували ІС «Конкурс». Система надає вичерпну інформацію про всі вступні кампанії, починаючи з 2008 року.

Аграрний етап

Основа — розвиток сільськогосподарських відносин, пов'язаних із системою землекористування

Індустріальний етап

Основа — розвиток промисловості та її технічного забезпечення

Інформаційний етап

Основа — розвиток технологій отримання, зберігання, накопичення, опрацювання, обміну та продажу даних

Рис. 3. Основні етапи розвитку суспільства

Коли ми говоримо про інформаційне суспільство, то насамперед виділяємо такі його характеристики:

- збільшення ролі інформації та інформаційних технологій у житті суспільства;
- створення та розвиток ринку інформації та знань як чинників виробництва, що доповнюють уже наявні ринки природних ресурсів, праці й капіталу, перехід інформаційних ресурсів суспільства до дійсних ресурсів соціально-економічного розвитку; підвищення частки інформаційних комунікацій, продуктів і послуг у валовому внутрішньому продукті;
- зростання інформатизації суспільства завдяки розвитку телефонії, радіо, телебачення, Інтернету, традиційних і електронних засобів масової інформації;
- створення глобального інформаційного простору, що забезпечує ефективну інформаційну взаємодію людей, їх доступ до світових інформаційних ресурсів та ін.

Формування інформаційного суспільства є результатом чотирьох інформаційних революцій, які відбулися в історії розвитку людської цивілізації. Кожна з них кардинально змінювала не лише способи опрацювання інформації, а й спосіб виробництва, стиль життя, систему цінностей.

Інформаційна революція (англ. Information Revolution) — це поняття, що відображає революційний вплив інформаційних технологій на всі сфери життя суспільства, особливо в останній чверті XX ст.

Джуді Маллой — американська письменниця та програміст-самоучка, яка винайшла власну систему баз даних для своїх романів. У 1986 році вона написала та запрограмувала новаторський гіпертекстовий роман «Дядько Роджер» («Uncle Roger»). Це перший онлайн-проект гіперлітератури з посиланнями, які змінюють сюжет залежно від уподобань читача.

Інформаційна революція інтегрує ефекти попередніх революційних винаходів в інформаційній сфері (книгодрукування, телефонія, радіозв'язок, комп'ютер). Це створює технологічну основу для подолання відстаней під час передавання інформації, що сприяє об'єднанню інтелектуальних здібностей людства:

Інформаційна революція	Опис
Перша	Пов'язана з винайденням писемності, що сприяло якісному стрибку в розвитку людства
Друга (сер. XVI ст.)	Пов'язана з появою книгодрукування; суттєво вплинула на розвиток індустріального суспільства
Третя (кін. XIX ст.)	Пов'язана з винайденням електрики, а потім телеграфу, телефону і радіо
Четверта (70-ті рр. XX ст.)	Пов'язана з винайденням мікропроцесорної техніки, персонального комп'ютера, а пізніше — Інтернету

Людина в інформаційному суспільстві стикається з величезним обсягом інформації, який потребує опрацювання. Таким чином, її мозок працює в інтенсивному режимі багатозадачності. Людина має вдосконалювати критичне мислення для того, щоб відрізняти достовірну інформацію від фейкової.

Запитання для перевірки знань

1. Що таке інформаційні технології?
2. Наведіть приклади використання інформаційних технологій у повсякденному житті.
3. Яке суспільство зазвичай називають інформаційним?
4. Назвіть основні етапи суспільного розвитку, яке пережило людство.
5. Яке суспільство називають постіндустріальним?
6. Назвіть характерні риси інформаційного суспільства.

2.2. Навчання в Інтернеті

Як би ви охарактеризували модель, за якою навчаєтеся?

На відміну від людини індустріального суспільства, яка має розвинену довгострокову пам'ять, людині інформаційного суспільства не потрібно тримати в пам'яті інформацію — достатньо знати, де її знайти.

За традиційною моделлю навчання учні для отримання середньої освіти мають щодня приходити до школи, слухати на уроках учителя, виконувати домашні завдання, готуватися до іспитів. Завдяки Інтернету створюються принципово нові умови для організації дистанційного навчання.

Нині здатність до самоосвіти, спроможність швидко опановувати нові професії стає однією з основних умов успішності людини. Враховуючи це, все більше провідних університетів світу викладають свої аудіо- та відеолекції в Інтернет у вільний доступ.

На сайті Гарвардського університету доступні безкоштовні відеолекції, понад 50 навчальних програм різної тематики — від мистецтва, соціальних наук і літератури до сучасної історії, точних наук і математики.

На сайті університету Берклі доступні безкоштовні аудіо- і відеокурси з фізики, біології, хімії, психології, інформатики,

іноземних мов, економіки, географії, історії, філософії, соціології та цілої низки інших дисциплін.

Зростає роль освітніх платформ, які пропонують масові відкриті безкоштовні онлайн-курси. Ці курси вже тепер створюють можливість мільйонам людей здобувати знання, перебуваючи в будь-якому куточку світу.

Усе більшою популярністю користуються відкриті онлайн-курси провідних університетів світу, розміщені на платформах Coursera, Udacity, EdX та ін. (див. рисунок). На Всесвітньому економічному форумі в Давосі (2018 рік) зазначалося, що онлайн-освіта змінює світ.

Coursera — проект у галузі масової онлайн-освіти, який засновано у квітні 2012 року професорами інформатики Стенфордського університету Ендрю Нґ та Дафною Коллер.

Наразі Coursera має 100 університетів-партнерів, в тому числі 33 університети США та близько 5 млн користувачів. Coursera пропонує близько 500 курсів за освітніми категоріями, що охоплюють гуманітарні науки, біологію, медицину, соціальні науки, математику, бізнес, економіку та фінанси, комп'ютерні науки та багато інших.

Курси на Coursera включають відеолекції з субтитрами, конспекти лекцій, домашні завдання, тести, підсумкові іспити; всі завдання мають бути виконані до певної дати, існує «форум» для обміну думками. За умови успішного виконання проміжних завдань і заключного іспиту слухач отримує сертифікат.

Udacity — комерційна освітня організація, заснована професором комп'ютерних наук Стенфордського університету Себастьяном Траном.

Udacity пропонує масові відкриті онлайн-курси, орієнтуючись насамперед на «професійні курси для професіоналів», присвячені переважно комп'ютерним наукам, фізиці та математиці. Головна ідея проекту — сприяння подальшому працевлаштуванню слухачів.

В Udacity за допомогою відеозв'язку (Office hours) студенти можуть протягом тижня поставити питання авторам курсів. Фінальна оцінка складається з оцінки за іспити й ураховує результати виконаних домашніх завдань. Сертифікація після проходження курсу дає змогу оцінити кожному свій загальний рівень.

Уже зараз Udacity і Coursera почали надавати слухачам університетські кредити за проходження окремих курсів, які певні університети вже зараховують як складову навчальної програми.

eDX — проект, який створено спільно двома вишами — Гарвардським університетом і Массачусетським технологічним інститутом.

eDX пропонує понад 250 безкоштовних онлайн-курсів від найвідоміших університетів та інститутів світу, переважно з комп'ютерних наук та електроніки. Після проходження курсів у слухачів також є можливість отримати сертифікати eDX.

Логотипи найпопулярніших освітніх платформ

Основні переваги дистанційного навчання

- Оперативне отримання будь-якої інформації з сайтів навчальних закладів у будь-якій точці земної кулі
- Зберігання навчальних матеріалів на ґаджетах протягом необхідного часу та можливість їх змінювати й редагувати
- Реалізація інтерактивності за допомогою оперативного зворотного зв'язку з учасниками процесу навчання
- Можливість порівняти інформацію, отриману з різних джерел

Курси на Udacity і Coursera містять цілий комплекс взаємопов'язаних складових. Тривалість курсів регламентовано в часі (6–9 тижнів). Щотижня зворотний зв'язок підтримується шляхом перегляду блока відеоматеріалів і заповнення бланків із запитаннями. Після цього — домашні завдання до кінця тижня.

Усе більшої популярності в Україні набувають відкриті онлайн-курси на україномовних платформах Prometheus і EdEra.

Prometheus — перший україномовний громадський проект масових відкритих онлайн-курсів. Головною метою проекту є безкоштовне надання онлайн-доступу до курсів університетського рівня всім бажаючим, а також надання можливості публікувати та розповсюджувати такі курси провідним викладачам, університетам та компаніям.

На платформі розміщено цикл курсів для всіх, хто хоче підготуватися до успішного складання ЗНО та ДПА з української мови та літератури, історії України й математики.

EdEra — другий за популярністю україномовний онлайн-ресурс, українська студія онлайн-освіти. Наразі команда створила понад 1000 освітніх роликів, запустила близько 20 відкритих онлайн-курсів та інтерактивних підручників.

EdEra ставить на меті зробити освіту якісною та доступною, орієнтуючись насамперед на базову середню освіту, створити онлайн-курси повного циклу — від лекцій до книжок.

Сервіс **EdEraBooks** — це соціальний проект інтерактивної освітньої літератури зі вбудованими відео, автоматизованими тестами й можливістю коментувати кожну сторінку. Проект незамінний для тих, хто хоче добре підготуватися до складання ЗНО.

Запитання для перевірки знань

1. У чому полягає сутність навчання в Інтернеті?
2. Наведіть приклади найпопулярніших освітніх онлайн-платформ.
3. Що таке онлайн-курс повного циклу? Наведіть приклади таких курсів.
4. Назвіть переваги дистанційного навчання.
5. Що, на вашу думку, стримує розвиток онлайн-навчання?
6. Які, на вашу думку, курси є найбільш затребуваними?

2.3. Професії майбутнього: аналіз тенденцій на ринку праці

Поміркуйте, які професії сьогодні користуються найбільшим попитом.

На кожному етапі розвитку людства завжди є певна група найбільш затребуваних професій. Потреби на ринку праці залежать від багатьох чинників.

Вирішальний вплив на формування тенденцій на ринку праці мають процеси, що безпосередньо відбуваються в соціально-економічному житті суспільства, у тому числі глобальних на світовому рівні.

Стрімкий розвиток ІТ створив передумови для «переміщення» діяльності бізнесу в глобальну мережу. Так, просування товарів і послуг, фінансові операції та пошук персоналу здійснюється тепер у Всесвітній павутині. Тому все більше бізнес-спеціалістів опановують інструменти, раніше притаманні веб-девелоперам та ІТ-спеціалістам.

Стрімко зростає потреба в спеціалістах із кібербезпеки.

Розвиток смарт-технологій у подальшому сприятиме появі нових професій: проектувальник інфраструктури «розумного дому», будівельник «розумних» доріг, фахівець у галузі альтернативної енергії.

Украй важливою сферою для людей завжди була медицина, і в майбутньому її значення зростатиме. Дослідження ДНК відкрили нову еру превентивної медицини, що має на меті виявити можливі захворювання і запобігти їм на ранній стадії. Активно використовуються біотехнології, що допомагають у розробці ліків і створенні тканин і органів для пересаджування. З'являться нові професії: IT-медик, оператор медичних роботів, молекулярний дієтолог та ін.

Розвиток технологій змінює уявлення людей про способи отримання знань і змушує переосмислити звичний підхід до навчального процесу. Це означає, що в майбутньому будуть досить затребуваними фахівці в галузі освіти. Сьогодні вже використовуються інструменти навчання із застосуванням IT: онлайн-курси, симулятори, тренажери, ігрові онлайн-світи. Це дає змогу учням розвивати вміння працювати з інформацією. Уже існують такі професії, як тьютор, координатор освітньої онлайн-платформи, ментор стартапів та ін.

Якими б різноманітними не були майбутні професії, завжди будуть залишатися певні вимоги до фахівців.

- Крос-функціональність — найбільше цінуються фахівці, які працюють «на стику» професій. Багато компаній прагнуть сформувати команди професіоналів, що володіють знаннями відразу в кількох галузях.
- Мобільність — експерти стверджують, що нині часта зміна місця роботи й навіть сфери діяльності є нормою. Американські дослідники припускають, що незабаром людина до 38 років зможе освоїти до 10–14 професій.
- Універсальність — зростатиме попит на фахівців, які володіють кількома іноземними мовами, знають основи менеджменту, маркетингу та фінансів та вміють працювати з Big Data — технологією опрацювання великих обсягів даних.
- Уміння працювати в команді — важливо узгоджувати свої дії під час колективної роботи, бути здатним не лише відстоювати власну позицію, а й поважати позицію інших.

Уміння застосовувати у своїй діяльності сучасні IT стає однією з основних складових професійної підготовки будь-якого фахівця. Перелік ключових компетентностей, визначених ЄС у 2018 р. наведено на рис. 1.

Невід'ємними характеристиками стають упевнені навички користування офісними пакетами (Microsoft Office, LibreOffice, OpenOffice), уміння організувати свій час (так званий тайм-менеджмент) за допомогою віртуальних цифрових помічників (рис. 2).

Користується попитом робототехніка: автоматичні пристрої перевершують у точності звичайних хірургів, а кіберпротези можуть не просто компенсувати фізичні вади, але й відкрити перед людиною нові можливості.

Ключові компетентності

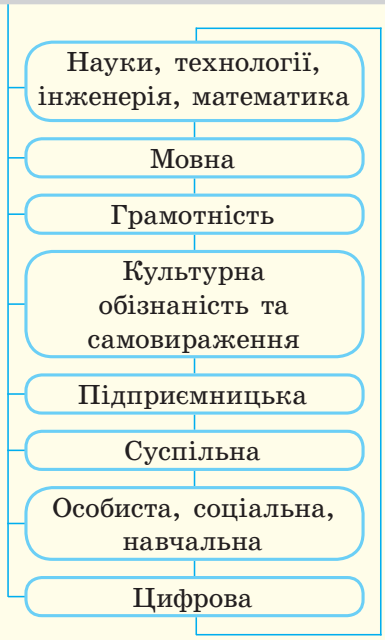


Рис. 1. Ключові компетентності для навчання впродовж життя

Рис. 2. Логотипи віртуальних цифрових помічників

Віртуальні цифрові помічники можуть розпізнавати природну людську мову в усній формі, і завдяки цьому вмінню та володінню елементами штучного інтелекту допомагають людям з інвалідністю: особам із обмеженими фізичними можливостями та вадами зору.

Віртуальний цифровий помічник (від англ. Virtual Digital Assistant, скорочено VDA) — додаток для смартфонів, який фактично виконує роль особистого секретаря користувача. Такий помічник «розв'язує» завдання планування, організації і виконання повсякденних справ, контекстного пошуку інформації для потреб конкретної людини тощо.

Віртуальний цифровий помічник може створювати нагадування, полегшити пошук і онлайн-бронювання квитків, замовлення таксі. У процесі виконання завдань він здатний до самонавчання, може аналізувати поведінку й інтереси користувача. Наразі найвідомішими віртуальними цифровими помічниками є Siri, Google Assistant, Microsoft Cortana.

Siri — персональний помічник, адаптований під iOS. Цей додаток спілкується природною мовою, відповідає на питання та дає рекомендації. Siri пристосовується до кожного користувача, вивчаючи його особливості протягом тривалого часу.

Google Assistant — розумний персональний асистент, розроблений компанією Google і представлений 18 травня 2016 року на конференції Google I/O. Додаткова функція програми — збирання персональних даних користувачів у корпоративній системі машинного навчання

Microsoft Cortana — персональна помічниця, покликана передбачати потреби користувача. За бажанням їй можна дати доступ до особистих даних користувача, таких як електронна пошта, адресна книга, історія пошуків у мережі, — усі ці дані вона буде використовувати для передбачення потреб власника.

Як відомо, будь-яка нова справа потребує чіткого планування: із яких етапів вона складається, скільки часу відводиться, як розподілити обов'язки, якщо справа командна. Можна розписати ланцюжок дій на аркуші паперу, але краще скористатися спеціальними сервісами — інструментами для індивідуального (або колективного) планування.

Існують різні види онлайн-інструментів (рис. 3):

- органайзер — інструмент організації інформації про особисті контакти та події, засіб управління часом;
- тайм-трекінг — інструмент управління часом, відстеження часу на різні завдання (http://www.yast.com/, http://www.rememberthemilk.com/);
- планувальник завдань — ToDo-лист, інструмент, що дає змогу організувати список завдань (http://todo.ly/).

Окрім згаданих, існує ще низка подібних органайзерів — онлайн- і офлайн-версії, мобільні та веб-додатки, платні й безкоштовні, орієнтовані на індивідуальну та колективну роботу, навчальну та бізнесову діяльність. Переважна їх більшість має інтуїтивно зрозумілий інтерфейс (приклад).

Найчастіше проектом керує одна людина, якщо проект досить простий. Але зазвичай керування проектами (Project management) передбачає колективну роботу.

Рис. 3. Популярні онлайн-інструменти

Приклад.

Розглянемо роботу органайзерів на прикладі веб-застосунка Google Календар (https://www.google.com/calendar/) (рис. 4–6).

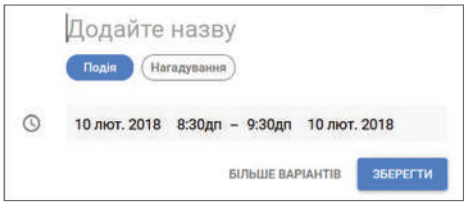


Рис. 4. Вибір події в Google Календар

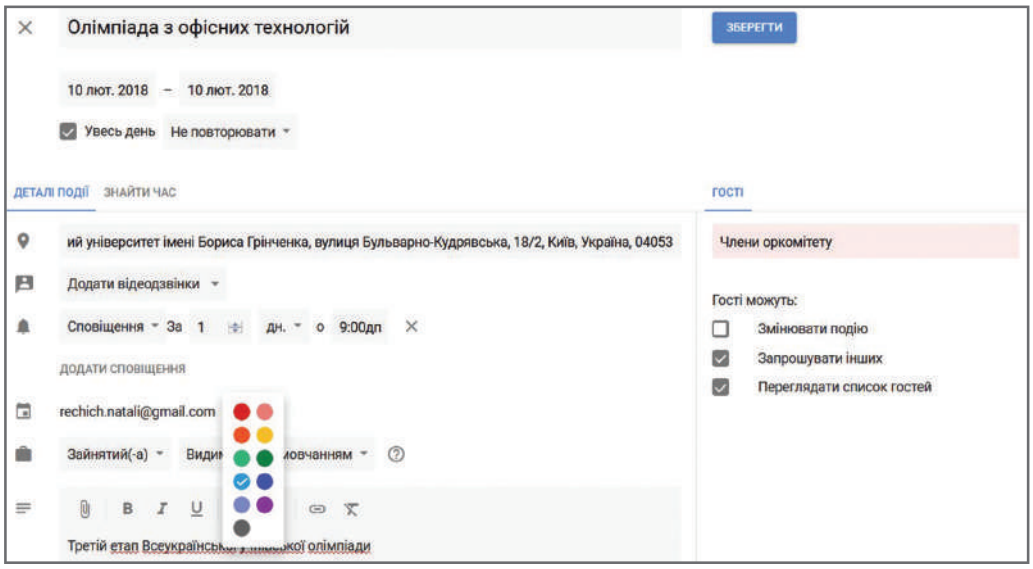


Рис. 5. Налаштування параметрів події

Рис. 6. Відображення події в Google Календар

Для організації мережевих проектів із великою кількістю завдань та багатьма особами, залученими до участі в них, буває необхідно використовувати спеціальні веб-інструменти для керування проектами. Такі сервіси створюють можливість зберігати інформацію про проекти в Інтернеті, спільно керувати проектами, розподіляти рівні доступу та відповідальність, планувати завдання і призначати тих, хто їх виконуватиме, із подальшим відстеженням ходу виконання.

Casual — наочний інструмент від українських розробників для керування проектами (http://www.casual.pm). Цей стартап реалізує досить цікавий підхід: він дає змогу намалювати

списки завдань за допомогою візуальних схем. Програма орієнтована на фахівців, яким важливо відстежувати виконання другорядних завдань.

Щоб спланувати етапи складних дослідницьких завдань, часто використовують нескінченні віртуальні дошки. На них можна працювати з будь-яким візуальним контентом (прикріплювати картинки, малювати схеми й графіки, створювати колажі й ін.) індивідуально або з командою.

Наведемо приклад таких сервісів (рис. 7).

- Twiddla (http://www.twiddla.com) — онлайн-дошка для сучасного класу, інструмент для співпраці в реальному часі. Простий у користуванні: немає плагінів або завантажень, розширених розкладів. Єдина вимога — комп'ютер із доступом до Інтернету та нескладний браузер.
- SpiderScribe (http://www.spiderscribe.net/) — онлайн-інструмент для створення інтелект-карт та мозкового штурму. Він допомагає організувати ідеї, підключивши нотатки, файли, події календаря тощо в картках вільної форми. Надає можливість співпрацювати та обмінюватися інформацією онлайн.
- Lino (http://linoit.com) та RealtimeBoard (http://realtimeboard.com) — віртуальні дошки для командної роботи.

Більшість помічників можуть програвати музику, читати новини з кількох джерел, надавати інформацію про погоду, спорт, кіно, маршрути, місцеві компанії. Можуть підказати, що подивитися по телевізору, взаємодіяти з пристроями «розумного дому», кількість яких із року в рік зростає.

Рис. 7. Логотипи онлайн-сервісів для віртуальних дошок

Запитання для перевірки знань

1. Наведіть приклади нових професій у медицині й освіті, пов'язаних із використанням ІТ.
2. Які основні вимоги висуваються в сучасному світі до майбутніх фахівців незалежно від обраної професії?
3. Які інструменти використовуються під час планування колективної роботи?
4. Що означає VDA? Які найвідоміші VDA ви можете назвати?
5. Знайдіть відомості в Інтернеті та проведіть порівняльний аналіз найпопулярніших таймтрекерів.
6. Чи користуєтеся ви онлайн-органайзерами? Якщо так, то якими?

2.4. Системи електронного врядування

Пригадайте, що таке електронний документообіг. Які переваги його використання?

Наразі ми говоримо про інформаційне суспільство як характерну ознаку XXI ст. Саме в такому суспільстві активно розвиваються інформаційні й комунікаційні технології, створюються умови для ефективного використання знань для розв'язування актуальних питань сьогодення. Одне з них — недостатньо ефективна діяльність системи органів державного управління й органів місцевого самоврядування, потреба в її якісному оновленні.

Упровадження системи електронного врядування покликане вдосконалити взаємодію влади та громадян та сприяти соціальним інноваціям е країні.

Електронне врядування — форма організації державного управління, з використанням інформаційно-телекомунікаційних технологій для формування нового типу держави, орієнтованої на задоволення потреб громадян, яка сприяє підвищенню ефективності, відкритості та прозорості діяльності органів державної влади та органів місцевого самоврядування.

Електронне врядування визначає спосіб взаємодії громадян та юридичних осіб із органами державної влади засобами інформаційних технологій із метою підвищення ефективності надання державних послуг. Це єдина міжвідомча автоматизована інформаційна взаємодія органів державної влади та місцевого самоврядування з громадянами та підприємствами.

Основними завданнями електронного врядування є:

- забезпечення прав громадян на доступ до всіх видів відкритої державної інформації;
- залучення громадян до участі в державних справах;
- удосконалення технології державного управління;
- подолання інформаційної нерівності.

Функціонування систем електронного врядування здійснюється на основі певних принципів. Це єдиноразова реєстрація документів, паралельне виконання різних операцій із метою скорочення руху документів, безперервність руху документів, розвинута система контролю та звітності органів державної влади.

Електронне врядування не замінює традиційний уряд і не є його аналогом.

У вересні 2017 року уряд України ухвалив головний е-документ країни — Концепцію розвитку електронного врядування до 2020 року, який визначає головні цілі, пріоритетні напрямки та заходи цифрової трансформації.

Електронне врядування створюється для реалізації таких основних функцій:

- забезпечення доступності й підвищення якості надання громадянам державних послуг, спрощення цих процедур і скорочення бюджетних витрат;
- надійний контроль за діяльністю державних органів виконавчої влади і підвищення якості управлінських та адміністративних процесів;
- відкритість інформації про діяльність органів влади, участь громадян і громадських організацій у підготовці й експертизі відповідних рішень;
- реєстрація суб'єктів підприємницької діяльності, прав власності, видача дозволів на проведення мітингів, займання окремими видами підприємницької діяльності, сертифікатів, акредитацій, атестацій тощо;
- призначення пенсій, субсидій та інших прав осіб.

За останні два роки Україна піднялася на 25 позицій у світовому рейтингу е-урядування, на 45 позицій та на 30 позицій — у рейтингах відкритих даних.

Електронне врядування створює умови для відкритого та прозорого державного управління. Відбувається активна взаємодія органів місцевого самоврядування і державної влади між собою, а також із кожним е-громадянином і суспільством загалом (рис. 1, с. 146).

Електронне врядування в Україні перебуває на стадії становлення та розвитку. Подальший розвиток можливий за умови застосування перспективних технологій, наприклад, блокчейн, яка дійсно може якісно оптимізувати безліч процесів.

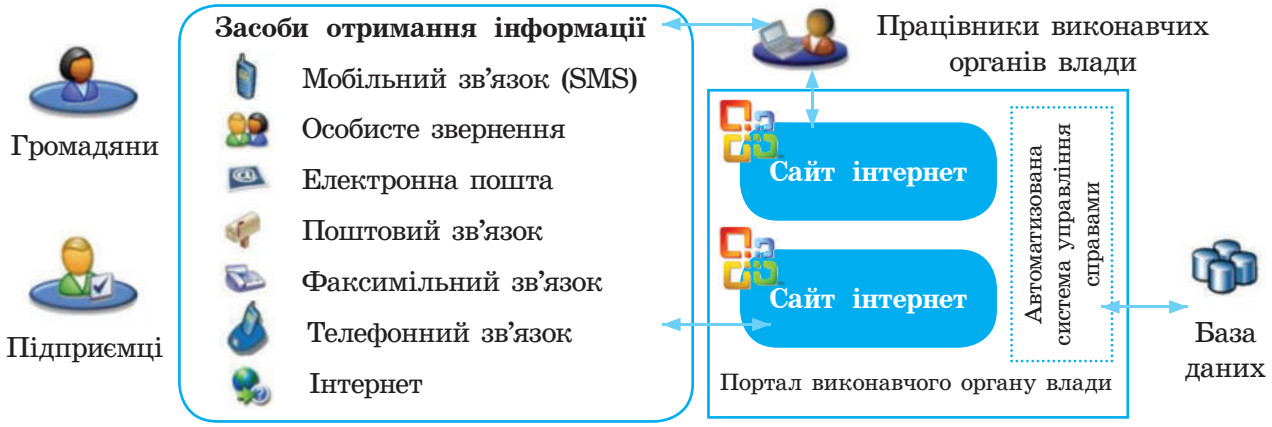


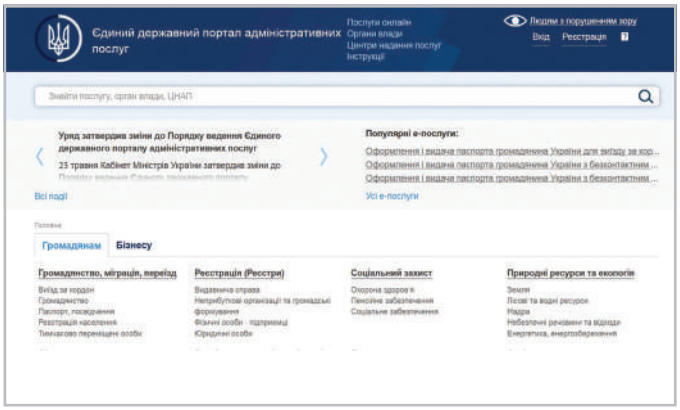
Рис. 1. Схема роботи е-уряду

Україна ввійшла в 14 світових блокчейн-лідерів. На технологію блойкчейн в Україні переведено Державну систему електронних торгів арештованим майном (СЕТАМ), у процесі переведення — Реєстр майнових прав та Державний земельний кадастр України.

Блокчейн — це розподілена база даних, у якій зберігається інформація про кожну транзакцію, зроблену в системі. Дані зберігаються у вигляді ланцюжка блоків (звідси й назва — blockchain) із записами про транзакції. Їх неможливо підробити, оскільки кожен новий запис здійснює підтвердження вже наявних ланцюжків. Інформація про записи в системі зберігається в усіх її учасників і автоматично оновлюється під час унесення змін. Переваги блокчейна — розгалуженість і прозорість транзакцій.

Один із головних блокчейн-експертів у світі Дон Тапскотт (Канада) на форумі в Давосі в 2018 році презентував світову блокчейн-карту з 14 країнами — лідерами з упровадження цієї технології. Україна — в їх числі.

Згідно з кращими практиками ЄС в Україні найближчим часом має запрацювати сервіс мобільної ідентифікації громадян MobileID та автоматичний обмін даними між реєстрами. На рис. 2 наведено приклади Всеукраїнських державних порталів надання е-послуг.



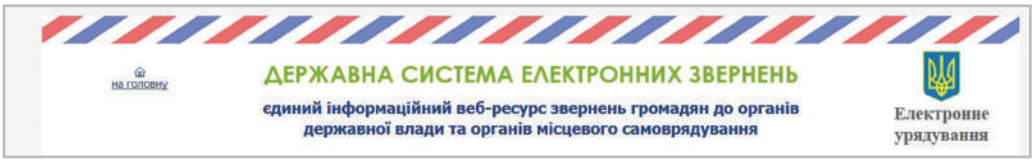


Рис. 2. Всеукраїнські державні портали адміністративних е-послуг

Нині існують різні види електронного врядування (е-урядування) (рис. 3).

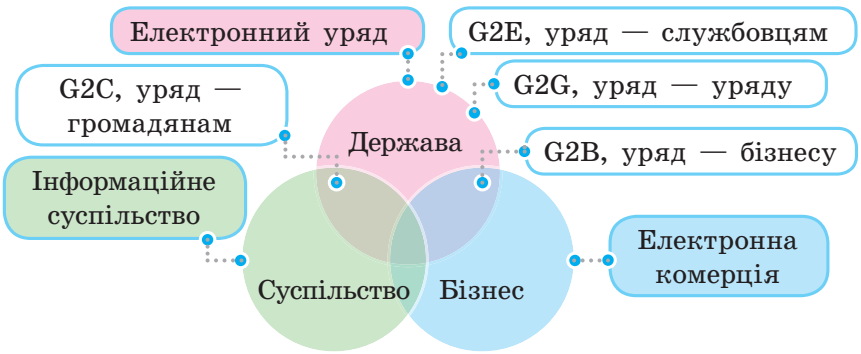


Рис. 3. Простір електронної взаємодії

В основі ідеї е-урядування лежить теорія нового державного менеджменту (англ. *new public management*), головними ідеологами якої були М. Тетчер і Р. Рейган. Найбільшого практичного відображення ідея набула в адміністративних реформах 1980–1990-х років у США, Великій Британії, Канаді, Австралії, Новій Зеландії.

- Уряд — громадянам (G2C, government to citizens). Цей вид скорочує для громадян час очікування в чергах за довідками та формами, їх фінансові витрати, а також бюджетні витрати на ці процедури.
- Уряд — службовцям (G2E, government to employees). Автоматизуються процеси співпраці урядової системи з чиновниками, службовцями та консультантами на місцях.
- Уряд — бізнесу (government to business). Автоматизуються процеси податкових виплат, проведення електронних тендерів на постачання продукції, послуг, виконання проектних робіт тощо.
- Уряд — уряду (government to government). Автоматизуються процеси документообігу між відомствами, діловодства, координується робота територіальних підрозділів.

Сучасні державні сайти містять сторінку е-урядування, які надають можливість налаштування для людей із вадами зору (рис. 4, 5).

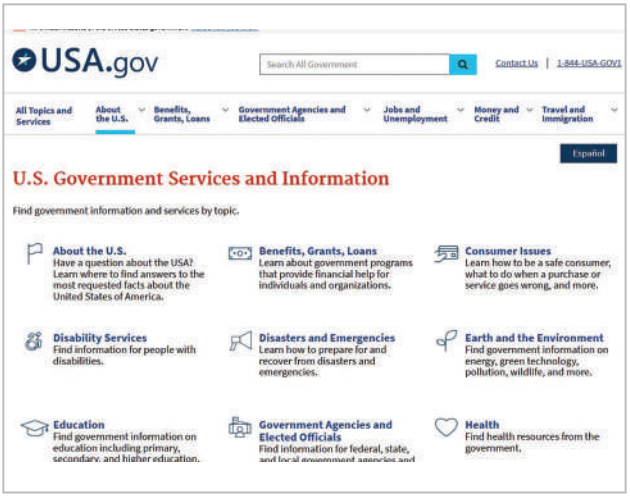
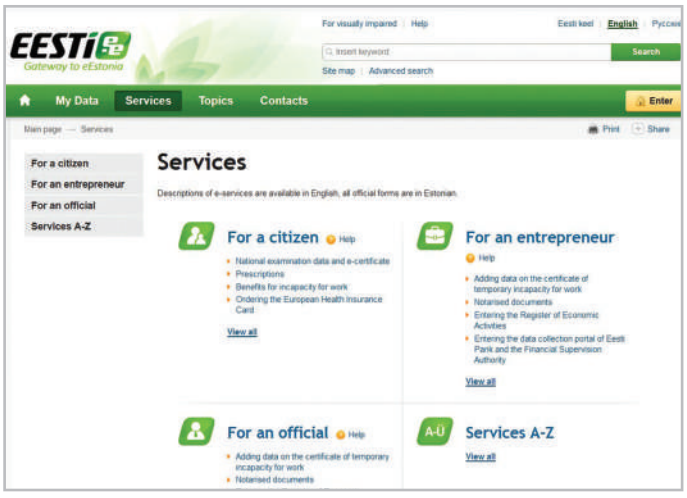
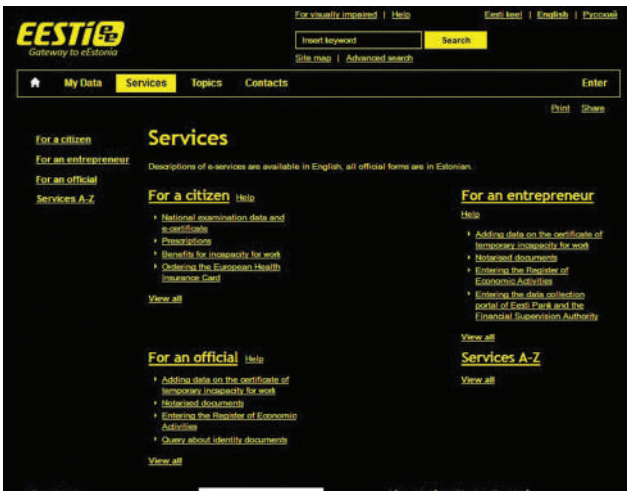


Рис. 4. Сайт уряду США



а



б

Рис. 5. Сайт уряду Естонії: а — звичайний сайт; б — сайт із налаштуванням для людей із вадами зору

Запитання для перевірки знань

1. Що називають системою електронного врядування?
2. Що таке технологія блокчейн? Назвіть її переваги.
3. Які існують види системи електронного врядування?
4. Які функції виконує система електронного врядування?
5. На основі яких принципів створюється система електронного врядування?
6. Поясніть порядок функціонування системи електронного врядування.

2.5. Поняття про штучний інтелект

Чи доводилося Вам користуватися перекладачем Google? Які технології використано в програмах-перекладачах?

Переглядаючи відео на YouTube, здійснюючи покупку на Amazon, звертаючись до Google Assistante, ми часто навіть не замислюємося, що в цих випадках задіяно штучний інтелект.

Штучний інтелект (англ. Artificial Intelligence) — наука (розділ математичної лінгвістики та комп'ютерних наук) і набір технологій, які дають змогу комп'ютеру виконувати різні функції, притаманні людині. Прикладами застосування штучного інтелекту є опрацювання й відтворення голосом друкованого тексту, опрацювання природних мов, розпізнавання образів (комп'ютерний зір).

Тест Тюрінга — емпіричний тест, ідею якого описано так: «Людина взаємодіє з одним комп'ютером і однією людиною. На підставі відповідей вона має визначити, із ким розмовляє: з людиною чи комп'ютерною програмою. Завдання комп'ютерної програми — ввести людину в оману, змусивши зробити неправильний вибір».

Історія штучного інтелекту як нового наукового напрямку бере початок із середини XX ст. у 1950 році англійський учений Алан Тюрінг написав статтю «Обчислювальні машини та інтелект» (англ. Computing Machinery and Intelligence). У статті описано процедуру, за допомогою якої можна визначити момент, коли машина зрівняється в плані розумності з людиною. Ця процедура отримала назву «тест Тюрінга».

Можна виділити такі основні категорії штучного інтелекту.

- Обмежений, або вузький (ANI, Artificial Narrow Intelligence) — спеціалізується в одній конкретній галузі.
- Загальний, або широкий (AGI, Artificial General Intelligence) — може виконувати ті самі завдання, що й людина: обґрунтовувати, планувати, розв'язувати питання, мислити абстрактно, порівнювати комплексні ідеї, швидко навчатися, використовувати накопичений досвід.
- Штучний суперінтелект (ASI, Artificial Superintelligence). Його можливості описав шведський філософ і професор Оксфордського університету Нік Бострем у книзі «Суперінтелект: шляхи, загрози, стратегії» (2014): «Це інтелект, який перевершує людський практично в усіх сферах, включаючи наукові винаходи, загальні пізнання та соціальні навички».

Наразі людство вже досить успішно застосовує вузький штучний інтелект.
У відкритому доступі є курс DeepLearning від Google: https://www.udacity.com/course/deep-learning--ud730.

Наразі людство досить успішно застосовує вузький штучний інтелект.

1. Google Deep Mind продемонструвала штучний інтелект, який володіє «уявою» і здатний аналізувати інформацію та планувати дії без участі людини.

2. Механізм рекомендацій забезпечує Amazon 35 % продажів.

3. Алгоритм Brain, який використовує YouTube для рекомендації контенту, забезпечує перегляд 70 % відео з числа усіх, які переглядаються на сайті.

4. Японська страхова компанія Fukoku Mutual Life Insurance уклала контракт із IBM, в результаті якого замість 34 співробітників система IBM Watson Explorer AI буде переглядати десятки тисяч медичних сертифікатів для визначення умов страхування клієнтів. Це підвищить продуктивність на 30 % та окупиться за 2 роки.

5. Інженери Microsoft спільно з ученими з ICRISAT застосовують штучний інтелект, щоб визначати оптимальний час посіву агрокультур в Індії. Застосунок, що використовує Microsoft Cortana Intelligence Suite, також стежить за станом ґрунту та підбирає необхідні добрива. Це дало змогу підвищити врожайність на 30 %.

6. У китайському місті Ченду відкрився готель Smart LYZ, у якому всі без винятку працівники мають штучний інтелект. Роботи зустрічають туристів, проводять реєстрацію, допомагають заселитися в номер. Найближчим часом планується відкрити ще 50 роботизованих готелів. (Перший у світі готель із роботами з'явився в Японії в 2015 році, проте там зберігається значна частка участі людей.)

7. Компанія IBM розробила інтелектуальну комп'ютерну систему Watson (на базі однойменого суперкомп'ютера), що розуміє англійську мову й може відповідати на широке коло питань (рис. 1).

Рис. 1. Суперкомп'ютер IBM Watson

8. Усе більшої популярності набувають «роботи, що розмовляють». На форумі в Києві восени 2017 року було представлено робота IBM Max (рис. 2).

Рис. 2. Робот IBM Max

9. Кілька років Когнітивна система IBM Watson працює в медичній сфері, допомагає ставити точний діагноз і знаходити ефективний спосіб лікування для онкологічних пацієнтів, ставити діагноз пацієнтам із легеневими хворобами, проблемами з головним мозком. У найближчому майбутньому розробники планують спрямувати можливості системи на діагностування глибокого тромбозу вен, кардіоміопатії, серцевих нападів різних видів.

10. Навіть досвічений фахівець не в змозі відслідковувати всі нові загрози у сфері кібербезпеки (а їх кількість, за даними низки джерел, досягла 75 000). Потрібен цифровий помічник, який сам би аналізував дані про уразливість, віруси, нові програмні інструменти тощо. Watson for Cyber Security засвоює інформацію, працює з даними, використовуючи інформацію про інтернет-загрози, які внесені в базу IBMX-Force Exchange.

11. Сучасна фінансова сфера — дуже складний для вивчення предмет. Банк DBS Bank вирішив випробувати можливості IBM Watson для оптимізації своєї роботи. Заходи, до яких удаються регулятори, соціальні й урядові зміни, флуктуації фінансових ринків — тепер усе це враховується в роботі банку. Окрім того, для оптимізації роботи менеджерів банку та клієнтів використовується Watson Engagement Advisor, консультант від IBM Watson.

12. Безпілотні автомобілі — концепт, над яким працює більшість великих концернів, а також технологічні компанії (Google, Uber тощо), у якому штучний інтелект відповідає за розпізнавання навколишніх об'єктів (автомобіль, пішохід або перешкода).

На рис. 3 зображено Olli, безпілотний автобус, який використовує технологію штучного інтелекту IBM Watson для допомоги людям з інвалідністю.

Рис. 3. Безпілотний автобус, створений LocalMotors

Запитання для перевірки знань

1. Дайте означення штучного інтелекту.
2. Наведіть кілька прикладів успішного застосування штучного інтелекту.
3. Знайдіть в Інтернеті відомості про Watson Kyiv Summit 2017. Підготуйте повідомлення в класі.
4. Коли використовується тест Тюрінга?
5. Як розрізняються категорії штучного інтелекту? Яка категорія нині є найбільш поширеною?
6. Що таке штучний суперінтелект?

2.6. Інформаційна безпека. Керування ризиками в інформаційних системах

Пригадайте, які шкідливі програми ви знаєте. Які способи захисту даних вам відомі?

Кожен користувач хоче бути впевненим у тому, що дані в його комп'ютері захищені. Але із розвитком технологій інтернет-загроз стає все більше.

Останнім часом небезпеку становлять DDoS-атаки — комплекс дій, що блокують доступ до ресурсів Інтернету. Боротьба з такими атаками можлива на рівні компаній та корпорацій.

Інформаційна безпека — це стан захищеності систем опрацювання й зберігання даних, за якого забезпечено конфіденційність, цілісність і доступність даних.

Протокол IPSec пропонує механізм захищеного передавання даних у IP-мережах, забезпечуючи конфіденційність, цілісність та достовірність даних, які передаються через незахищені мережі типу Інтернет.

Пригадаємо основні принципи, на яких базується інформаційна безпека:

- конфіденційність — можливість доступу до інформації тільки визначеним особам;
- цілісність — забезпечення достовірності й повноти інформації і методів її опрацювання;
- доступність — вільний доступ до інформації авторизованим користувачам.

За масштабом загроз розрізняють три рівні інформаційної безпеки (рис. 1). На кожному з них визначається певна сукупність заходів, які спрямовано на убезпечення інформації.

Найчастіше загрози інформаційній безпеці виникають через неуважність працівників IT-сфери (втрата флешки, випадкове пересилання даних іншим особам тощо); використання неліцензійного програмного забезпечення (без супроводження й оновлення розробником); розповсюдження вірусів та інших шкідливих програм через електронну пошту.

Для захисту інформації здебільшого застосовуються:

- фізичні заходи (заборона на доступ стороннім особам, використання безконтактних карт);
- антивірусні програми, системи фільтрування електронної пошти;
- резервне копіювання даних на інші носії інформації.

У деяких випадках застосовується шифрування даних в електронній формі, а на рівні компаній і корпорацій — анти-DDoS.

Основна загроза інформаційній безпеці виникає в комп'ютерних мережах. Для передавання інформації та її убезпечення існують мережеві протоколи різних рівнів, які зазвичай називають протоколами інформаційної безпеки.

Широко відомий протокол IP не має засобів захисту даних, які передаються, навіть не гарантує, що відправник є саме тим, за кого себе видає.

Для убезпечення даних використовують такі технології.

IP Security (скорочено IPSec) — це комплект протоколів, що стосуються питань шифрування, аутентифікації та забезпечення захисту під час транспортування IP-пакетів.

Засоби IPSec використовують стандартні алгоритми (тобто математичні формули) шифрування й аутентифікації, які мають назву «перетворення».

Технологія IPSec і пов'язані з нею протоколи захисту (наприклад, IKE) відповідають відкритим стандартам, які підтримуються групою IETF (Internet Engineering Task Force — проблемна група проектування Internet). IPSec діє на мережевому рівні, забезпечуючи захист і аутентифікацію пакетів IP, що пересилаються між пристроями (сторонами) IPSec — такими, як маршрутизатори, брандмауери, клієнти та концентратори тощо.

IKE (Internet Key Exchange — обмін Internet-ключами) — гібридний протокол, який забезпечує спеціальний сервіс для IPSec: аутентифікацію сторін IPSec, вибір ключів для алгоритмів шифрування, що використовуються в рамках IPSec.

Для надання доступу віддаленим і мобільним користувачам розроблено технологію SSL VPN. Клієнт отримує захищений доступ лише до тих ресурсів, які вважаються необхідними для певних користувачів.

Технологія SSL VPN використовує протокол HTTPS, доступний у всіх стандартних веб-браузерах як безпечний механізм доставки без залучення додаткового програмного забезпечення.

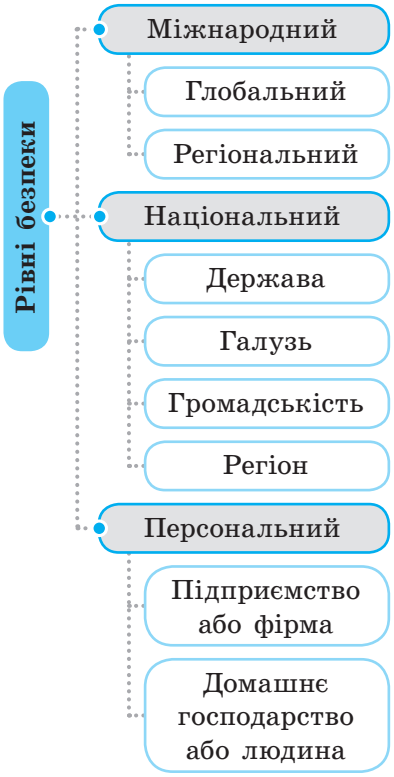


Рис. 1. Рівні інформаційної безпеки

SSL (англ. Secure Sockets Layer — рівень захищених сокетів) — криптографічний протокол, який забезпечує встановлення безпечного з'єднання між клієнтом і сервером.

VPN (англ. Virtual Private Network — віртуальна приватна мережа) — узагальнена назва технологій, що створюють можливість забезпечити одне або кілька мережевих з'єднань (віртуальну мережу) як надбудову над іншою мережею (наприклад, Інтернет).

Класифікація IT-ризиків

- Ризики, викликані діями персоналу (навмисні, ненавмисні)
- Технологічні ризики (збої, відмова в роботі устаткування)
- Ризики, пов'язані з використанням неліцензійного програмного забезпечення

HTTPS (від англ. Hyper Text Transfer Protocol Secure — безпечний протокол передавання гіпертексту) — розширення протоколу HTTP, який підтримує шифрування за допомогою криптографічного протоколу SSL.

Особливістю під'єднання SSL VPN є чітко розмежований контроль доступу до застосунків на основі аутентифікації користувача, мережі, з якої він під'єднується, і рівня безпеки пристрою. Це робить SSL ідеальним інструментом для мобільних співробітників і користувачів неконтрольованих точок.

Застосування ІТ у бізнесі збільшує кількість можливих ризиків. Таким чином, розв'язання та прогнозування проблем, пов'язаних із ІТ, стає одним із основних завдань.

Розглядаючи функціонування ІС, слід ураховувати, що їх інформаційні ресурси та об'єкти можуть піддаватися діям, що призводять до негативних наслідків, які спотворюють дані й можуть призвести до руйнування ІС.

Рис. 2. Оцінювання ризиків

Можливість настання випадкової події в інформаційній системі, унаслідок якої може буде завдано збитків, називають інформаційним ризиком.

Впливаючи на ІС, ризики призводять до збитків підприємства, у чому й полягає економічний зміст поняття «інформаційний ризик». Якщо причини інформаційного ризику породжуються всередині підприємства, то такий ризик є внутрішнім, поза межами підприємства — зовнішнім.

Управління інформаційними ризиками має на меті мінімізацію суми витрат підприємства на протидію інформаційним ризикам і сумарного збитку від цих ризиків (рис. 2).

На підставі всебічного аналізу можливих стратегій керування інформаційними ризиками пропонуються такі стратегії, як прийняття ризику, запобігання ризику, зменшення можливих збитків від ризику, запобігання ризику та зменшення можливого збитку від нього.

Запитання для перевірки знань

1. Що таке інформаційна безпека?
2. Назвіть рівні інформаційної безпеки.
3. Наведіть технології інформаційної безпеки.
4. Які протоколи призначені для захисту даних?
5. Що таке технологія SSL VPN?
6. Наведіть приклади інформаційних ризиків.

Виконайте тестове завдання до розділу 2 з автоматичною перевіркою результату на сайті interactive.ranok.com.ua

Розділ 3. АНАЛІЗ І ВІЗУАЛІЗАЦІЯ ДАНИХ

3.1. Комп'ютерне моделювання об'єктів і процесів. Електронні таблиці

Пригадайте означення понять «модель» і «моделювання».

Із поняттями «модель» і «моделювання» людина знайомиться в дитинстві. Під час гри діти моделюють процес навчання, сімейні відносини (гра «доньки-матері»). Моделювання є обов'язковим етапом наукових досліджень, оскільки створює можливість вивчати об'єкти процеси та явища, за поведінкою яких неможливо безпосередньо спостерігати.

Створюючи модель, зазвичай мають на меті:

- показати вигляд об'єкта, якого реально ще не існує;
- дослідити на моделі поведінку об'єкта, якщо дослідження оригіналу взагалі є неможливим, небезпечним або дорого коштує. Такі ситуації часто трапляються в медицині, авіації, ядерній фізиці й інших галузях;
- навчитися керувати реальним об'єктом, наприклад, автомобілем, літаком чи підводним човном.

Іграшковий автомобіль, плюшевий ведмедик або лялька для багатьох дітей є улюбленими іграшками. Наприклад, коли згодом дитина побачить у зоопарку справжнього ведмедя, то впізнає його.

Модель (від латин. *modulus* — зразок) — спрощений аналог (образ) будь-якого об'єкта, процесу або явища, який використовується як замінник оригіналу, для вивчення його властивостей.

Досліджуваний об'єкт, стосовно якого створюється модель, називають **оригіналом**, **прототипом**.

Властивості об'єкта, які відтворені в моделі, називають її параметрами. Залежно від мети дослідження вибраних параметрів для одного й того самого об'єкта, процесу або явища можуть бути створені різні моделі. Пригадайте типи моделей, за якими ознаками їх класифікують. Тип моделі визначається не тільки метою її створення, й засобами, які для цього використовуються. Схематично процес моделювання подано на рисунку:

Схематичне подання процесу моделювання

Кожен об'єкт має велику кількість різних властивостей. У процесі побудови моделі виділяють найсуттєвіші властивості, які відповідають тематиці дослідження.

Основним завданням процесу моделювання є створення моделі, яка найбільше наближена до оригіналу та властивості

Основні етапи реалізації інформаційної моделі на комп'ютері

I. Постановка задачі
II. Розроблення інформаційної моделі
III. Вибір програмного засобу для реалізації моделі
IV. Безпосередня реалізація моделі
V. Аналіз отриманих результатів
VI. Прийняття відповідного рішення

Особливості MS Excel

- Велика кількість убудованих функцій (математичних, статистичних тощо, для розв'язування задач із багатьох галузей)
- Зрозумілий інтерфейс
- Довідкова система з прикладами використання можливостей (що дозволяє легко реалізувати попередньо розроблену математичну модель)
- Можливість вибору різних типів графіків і діаграм для візуалізації та аналізу даних

якої відповідають меті дослідження та перенесення результатів дослідження на оригінал.

Усі моделі можна розподілити на дві великі групи: матеріальні моделі, у яких відтворено ті або інші геометричні, фізичні характеристики оригіналу — їх досліджують у реальних експериментах, та інформаційні моделі, що є описами об'єктів-оригіналів за допомогою схем, графіків, формул, креслень тощо — їх використовують в інформаційному моделюванні.

Комп'ютерне моделювання — це процес створення інформаційних моделей комп'ютерними засобами.

До основних переваг комп'ютерного моделювання належать:

- можливість багаторазового повторення тих самих дій;
- висока наочність візуалізації процесів, які виконуються в моделі;
- безпечність реалізації моделі;
- висока швидкість виконання дослідження моделі;
- отримання результатів моделювання в зручному вигляді для аналізу.

Інформаційна модель — це модель, яка містить інформацію про суттєві для даного розгляду властивості об'єкта, зв'язок між ними та призначена для дослідження стану об'єкта в разі зміни його властивостей.

Прикладом програмних засобів для реалізації комп'ютерного моделювання є системи програмування, табличні процесори, математичні та графічні програмні засоби, системи керування базами даних, програмні засоби тощо. Одним із найдоступніших програмних засобів для використання обчислювальних і графічних можливостей комп'ютера в моделюванні є офісний застосунок MS Excel.

Пригадайте, що в клітинку MS Excel можна ввести *числові дані* (до них належать дата, час, дані грошового формату), *текстові дані* та *формули*. У формулах використовують константи, посилання на клітинки та функції.

Алгоритм вставлення функцій у формулу

Крок 1	Відкрийте вікно **Вставлення функцій**
Крок 2	На стрічці **ФОРМУЛИ** вибиріть функції з потрібних груп або скористайтеся командою **Вставити функцію**
Крок 3	Відкривається вікно введення аргументів функції з підказками щодо значень кожного з них. У поле введення аргументів пропишіть вираз або посилання на клітинку
Крок 4	Якщо формула складається тільки з функції, то спочатку виділіть клітинку (надайте їй статус активної), виберіть необхідну функцію та її параметри
Крок 5	Підтвердьте командою **ОК** для обчислення

Пригадайте типи посилань у формулах MS Excel (приклад). Посилання можна вводити вручну або перейти до аркуша книги MS Excel з необхідними даними й вибрати клітинки вказівником миші.

Щоб вибрати *одну клітинку*, її потрібно виділити й клацнути лівою кнопкою миші (ЛКМ), щоб вибрати *кілька клітинок* блоку, — клацнути ЛКМ на першій клітинці, «протяжкою» виділити необхідний діапазон і відпустити кнопку.

Клітинки, *розташовані не поруч*, виділяють із використанням клавіші Ctrl. Але не для всіх функцій допустимі значення не з сусідніх клітинок.

Формули в MS Excel можна копіювати в інші клітинки. Під час копіювання формул із відносними посиланнями змінятся клітинки з початковими значеннями, а з використанням абсолютних посилань клітинки не змінюються.

Приклад.

Прикладами посилань різного типу є: K10 — відносне; $K$10 — абсолютне; $K10 і K$10 — змішані посилання. Змінити посилання дає змогу функціональна клавіша F4.

Запитання для перевірки знань

1. Із якою метою створюють моделі? Назвіть їх призначення і наведіть приклади.
2. Яку модель називають інформаційною?
3. Скільки моделей може мати досліджуваний об'єкт чи явище? Відповідь поясніть на прикладах.
4. Назвіть етапи побудови інформаційної моделі.
5. Як використати MS Excel для реалізації моделі рівноприскореного руху об'єкта?
6. Які моделі, на вашу думку, доцільно реалізувати на комп'ютері? Наведіть приклади.

3.2. Розв'язування рівнянь, систем рівнянь, оптимізаційних задач із різних предметних галузей засобами інформаційних технологій

Пригадайте, як вставити стандартну функцію, посилання на клітинку в форму Excel.

Комп'ютерне моделювання ефективно використовується для розв'язування задач будь-якої галузі. Математичні моделі процесів, явищ реалізують за допомогою рівнянь або систем рівнянь, і в MS Excel є засоби їх розв'язування. Розглянемо можливості MS Excel, на прикладах.

- **Розв'язування рівнянь методом Підбір параметра**

Приклад 1 (стародавня задача)

Купець придбав 138 аршин чорного та червоного сукна за 540 карбованців (крб). Скільки аршин сукна одного та іншого кольорів придбав купець, якщо червоне коштувало 5 крб за аршин, а чорне — 3 крб.

Складемо математичну модель розв'язання задачі. Нехай кількість аршинчорного сукна x, тоді кількість червоного сукна — $138 - x$. Складемо рівняння: $3x + 5(138 - x) = 540$.

Звичайно, таке рівняння можна розв'язати й усно, але на цьому прикладі розглянемо як працює метод MS Excel Підбір параметра.

Отже, використаємось методом Підбір параметра. Необхідно до задачі скласти математичну модель у вигляді рівняння з одним невідомим. У рівнянні вираз із невідомим — у лівій частині, а в правій — значення цього виразу, константа.

Метод добирає значення невідомого, поки не буде отримано необхідне значення результату обчислення виразу

Складемо таблицю за умовою задачі (рис. 1). У клітинках наведено формули та результати їх обчислення (рис. 2).

Таблиця має містити початкові значення з умови задачі та клітинку з формулою, яка відповідає рівнянню математичної моделі.

Клітинка C2 призначена для невідомого x: видно, що в клітинці C2 кількість чорного сукна дорівнює 1. Оскільки MS Excel не здійснює арифметичні операції з нечисловими даними, то на початку замість x вставляють будь-яке значення. Нуль вставляти не варто, оскільки в рівняннях може виникнути ділення на 0, що дасть помилку обчислень. Із таким довільним значенням, звичайно, не отримаємо правильну вартість усього сукна з умови задачі (540). Але головним є те, що в клітинці D4 вставлено формулу, результатом обчислення якої має бути значення з умови задачі.

	A	B	C	D
1		ціна	кількість	вартість
2	Чорне сукно	3	1	3
3	Червоне сукно	5	137	685
4				688
5				

Рис. 1. Формули обчислення методом Підбір параметра

	A	B	C	D
1		ціна	кількість	вартість
2	Чорне сукно	3	1	=B2*C2
3	Червоне сукно	5	=138-C2	=B3*C3
4				=SUM(D2:D3)
5				

Рис. 2. Результат обчислення методом Підбір параметра

Приклад 2.

Кредит в 1000 у. о. виданий на 5 років під 20 % річних за таких умов повернення: наприкінці 2 року повернути 200 у. о., наприкінці кожного наступного — по 400 у. о. Визначити, яку суму необхідно внести наприкінці першого року, щоб кредит був погашений вчасно (до кінця 5 року). Щороку борг із кредитування обчислюється за формулою:

Борг = Залишок_внеску + Залишок_внеску * Річні/100

(/100 використовується для перетворення кількості відсотків у частини від 1)

Погашення кредиту означає, що борг відсутній: Борг = 0.

У моделі необхідно скласти рівняння для обчислення боргу наприкінці кожного року за наведеною формулою. Результат обчислення боргу за рік залежить від значення залишку боргу за попередній рік.

Позначимо борг наприкінці кожного року як борг1, борг2 тощо. Отже, якщо x — повернення грошей у перший рік, то:

борг1 (наприкінці першого року) = 1000 – x + 0,2 (1000 – x)
борг2 = борг1 – 200 + 0,2 (борг1 – 200)
борг3 = борг2 – 400 + 0,2 (борг2 – 200)
борг4 = борг3 – 400 + 0,2 (борг3 – 200)
борг5 = борг4 – 400 + 0,2 (борг4 – 200)

Оскільки кредит необхідно сплатити за 5 років, то борг5 = 0 (наприкінці п'ятого року необхідно з боргом5 вийти в 0), тобто рівняння матиме такий вигляд:

$$борг4 - 400 + 0{,}2\,(борг4 - 200) = 0.$$

Указане рівняння є математичною моделлю, адже борг4 залежить від значення борг3 і, розмірковуючи так, дійдемо до 1 року, борг якого залежить від x.

У таблиці, побудованій в MS Excel, x у моделі — значення клітинки B4. На рис. 3 подано таблицю зі значеннями після виклику методу Підбір параметра. У цю клітинку введемо 1 — сплачено в 1 рік, у наступні клітинки стовпця B занесемо 200, 400 і 400, як зазначено в умові. Для зручності залишок внеску як різницю боргу та сплаченого обчислимо в стовпці C і остаточний борг на кінець року — у стовпці D.

Вікно Підбір параметра зі введеними посиланнями та значеннями подано на рис. 4.

	A	B	C	D
1		Річні	20	
2	Роки	Повернення	Залишок внеску	Борг
3				1000
4	1	131,17	868,83	1042,59
5	2	200	842,59	1011,11
6	3	400	611,11	733,33
7	4	400	333,33	400,00
8	5	400	0,00	0,00

Рис. 3. Реалізація задачі про борг у **MS Excel**

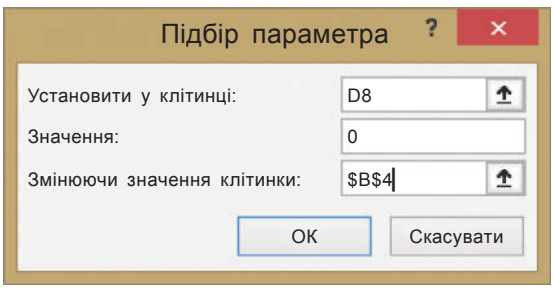


Рис. 4. Вікно **Підбір параметра** для задачі про борг

• Розв'язування оптимізаційних задач методом Пошук рішення

Метод Підбір параметра зручний для складання математичної моделі з одним невідомим. Але часто в описі моделі є кілька невідомих. Залежно від того, якою моделлю описують розв'язання, використовують різні засоби MS Excel. Для систем рівнянь використовують методи їх розв'язування, а для оптимізаційних задач користуються командою Розв'язувач.

У математичній моделі розв'язування оптимізаційних задач необхідно знайти значення параметрів, від яких залежить результат обчислення певної математичної функції (у MS Excel ця функція подається формулою). Шуканий результат може бути максимальним (наприклад, прибуток організації), мінімальним (витрати на сировину) або дорівнювати певному значенню (потужність електричної мережі).

У таких випадках говорять про оптимізацію результату, задача належить до оптимізаційних задач, функція називається цільовою, а параметри — оптимізаційними.

Щоб розв'язок і значення параметрів були реальними, модель обов'язково містить обмеження або на значення параметрів, або на допоміжні величини, які залежать від оптимізаційних параметрів. Обмеження на дані завжди задаються в умові задачі (приклад 3).

Інструмент **Розв'язувач** дає змогу:

- вказувати кілька клітинок для оптимізаційних параметрів
- вказувати обмеження на їх значення
- знаходити розв'язок: цільова функція описується в клітинці формулою, у якій обов'язково є посилання на клітинки з оптимізаційними параметрами
- визначати відповідно до умови один із варіантів роботи з цільовою функцією: знаходити її максимальне, мінімальне, конкретне значення
- отримувати кілька розв'язків

Приклад 3.

Підприємство виготовляє вироби трьох видів: A, B і C. Денний дохід від кожного виду одиниць виробу становить 13, 18 і 22 у. о. Необхідно визначити, за якої кількості виробів підприємство отримає максимальний дохід, з урахуванням того, що:

- загальний обсяг виробництва — 300 одиниць виробів на день;
- підприємству потрібно виготовити 50 одиниць виробу A для виконання існуючого замовлення; 40 одиниць виробу B — для виконання планового замовлення; а оскільки збут виробів C відносно невеликий, то їх необхідно виготовляти не більше ніж 40 одиниць.

Складемо математичну модель. Позначимо кількість виробів кожного виду через a, b, c. Складемо рівняння для цільової функції:

$$\text{дохід} = 13a + 18б + 22c.$$

Цільова функція — дохід — залежить від значень параметрів a, $б$, c і має бути спрямована на отримання максимального значення. Оптимізаційними параметрами є a, b, c — їх значення впливають на значення цільової функції.

Під час пошуку максимального значення доходу необхідно врахувати обмеження на значення параметрів a, $б$, c:

$a + b + c = 300;$

$a \geq 50;$

$b \geq 40;$

$c \leq 40;$

a, b, c — цілі додатні значення.

Щоб реалізувати модель у MS Excel із використанням надбудови Розв'язувач необхідно дотримуватися певних правил:

- у MS Excel обов'язково має бути діапазон клітинок для значень оптимізаційних параметрів;
- у таблиці має бути клітинка з формулою обчислення цільової функції;
- таблиця обов'язково містить клітинки зі значеннями параметрів, щоб мати можливість посилатися на них.

Приклад 3 (продовження)

Розв'яжемо задачу за допомогою надбудови Розв'язувач. У стандартному наборі команд MS Excel команда запуску надбудови Розв'язувач відсутня, її необхідно встановити на стрічку ДАНІ. Для цього виконаємо такі дії.

1. Перейдемо в меню ФАЙЛ — Параметри. Виберемо Надбудови.

2. У вікні, що відкрилося, виберемо Пакет аналізу й активуємо кнопку Перейти — відкриється вікно Надбудови.

3. Поставимо галочку біля Розв'язувач. Підтвердимо дії кнопкою ОК. На стрічці ДАНІ з'явилася команда Розв'язувач. Скористаємося математичною моделлю для створення таблиці в середовищі MS Excel (рис. 5).

4. У клітинки C2:C4 введемо значення для виробів a, b, c, можна ввести і по 1.

5. Після запуску команди Розв'язувач відкриється вікно Параметри розв'язувача (рис. 6, с. 169), у якому:

1) у поле Встановіть цільову клітинку введемо клітинку D5 з підрахунком загального доходу і вкажемо, що функція прямує до максимуму;

2) у поле Змінні клітинки введемо клітинки C2:C4 (клітинки містять невідомі).

	A	B	C	D
1	Вироби	дохід з одиниці	кількість	дохід
2	a	13	100	1300
3	б	18	100	1800
4	c	22	100	2200
5	всього		300	5300
6				

Рис. 5. Таблиця MS Excel для моделі задачі про виробництво

6. Для введення в поле Обмеження скористаємося кнопкою Додати, щоб увести обмеження на значення невідомих — обмеження вводять у вікні Додати обмеження (рис. 7, с. 159).

7. Підтвердимо дії — кнопкою Розв'язати у вікні Розв'язувач. У вікні, що відкрилося, залишаємо активною команду Зберегти знайдений розв'язок; клацнемо кнопку ОК.

Дані таблиці будуть змінені (рис. 8):

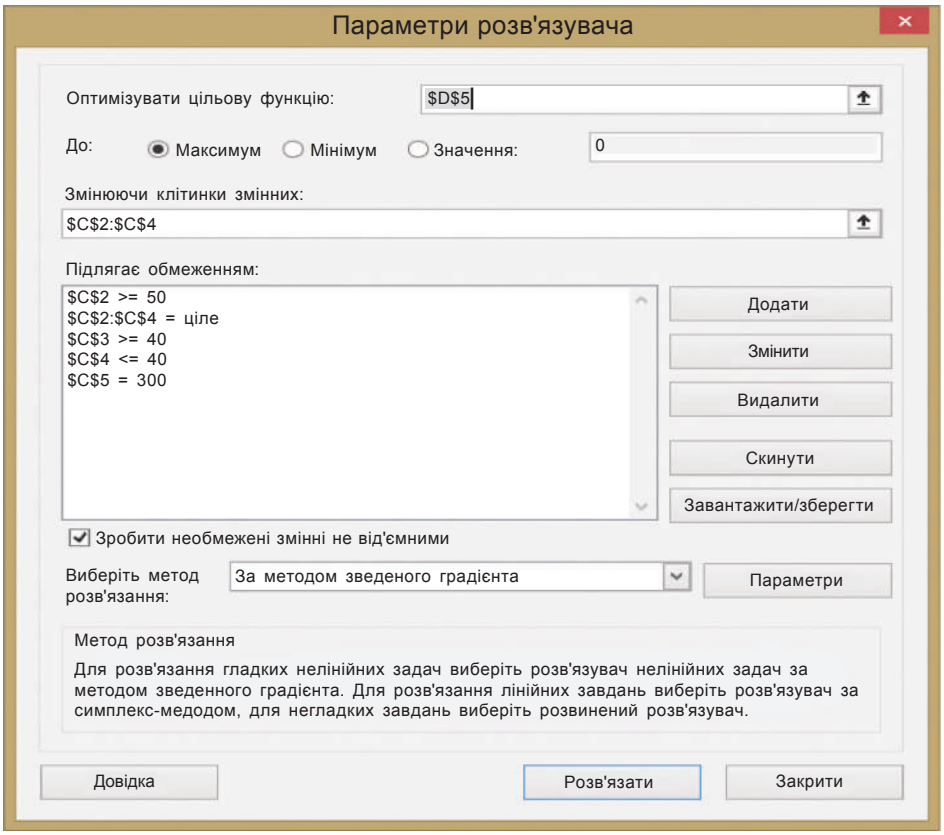


Рис. 6. Вікно Параметри розв'язувача

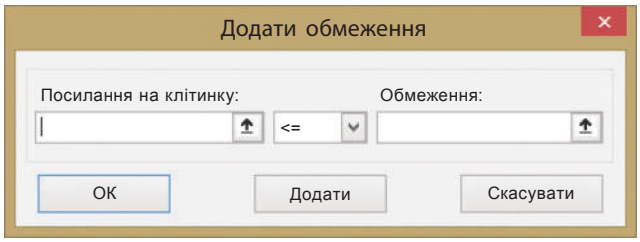


Рис. 7. Вигляд вікна додавання обмежень для Розв'язувача

	A	B	C	D
1	Вироби	дохід з одиниці	кількість	дохід
2	а	13	50	650
3	б	18	210	3780
4	с	22	40	880
5	всього		300	5310

Рис. 8. Таблиця з результатом розв'язання задачі про виробництво

Запитання для перевірки знань

1. Для чого використовують метод **Підбір параметра** в MS Excel?
2. Які задачі належать до оптимізаційних? Наведіть приклади.
3. Для чого використовують надбудову **Розв'язувач** у MS Excel?
4. Як установити **Розв'язувач** на стрічку?
5. Якою має бути математична модель для використання методу **Підбір параметра** в MS Excel?
6. Які правила побудови таблиці на основі математичної моделі для використання методу **Розв'язувач** у MS Excel?

Завдання для самостійного виконання

1. Відкрийте MS Excel і заповніть діапазон A1:D4 за зразком (див. рис. 4). Зробіть активною клітинку D4.
2. Запустіть метод Підбір параметра. Для цього:
   1) перейдіть до стрічки **ДАНІ**;
   2) перейдіть до групи **Робота з даними**;
   3) зверніться до списку **Аналіз «якщо»**;
   4) виберіть команду **Підбір параметра** — відкриється однойменне вікно (рис. 8).
3. У вікні **Підбір параметра** в полі **Встановити в клітинці** показано D4, оскільки ця клітинка була активною перед викликом метода. Якщо була активною інша клітинка, вказують клітинку з формулою моделі.
4. У поле **Значення** введіть значення з умови задачі — 540.
5. У поле **Змінюючи значення клітинки** введіть або клацніть клітинку C2, яка у математичному рівнянні відповідає за **x**.
6. Підтвердіть кнопкою OK і подивіться на дані таблиці.

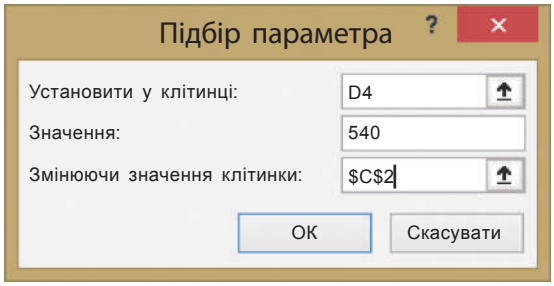


Рис. 8. Вікно **Підбір параметра**

3.3. Матричні операції. Розв'язування систем лінійних рівнянь

Пригадайте, що таке масив значень.

Приклад 1.

Група людей подорожує до іншої країни. У туроператора створено таблицю для перерахунку вартості подорожі в різній валюті (рис. 1).

	A	B	C	D	E	F	G
1	Значення курсу	Вартість на 1 особу, у. о.			Вартість на 1 особу, грн		
2	Тур групи	Подорож1	Подорож2	Подорож3	Подорож1	Подорож2	Подорож3
3	Група 1						
4	Група 2						
5	Група 3						
6	Група 4						

Рис. 1. Таблиця перерахунку вартості подорожі

Діапазон B3:D6 містить числові значення — вартість подорожі одного туриста чи однієї туристки в умовних одиницях. Необхідно перерахувати вказану вартість на гривні, результат занести в діапазон E3:G6. У клітинці A1 вказано курс валюти на поточний час. У клітинку E3 можна вставити формулу множення = B3*$A$1, скопіювати її і вставити в діапазон E3:G6.

MS Excel завдяки своїм можливостям дозволяє розв'язувати задачі різними методами, використовуючи комбінації різних функцій. У прикладі, можна також використати множення масиву на число.

Масивом у MS Excel є прямокутний діапазон значень. Операції з масивами здійснюються функціями в групі Математичні стрічки ФОРМУЛИ.

Здійснюючи операції із масивами, у формулах необхідно вказувати посилання на діапазони клітинок масивів та арифметичні знаки або функції. Після введення формули опрацювання масивів слід натиснути сполучення клавіш Ctrl + Shift + Enter, щоб відбулося заповнення вихідного діапазону, який теж є масивом значень. У масиві після використання формул не можна видаляти елементи.

Повернемося до його прикладу 1. Тут масивом буде діапазон вартості в умовних одиницях. Кожен елемент масиву необхідно помножити на курс валюти, у результаті теж отримаємо масив — прямокутний діапазон клітинок із результатами обчислень: скористаємося операцією множення значення на масив = E3:G6 * $A$1 і підтвердимо операцію з масивом клавішами Ctrl + Shift + Enter.

Операцією транспонування є заміна орієнтації початкового масиву з вертикальної на горизонтальну і навпаки.

Наприклад, масив із 3 стовпців і 6 рядків після операції буде мати 6 стовпців і 3 рядки зі значеннями початкового масиву: перший рядок масиву стає першим стовпцем нового масиву, другий рядок масиву стає другим стовпцем нового масиву тощо. А горизонтальний одновимірний масив стане вертикальним. Це дуже корисна функція, оскільки в деяких інших функціях опрацювання одновимірних масивів вказано, що масив має бути вертикальним.

У MS Excel функція транспонування записується так:
TRANSPOSE(масив) (ТРАНСП(масив)).
Для множення масивів користуються функцією
MMULT(масив; масив) (МУМНОЖ(масив; масив)).

Щодо доцільності використання масивів розглянемо приклад 2 (приклад 2).

Операції з масивами:
- додавання (віднімання, множення або ділення) значень масиву на число
- додавання (віднімання, множення або ділення) значень одного масиву на значення іншого
- транспонування масиву;
- знаходження визначника масиву
- знаходження оберненого визначника масиву тощо

Приклад 2.

Для шкільних змагань із волейболу готуються три команди: команда дівчат і дві команди хлопців. Школі необхідно закупити м'ячі для тренувань, взуття та форму для змагань (кількість буде різною, оскільки в декого залишилося взуття та форма з минулого року). Порахуємо вартість закупок для дівчат і хлопців.

Дані задачі одразу перенесено в таблицю MS Excel (рис. 2, таблицю подано з результатом обчислень).

Результатом обчислень буде сума добутків кожного типу закупівлі на вартість одиниці. Для підрахунку витрат на команду дівчат у клітинку C7 можна ввести формулу:

=$B4*C4+$B5*C5+$B6*C6

	A	B	C	D
1	Вартість замовлення			
2		Вартість	Кількість	
3	Назва	одиниці	Команда дівчат	Команди хлопців
4	М'ячі	20	4	7
5	Форма	300	9	20
6	Взуття	250	7	16
8			Вартість для команди дівчат	Вартість для команд хлопців
9			$4 530	$10 140

Рис. 2. Таблиця підрахунку вартості закупівлі

Далі слід скопіювати та вставити формулу в клітинку D7 і отримати вартість витрат на команди хлопців. Для наведеної у прикладі кількості даних уведення формули не потребує багато часу. Але якщо кількість даних набагато більша, краще використати функцію підрахунку суми добутків елементів масивів:

SUMPRODUCT(масив; масив...) або (СУММПРОИЗВ(масив; масив...)).

Для команди дівчат підрахунок здійснюють за формулою:

=SUMPRODUCT($B$4:$B$6;C4:C6).

Пригадаємо функцію обчислення середнього значення кількох чисел у MS Excel і розглянемо приклад (приклад 3).

Приклад 3.

На багатьох онлайн — курсах результати не обчислюють як середнє значення результатів тестування впродовж слухання курсу, оскільки кожен тест може мати свій коефіцієнт. У такому випадку результатом буде сума добутків результату кожного тесту на коефіцієнт цього тесту. Так само обчислюються прохідні бали у ВНЗ за результатами ЗНО й атестату.

На рис. 3 в таблиці наведено початкові дані тестування. Якщо здійснити множення масиву з результатами тесту на масив із коефіцієнтами, отримаємо шукані значення для кожного. Таке обчислення називають знаходженням середньозваженого значення.

1	Результати тестів					
2	Им'я	Тест 1	Тест 2	Тест 3	Тест 4	Средневз.
3		91	94	91	87,5	
4		98	87,5	79	86	
5		73	36	56	78	
6		65	70	72	84	
7						
8	Вага					
9	Тест	Вес				
10	Тест 1	0,2				
11	Тест 2	0,25				
12	Тест 3	0,15				
13	Тест 4	0,4				
14	Сумарно	1				

Рис. 3. Таблиця підрахунку результатів тестування

У MS Excel запропоновано **метод оберненої матриці** (аналогічний методу Крамера), **метод Гауса** і **метод Зейделя**.

Операції з масивами часто використовують для розв'язування систем лінійних рівнянь.

Математичною моделлю розв'язування задач може бути система лінійних рівнянь із кількома невідомими. Із курсу математики ви знаєте кілька методів їх розв'язування.

Познайомимося з методом оберненої матриці та алгоритмам його використання. Кількість рівнянь у системі має бути така сама, як і кількість невідомих. Рівняння системи перетворюють так, щоб у лівій частині був вираз із невідомими, а в правій — константи. Надалі вже використовують середовище MS Excel.

Алгоритм використання методу оберненої матриці для системи лінійних рівнянь

Крок 1	Створіть таблицю з даними, які відповідають конкретній системі рівнянь. Таблиця містить стільки стовпців, скільки є невідомих, і стовпець для констант із правої частини. Кількість рядків відповідає кількості рівнянь. Зазвичай таблиця з коефіцієнтами при невідомих є матрицею. Якщо в якомусь рівнянні немає невідомого, це означає, що його коефіцієнт дорівнює 0; у таблицю заносять 0.
Крок 2	Для знаходження оберненої матриці виділіть діапазон за розміром матриці коефіцієнтів, викличте функцію =MINVERSE(діапазон матриці) або (=МОБР(діапазон матриці)) і підтвердіть сполученням клавіш Ctrl + Shift + Enter (працюють з матрицею, а не з окремим значенням).

Крок 3

Знайдіть розв'язок:
1) виділіть вертикальний діапазон із такою кількістю клітинок, скільки є невідомих;
2) вставте формулу множення матриць =MMULT(діапазон матриці; діапазон стовпця з коефіцієнтами) або =МУМНОЖ(діапазон матриці; діапазон стовпця з коефіцієнтами);
3) підтвердіть сполученням клавіш Ctrl + Shift + Enter.

Запитання для перевірки знань

1. Що у MS Excel називають масивом?
2. Як підтверджується введення у формулу функцій опрацювання масивів?
3. Що означає середнє рівнозважене значення в масиві? Як його обчислити?
4. Який метод можна використати для розв'язування системи лінійних рівнянь у середовищі MS Excel?
5. Для якої системи рівнянь ввели такі дані в таблицю MS Excel:

1	–2	1		–4
0	1	2		4
2	3	–2		2

Завдання для самостійного виконання

1. Підрахуйте результат проходження курсів ваших однолітків. Для цього виконайте такі дії.

1) Створіть таблицю за зразком (див. рис. 3).
2) Уведіть значення імен у стовпець А, виділіть діапазон F3:F6.
3) Перейдіть на стрічку ФОРМУЛИ. Зверніться до групи функцій Математичні. Викличте функцію MMULT(МУМНОЖ).
4) У поле першого масиву введіть діапазон B4:E7; у поле другого масиву введіть діапазон И11:И14.
5) Підтвердьте сполученням клавіш **Ctrl + + Shift + Enter.**

2. Застосуйте описаний алгоритм для розв'язування системи рівнянь

$$\begin{cases} 2x_1+6x_2+4x_3=8, \\ x_1+5x_2+4x_3=8, \\ x_1+5x_2+7x_3=17. \end{cases}$$

1) У таблицю в діапазон A2:C4 внесіть коефіцієнти при невідомих: 2, 6, 4; 1, 5, 4; 1, 5, 7.
2) У діапазон D2:D4 внесіть значення констант (8, 8 і 17).
3) Обчисліть визначник матриці: у клітинку D5 введіть формулу

=MDETERM(A2:C4) або (=МОПРЕД(A2:C4)).

3. Порахуйте визначник вручну й порівняйте значення.
4) Для знаходження оберненої матриці виділіть діапазон A7:C9, викличте функцію =MINVERSE(A2:C4) або (=МОБР(A2:C4)) і підтвердьте сполученням клавіш Ctrl + Shift + Enter.
5) Для отримання розв'язку виділіть діапазон D6:D9; вставте формулу множення матриць 0:

=MMULT(A7:C9;D2:D4) або
(=МУМНОЖ(A7:C9;D2:D4)).

6) підтвердьте сполученням клавіш **CtrL + Shift + + Enter** і отримайте результат (рис. 4). Отже, маємо: $x_1 = 1$; $x_2 = -1$ і $x_3 = 3$.

	A	B	C	D
1	Коефіцієнти при невідомих			Константи
2	2	6	4	8
3	1	5	4	8
4	1	5	7	17
5	Визначник матриці			12
6	Обернена матриця			Розв'язок
7	1,25	-1,83	0,33	1
8	-0,25	0,83	-0,33	-1
9	0,00	-0,33	0,33	3
10				

Рис. 4. Таблиця розв'язування системи рівнянь

4. Розв'яжіть зазначену систему рівнянь вручну і перевірте правильність розв'язування в MS Excel.

3.4. Основи статистичного аналізу даних. Ряди даних. Кореляційний аналіз даних

У будь-якому експерименті досліджують властивості певної кількості однотипних об'єктів, щоб дійти висновку щодо досліджуваних властивостей усіх таких об'єктів (приклад 1).

Вибірка, або **вибіркова сукупність,** — частина генеральної (загальної) сукупності об'єктів, вибраних випадковим чином, яка охоплюється експериментом.

Генеральна сукупність може бути як скінченною, так і нескінченою, а вибіркова сукупність є скінченною завжди. Для отримання достовірної характеристики вибірки необхідно визначитися, які властивості досліджуються та які способи побудови вибірки використовуються.

Опрацювання даних, які є властивостями певних об'єктів, здійснення аналізу цих даних для визначення властивостей усієї сукупності таких об'єктів здійснюється за правилами математичної статистики.

Приклад 1.
Для дослідження впливу певного препарату на рухові можливості птахів випадковим чином добирають групу птахів. Ця група є вибіркою з генеральної сукупності — кількості всіх птахів цього виду чи взагалі всіх птахів світу залежно від характеристик дослідження.

Математична статистика — наука про математичні методи систематизації, опрацювання та використання статистичних даних для наукових і практичних висновків.

Мета статистики (від лат. status — стан справ/речей) — виявити особливі закономірності. Сьогодні статистика застосовується практично в усіх сферах — від моди, кулінарії, садівництва до астрономії, економіки, медицини й ін.

Для аналізування кількісних характеристик використовують статистичні дослідження, кожне з яких складається, по-перше, із отримання кількісних даних і, по друге, зі статистичнго опрацювання даних.

Сукупність даних, які дають кількісну характеристику властивостей досліджуванних об'єктів, процесів або явищ, називають статичними даними, або рядом даних.

Розглянемо, які характеристики вибірки опрацьовуються в статистичному аналізі.

- Обсяг вибірки — кількість елементів у вибірці.
- Розмах вибірки — різниця між максимальним і мінімальним значеннями елементів вибірки.
- Середнє арифметичне — частка від ділення суми усіх значень елементів вибірки на обсяг вибірки.
- Мода ряду чисел — значення, яке найчастіше повторюється в ряді даних. Якщо дані у вибірці не повторюються, мода не обчислюється. Якщо в ряді даних є числа, які трапляються однакову кількість разів, мода матиме кілька значень.
- Медіана впорядкованого ряду чисел із непарним числом членів — число, яке виявиться посередині.
  Медіана впорядкованого ряду чисел із парним числом членів — середнє арифметичне двох чисел, записаних

У математичних моделях вибіркою є часовий або просторовий ряд статистичних даних. Якщо досліджувалися властивості одного об'єкта в певні моменти часу, то дані складають **часовий ряд**, а якщо дані збиралися з різних об'єктів, то вони належать **просторовому ряду**.

посередині. Отже, половина значень вибірки менші за медіану, а половина — більші.

- Частота — число повторень значень вибірки в заданих інтервалах. Якщо інтервалом є вибірка, то частотою є повторення кожного значення у вибірці.
- Відносна частота — це відношення частоти до загальної кількості даних у вибірці.
- Стандартне відхилення (середньоквадратичне відхилення, СКВ) — показник розсіювання статистичних даних відносно середнього значення вибірки.

Розглянемо кожну характеристику на прикладі 2.

Приклад 2.

Розглянемо зріст учнів паралелі класів на прикладі довільно вибраних 10 хлопців. Дані занесемо в таблицю й упорядкуємо за зростанням.

Обсяг вибірки: 10 (вимірювали зріст 10 осіб).

Розмах вибірки 14; максимальне значення 182, мінімальне 168, різниця: 182 − 168 = 14.

Середнє арифметичне значення ряду чисел: 174.

Мода: 172, оскільки 172 трапляється частіше за інші значення — тричі.

Медіана: 173; у нашому прикладі 10 елементів — число парне, в упорядкованому наборі цих елементів посередині (тобто 5-й і 6-й елементи мають значення 172 і 174 відповідно). Середнім арифметичним чисел є 173, бо (174 + 174) : 2 = 173. Якби досліджували зріст 9 осіб, посередині був би 5-й елемент зі значенням 172 — медіана мала б саме таке значення.

Для обчислення частоти та відносної частоти додамо додаткові стовпці в таблицю, введемо дані. Міркуємо так: зріст 168 см має 1 учень, отже, частота повторень значення 168 дорівнює одиниці і т. д.

Відносна частота подана часткою від ділення частоти на 100 % (її часто подають у відсотковому вигляді):

№ з/п	Учень	Зріст, см	Частота	Відносна частота
1	Учень 1	168	1	0,1
2	Учень 2	169	1	0,1
3	Учень 3	172	3	0,3
4	Учень 4	172	3	0,3
5	Учень 5	172	3	0,3
6	Учень 6	174	1	0,1
7	Учень 7	175	1	0,1
8	Учень 8	178	2	0,3
9	Учень 9	178	2	0,3
10	Учень 10	182	1	0,1

Для обчислення стандартного відхилення (СКВ) використовують формулу

$$\text{СКВ} = \sqrt{\frac{\sum_{i=1}^{n}(CA - x_i)^2}{n}},$$

де CA — середнє арифметичне елементів вибірки;
x_i — і-тий елемент вибірки;
n — обсяг вибірки.

Якщо кількість елементів у вибірці перевищує 30, то знаменник дробу під коренем набуває значення n-1 (у деяких джерелах зазначено, що до 30 необхідно ділити на n-1, в інших випадках — на n).

Як бачимо, СКВ є коренем із суми квадратів різниць між елементами вибірки x_i і середнім арифметичним (СА), поділеним на кількість елементів у вибірці n.

Алгоритм обчислення СКВ

Крок 1	Обчисліть середнє арифметичне вибірки.
Крок 2	Відніміть середнє арифметичне від кожного елемента вибірки.
Крок 3	Усі отримані різниці піднесіть до квадрата.
Крок 4	Підсумуйте всі отримані квадрати.
Крок 5	Поділіть отриману суму на кількість елементів вибірки.
Крок 6	Обчисліть квадратний корінь із отримані суми.

Поміркуємо, навіщо обчислювати СКВ, якщо є середнє арифметичне. Розглянемо приклад 3.

Приклад 3.

Порівняємо дві невеликі вибірки. Продажі певного товару за тиждень в одному магазині такі: 31, 33, 32, 36, 32, 31, а в другому — 22, 34, 56, 52, 10, 21. Обома магазинами володіє одна особа, і вона хоче проаналізувати діяльність менеджерів магазинів. Середні продажі першого магазину 32,5, другого — теж 32,5.

Отже, середнє арифметичне в нашому випадку однакове. А якщо порахувати статистичне відхилення, то для першого магазину воно буде приблизно 2, а для другого — 19. Який магазин працює стабільніше? Перший. А для працівників другого магазину виникає завдання стабілізувати роботу з продажів. Отже, СКВ вказує на розбіжність між даними вибірки і середнім арифметичним: що ближче значення СКВ до 0, то менше розбіжність даних у вибірці.

Зазвичай для обчислень як статистичного відхилення, так і інших статистичних характеристик користуються програмними засобами, наприклад MS Excel (приклад 4).

Приклад 4.

В Україні за результатами складання ЗНО з певного предмета (знайдіть дані пор випускників!) потрібно рахувати кількість значень, які потрапляють у зазначені інтервали (1–3; 4–5; 7–9 і 10–12 — знайомі вам оцінки). Для цього зручно використати функцію FREQUENCY(ЧАСТОТА) в MS Excel. Як правильно створити таблицю, як у середовищі MS Excel опрацювати набори даних, як візуально проаналізувати отримані результати, — обговоримо в наступних параграфах.

У вже розглянутих прикладах ми аналізували статистичні характеристики для параметрів одного типу. Але є області статистичного дослідження, у яких аналізують вплив зміни одного параметра на зміну іншого.

Кореляційний аналіз, або кореляція (від латин. *correlatio* — співвідношення, взаємозв'язок) — статистичний взаємозв'язок двох або більше показників.

Кореляція використовується для визначення, чи є між показниками в одній або двох вибірках зв'язок, і для аналізу, чи можна за величиною одного показника передбачити можливе значення іншого. Наприклад, необхідно встановити зв'язок між ціною техніки і тривалістю експлуатації, збільшенням зросту і вагою дітей тощо.

Основним у кореляційному аналізі є коефіцієнт кореляції, який може розраховуватися різними методами залежно від класу задач. Діапазон його значень від –1 до 1.

Розрізняють додатну кореляцію: коли зміна значення одного показника підвищує значення іншого і, навпаки, від'ємну кореляцію : зменшує значення іншого показника.

При значенні коефіцієнта 0 лінійної залежності між вибірками не існує.

Зазвичай коефіцієнт кореляції позначається *r*.

Для візуалізації в кореляційному аналізі користуються **кореляційним полем** — прямокутною системою координат, яка має дві осі: по одній відстежуються значення одного показника, по другій — іншого; на перетині — позначки. За розташуванням позначок у полі визначають взаємозв'язок між показниками.

Запитання для перевірки знань

1. Що вивчає статистика?
2. Що таке вибірка даних?
3. Назвіть і дайте означення статистичним характеристикам.
4. Як ви розумієте поняття кореляції?
5. Наведіть приклади доцільності кореляційного аналізу.
6. Чому обсяг вибірки завжди є скінченною величиною, а обсяг генеральної сукупності може бути нескінченним?

3.5. Обчислення основних статистичних характеристик вибірки засобами табличного процесора

Пригадайте можливості табличного процесора. Наведіть приклади використання автоматизованого опрацювання табличних даних.

Розглянемо, як використовувати програмний засіб MS Excel для обчислення статистичних даних на прикладі 1.

Приклад 1.

Проаналізуємо температурний режим за 15 днів липня. Побудуємо таблицю за зразком (стовпці А і Б у табл. 1).

Для обчислення статистичних характеристик скористаємося функціями табличного процесора MS Excel. У табл. 1, 2 проілюстровано статистичне опрацювання температурних даних: у клітинках табл. 1 показано формули обчислень; у клітинках табл. 2 — результати.

Таблиця 1. Формули розрахунку статистичних характеристик

	A	B	C	D	E
1	Липень				
2	День	Темп., °C	Мода	Частота	Відн. частота
3	1	24	=MODE.MOLT(B3:B17)	=FREQUENCY(B3:B17;B3:B17)	=D3/$D$20
4	2	24	=MODE.MOLT(B3:B17)	=FREQUENCY(B3:B17;B3:B17)	=D4/$D$20
5	3	23	=MODE.MOLT(B3:B17)	=FREQUENCY(B3:B17;B3:B17)	=D5/$D$20
6	4	27	=MODE.MOLT(B3:B17)	=FREQUENCYB3:B17;B3:B17)	=D6/$D$20
7	5	30	=MODE.MOLT(B3:B17)	=FREQUENCY(B3:B17;B3:B17)	=D7/$D$20
8	6	30		=FREQUENCY(B3:B17;B3:B17)	0
9	7	26		=FREQUENCY(B3:B17;B3:B17)	=D9/$D$20
10	8	30		=FREQUENCY(B3:B17;B3:B17)	0
11	9	26		=FREQUENCY(B3:B17;B3:B17)	0
12	10	18		=FREQUENCY(B3:B17;B3:B17)	=D12/$D$20
13	11	25		=FREQUENCY(B3:B17;B3:B17)	=D13/$D$20
14	12	27		=FREQUENCY(B3:B17;B3:B17)	0
15	13	25		=FREQUENCY(B3:B17;B3:B17)	0
16	14	21		=FREQUENCY(B3:B17;B3:B17)	=D16/$D$20
17	15	26		=FREQUENCY(B3:B17;B3:B17)	0
18					
19					
20			Обсяг	15	
21			СА	=AVERAGE(B3:B17)	
22			Мін	=MAX(B3:B17)	
23			Макс	=MIN(B3:B17)	
24			Розмах	=D23-D22	
25			Медіана	=MEDIAN(B3:B17)	

Таблиця 2. Результати розрахунку статистичних характеристик

	A	B	C	D	E
1	Липень				
2	День	Темп.	Мода	Частота	Відн. частота
3	1	24	30	2	0,13
4	2	24	26	0	0
5	3	23	#Н/Д	1	0,07
6	4	27	#Н/Д	2	0,13
7	5	30	#Н/Д	3	0,20
8	6	30		0	0,00
9	7	26		3	0,20
10	8	30		0	0
11	9	26		0	0

Продовження табл. 2

	A	B	C	D	E
12	10	18		1	0,07
13	11	25		2	0,13
14	12	27		0	0
15	13	25		0	0
16	14	21		1	0,07
17	15	26		0	0
18				0	
19					
20			Обсяг	15	
21			СА	25,5	
22			Мін	18	
23			Макс	30	
24			Розмах	12	
25			Медіана	26	

Оскільки найбільш поширені функції винесено окремо на стрічках Формули та Основне, останньою краще скористатись для обчислення значень функцій середнього арифметичного, максимуму та мінімуму (рис. 1).

Розглянемо, як використовувати функції статистичного аналізу в MS Excel.

Група Статистична міститься в групі Інші функції (рис. 2). Для обчислення моди призначено дві функції: MODE.MULT(масив) або МОДА.НСК(масив) і MODE.SNGL(масив) або МОДА.ОДН(масив). Аргументами є масив даних, для якого й обчислюється мода.

Функція MODE.SNGL(масив) повертає одне значення. Її використовують для ряду значень, у яких групи з однаковими значеннями не повторюються.

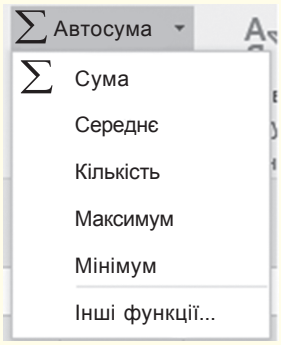


Рис. 1. Меню основних функцій MS Excel

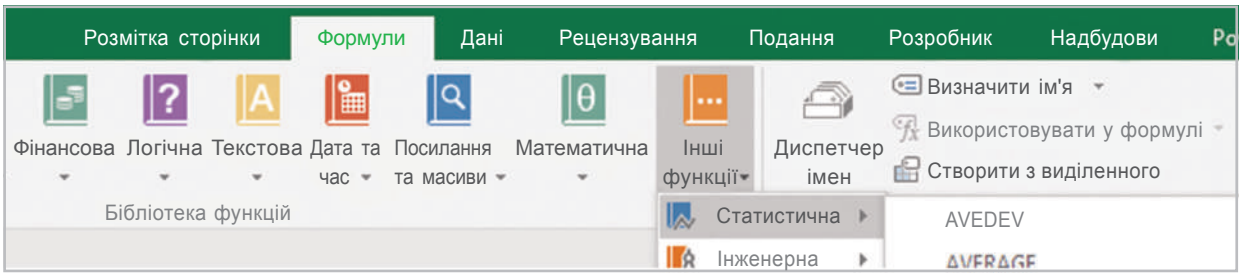


Рис. 2. Група статистичних функцій на стрічці **ФОРМУЛИ**

Функція MODE.MULT(аргументи) або (МОДА.НСК(аргументи)) — гнучкіша. Її використовують, по-перше, для вертикального ряду значень, а по-друге, як функцію масиву. Тобто опрацьовується ряд чисел — масив чисельних значень. Повертає функція не одне значення, а кілька — теж масив чисел.

Як видно з таблиці 2 у прикладу 1 температура 26 °С і 30 °С повторюється найчастіше — по тричі. Функція MODE.SNGL(МОДА.ОДН) поверне результат 30, оскільки саме така температура трапиться вперше в запропонованому ряді значень. Але якщо ряд значень впорядкувати за зростанням, то отримаємо результат 26.

У прикладі 1 для функції обчислення моди виділили 5 клітинок — знайдено 2 найбільші групи зі значеннями, які повторюються, тому в клітинках будуть значення 30 і 26, а в решті — #Н/Д.

Приклад 2.

Нехай є ряд чисел — вік осіб для деякого дослідження. Необхідно порахувати, скільки є осіб, молодших за 14 років скільки віком від 14 до 17 років, від 17 до 22, старших від 22 років. Початковими даними буде діапазон із віком, а в клітинках інтервалів — значення 14, 17, 22.

Отже, для отримання результату потрібно мати чотири клітинки:

1 — для числа осіб, молодших за 14, або яким 14;

2 — для числа осіб віком від 14 до 17 включно;

3 — для числа осіб віком від 17 до 22 включно;

4 — для числа осіб, старших від 22.

Для таких обчислень користувалися функцією **COUNTIF(СЧЁТЕСЛИ)** — функція опрацьовує як числові дані, так і текстові (для них користуються виключно **COUNTIF(СЧЕТЕСЛИ)**). Функція **FREQUENCY(ЧАСТОТА)** використовується для опрацювання цілих і дійсних чисел.

Отже, для вставлення формули з функцією виділяють слід одразу кілька клітинок (ми не знаємо скільки, тому краще зарезервувати більше), після зазначення аргументу функції — діапазону клітинок із числами, натискають сполучення клавіш Ctrl + Shift + Enter.

Якщо масив не містить однакових даних, функція MODE.MULT(масив) або (МОДА.НСК(масив)) повертає значення помилки #Н/Д. Якщо дані введено в рядок (див. приклад 1), B3:P3 — отримаємо горизонтальний ряд даних; наприклад якою застосовуємо функцію транспонування, отримаємо вертикальний — загальний вигляд обчислення моди буде такий:

{=MODE.MULT(TRANSPOSE(масив))} або
({=МОДА.НСК(ТРАНСП(масив))}).

Тут фігурні дужки для формули вказують на опрацювання масиву даних і проставляються автоматично, якщо скористатися сполученням клавіш Ctrl + Shift + Enter.

Для обчислення медіани потрібно викликати однойменну функцію і як аргументи вставити ряд даних (у нашому прикладі B2:B16).

Для підрахунку частоти слід скористися однойменною функцією FREQUENCY(аргументи) (ЧАСТОТА(аргументи)), аргументами якої є ряд даних та інтервали, у яких і рахують повторення значень.

Алгоритм підрахунку частоти

Крок 1	Заповніть діапазон клітинок значеннями інтервалів
Крок 2	Виділіть порожній діапазон клітинок за розміром на одну клітинку більший, ніж діапазон інтервалів
Крок 3	Викличте функцію **FREQUENCY(ЧАСТОТА)**
Крок 4	У поле **Масив даних** уведіть імена діапазону клітинок із числами — початковими даними
Крок 5	У полі **Масив інтервалів** зазначте діапазон клітинок з інтервалами
Крок 6	Підтвердіть дії сполученням клавіш **Ctrl + Shift + Enter**

Функція поверне число значень, які потрапляють у задані інтервали (приклад 2).

У разі порівняння результатів обчислень двох функцій функція FREQUENCY(ЧАСТОТА) повертає кількість повторень для значення, яке першим трапляється в ряді. Для інших таких самих значень повертається 0, а функція COUNTIF(СЧЁТЕСЛИ) повторює обчислюваний результат для всіх даних. У нашому випадку це не впливає на статистичні характеристики. Але під час обчислення відносної частоти у відсотках загальну кількість отримаємо 100 %.

Якщо рахувати число повторень через COUNTIF(СЧЕТЕСЛИ) і результат взяти за основу підрахунку відносної частоти, то під час перевірки отримаємо число, більше від 100 %.

MS Excel містить кілька функцій для обчислення стандартного відхилення. Функції STDEVA(СТАНДОТКЛОНА) і STDEVA(СТАНДОТКЛОНПА) відрізняються значенням середнього: у першій функції використовується середнє вибірки, а в другій — середнє генеральної сукупності.

У наших прикладах не вказуються значення генеральної сукупності, вони одні й ті самі для вибірки й для генеральної сукупності. Обидві функції повернуть однаковий результат.

Функції, як аргументи, можуть набувати і числових значень, і логічних; інші функції MS Excel для обчислення СКВ працюють із числовими значеннями.

Алгоритм обчислення СКВ

Крок 1	Виділіть клітинку для результату.
Крок 2	Викличте функцію STDEVA(СТАНДОТКЛОНА).
Крок 3	У вікні, що відкриється, вставте діапазон значень температури B2:B16.
Крок 4	Підтвердьте кнопкою ОК

Запитання для перевірки знань

1. Які групи функцій містять функції обчислення статистичних характеристик?
2. Якою комбінацією клавіш користуються для підтвердження опрацювання даних масиву?
3. Назвіть функції для обчислення моди.
4. Чому для обчислення моди резервують більше однієї клітинки?
5. Як ви розумієте «інтервал значень» як аргумент функції FREQUENCY(ЧАСТОТА). Поясніть на прикладі.
6. Яке призначення статистичної характеристики відносної частоти? У якому форматі краще аналізувати вказану характеристику: у відсотковому чи ні?

3.6. Візуалізація рядів і трендів даних. Інфографіка

Як краще проаналізувати результати контрольної роботи: порівнювати оцінки учнів у списку чи побудувати стовпчикову діаграму? Які типи діаграм у MS Excel ви знаєте?

Дослідження завжди супроводжуються візуалізацією даних. У наукових експериментах і статистичному аналізі графіки й діаграми не лише наглядно відображають значення, мають змістове навантаження щодо теми дослідження.

Інфографіка (від латин. *informatio* — інформування, роз'яснення, і грец. γραφικός — письмовий) — це візуальне відображення інформації, статистичних даних для простої і наочної демонстрації тенденцій, співвідношень, а також зацікавлення в предметі дослідження.

Завдяки візуалізації відразу видно загальну картину результатів дослідження, велику кількість даних на обмеженому проміжку; акцентується увага на деяких елементах ряду даних; порівнюються фрагменти даних, демонструється тенденція зміни досліджуваних властивостей тощо.

Приклад 1.

В історії інфографіки є кілька відомих робіт, створених у XIX ст. Одна з них — карта походу військ Наполеона француза Жозефа Мінара (рис. 1; джерело: https://pikabu.ru/story/pokhod_napoleona_v_rossiyu_naglyadno_4192832). На реальну географічну карту (простір Париж — Москва) нанесено лінії, що позначають поля відомих битв. Товщина лінії визначає кількість військ (1 мм — 1 тис. воїнів), колір — напрямок руху: червоний — на Москву, чорний — повернення у Францію.

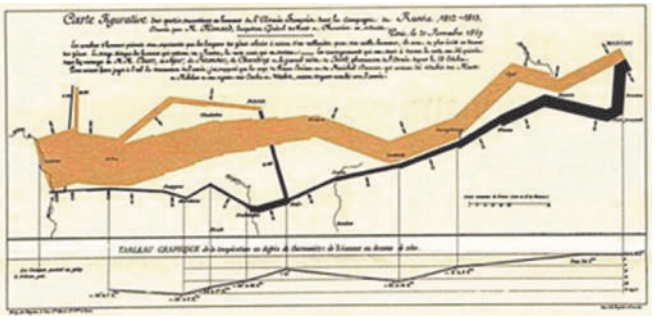


Рис. 1. Жозеф Мінар. Карта руху і повернення з походу військ Наполеона

Поміркуємо, яке інформаційне навантаження має наведена карта: це карта руху французького війська; результат битв у певних точках країн; аналіз військового походу Наполеона; також можна дійти висновку щодо зміни чисельності населення Франції.

У 1854 році Джон Сноу наніс на карту Лондона точки захворювання на холеру, навів статистику смертей у різних будинках, позначив міські джерела води. Так було виявлено колодязь, що був джерелом інфекції.

Для змістового навантаження графіка чи діаграми важливо правильно вибрати їх тип. Розглянемо різні типи діаграм на прикладах.

- **Гістограми**

Гістограми використовують для наочного подання динаміки зміни даних у часі або розподілу даних. Вони мають форму прямокутників (областей), розмір яких відповідає значенням, отриманим в результаті статистичного дослідження.

Гістограми можуть бути як вертикальними (стовпчасті діаграми), так і горизонтальними.

Приклад 2.

За даними, наведеними в прикладі 2, с. 165, побудовано вертикальні гістограми, а на рис. 3 зросту учнів за абсолютним значенням (рис. 2) та за відносною частотою (рис. 3). За другою діаграмою можна легко дійти висновку, якого зросту є більшість учнів.

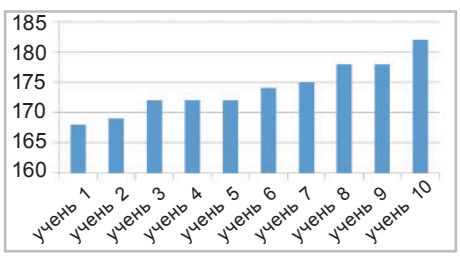


Рис. 2. Гістограма аналізу зросту учнів

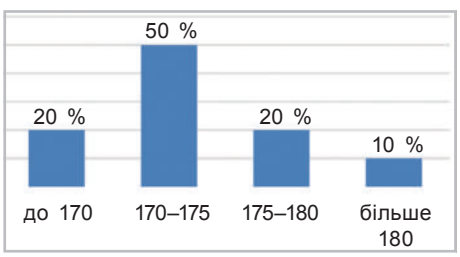


Рис. 3. Гасторгама статистики зросту учнів за інтервалами

Приклад 3.

Якщо у таблицю MS Excel ввести оцінки, які учні класу отримали за практичну роботу з інформатики, то за допомогою функції FREQUENCY (ЧАСТОТА) можна порахувати кількість учнів, які отримали високий бал (інтервал 10–12), аналогічно — інші бали. Гістограма матиме чотири стовпці, висота яких відповідатиме кількості оцінок, які потрапляють у той чи інший інтервал.

Приклад 4.

Щороку Український центр оцінювання якості освіти оприлюднює статистичний аналіз результатів ЗНО. На рис. 4 подано горизонтальну гістограму аналізу складання ЗНО з української мови учнями міста Києва (джерело: https://zno2017.monitoring.in.ua/). Результати відображено по районах, по місту вцілому. Кожна смуга відповідає результаті певного району та міста; кольором позначено інтервали балів.

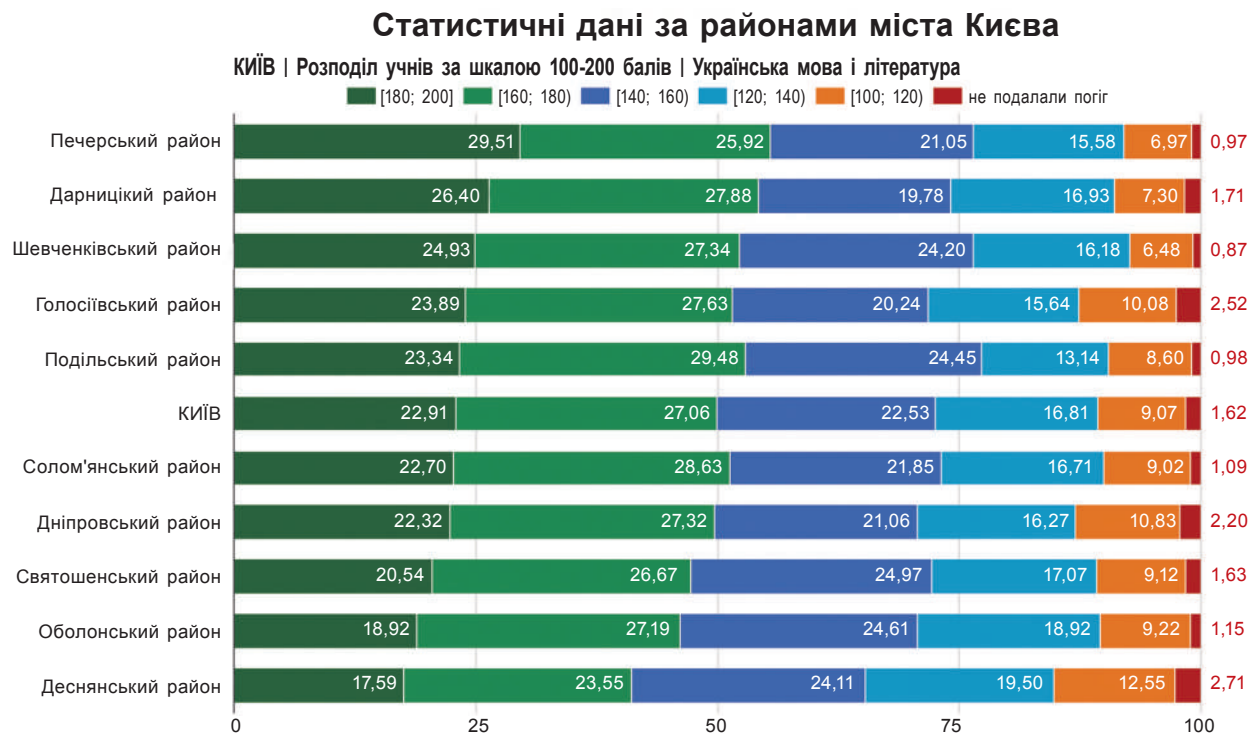


Рис. 4. Гістограма статистичного аналізу результатів ЗНО в місті Києві

- **Кругові діаграми**

Кругові діаграми використовують для унаочнення співвідношення між частинами вибірки. Їх доцільно застосовувати для аналізу розподілу цілої величини (що приймається за 100 %) на частину. Наприклад, зручно розділити піцу між друзями.

На рис. 5 за даними, наведеними в прикладі 2 (с. 165), побудовано кругову діаграму аналізу зросту учнів за відносною частотою.

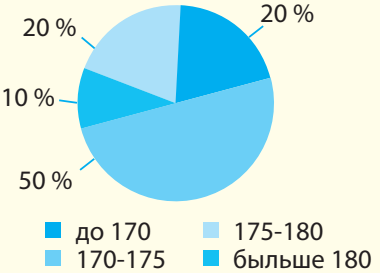
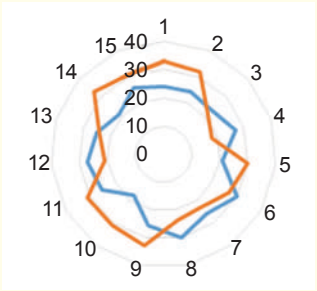
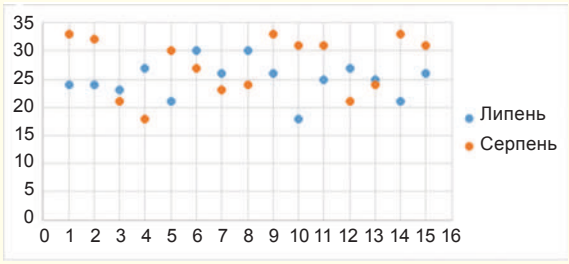


Рис. 5. Діаграма аналізу зросту учнів



а



б

Рис. 6. Приклади діаграм: а — пелюсткової; б — точкової.

Для наочного аналізу рядів даних зручно користуватися *точковою* або *пелюстковою* діаграмою. На рис. 6 показано два типи діаграм для температурних режимів липня і серпня.

Особливою популярністю в дослідженнях користуються *точкові діаграми* та їх різновид — *бульбашкова діаграма*. Розмір бульбашки вказує на значення, а її розташування — на певний ряд досліджень (рис. 7).

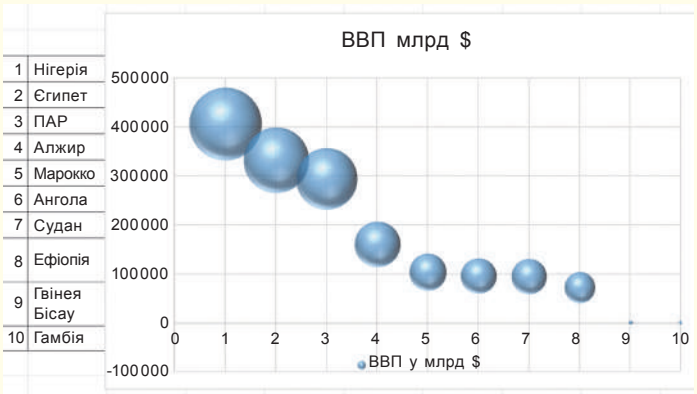


Рис. 7. Діаграма ВВП країн Африки

Якщо в точковій діаграмі для осі x узяти значення температури липня, а для осі y — температуру серпня, то отримаємо діаграму для кореляційного аналізу. Із діаграми на рис. 8 видно збіг зміни значень.

Рис. 8. Візуалізація кореляції температур двох місяців

• Комбіновані діаграми

Ми розглянули деякі діаграми, у яких для порівняння використовували кілька рядів даних. Тип діаграми для цих рядів вибрано однаковий. У MS Excel є можливість створювати комбіновані діаграми, у яких для зображення різних рядів даних використовуються різні типи діаграм (рис. 9).

Прикладом для доцільності використання комбінованих діаграм може бути аналіз підсумкової контрольної з певного навчального предмета порівняно з середньою успішністю учнів: для оцінок за контрольну вибирають тип діаграми — гістограма, а для середнього бала за семестр — графік. Значення горизонтальної осі ідентифікують учнів, а значеннями вертикальної осі — значення оцінок від 1 до 12.

Приклад 5.

На рис. 9 показано комбіновану діаграму двох рядів температурних даних за липень і серпень.

Рис. 9. Приклад комбінованої діаграми

• Лінія тренду в Excel

Лінія тренду — це графічне подання загальної закономірності зміни ряду даних. Лінія тренду є графіком, який описується математичними формулами та до значень якого наближаються значення діаграми. Лінію тренду додають на гістограми, діаграми з областями, лінійчаті діаграми, графіки або точкові діаграми.

За значеннями діаграм не завжди зручно прогнозувати динаміку їх зміни, а лінія тренду дає змогу це зробити. Для точнішого прогнозу треба правильно вибрати тип лінії трендуу.

Говорячи мовою математики, **лінія тренду** є апроксимацією (апроксимація — наближення) значень діаграми до графіка, який описується математичними формулами.

Приклад 6.

На рис. 10 на гістограмі показано оцінки учня чи учениці з одного зі шкільних предметів (стовпці синього кольору). Лінія тренду, подана червоним пунктиром, вказує на зростання рівня знань учнів.

У прикладі скористалися лінійною лінією тренду, бо саме цей тип найкраще описує простий лінійний набір даних і застосовується у випадках, коли точки даних розташовані близько до прямої.

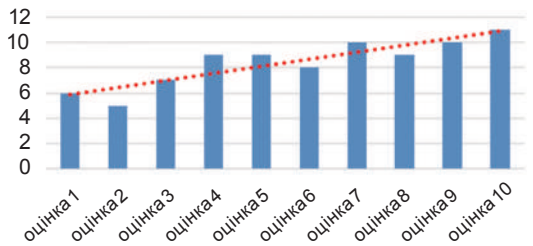


Рис. 10. Приклад використання лінійної лінії тренду

Для значень ряду даних, які різко підвищуються або знижуються, або можуть потім стабілізуватися, використовують не зазначені, а інші типи лінії тренду.

Для значень швидкості автомобіля, який почав рухатися (показники швидкості для рівноприскореного руху), підійде степенева лінія тренду.

Показники кількості популяції деяких тварин у замкненому просторі наближаються до логарифмічного графіка — користуються логарифмічною лінією тренду (кількість спочатку зростає, потім стабілізується).

Якщо швидкість зміни даних безупинно підвищується — користуються експоненціальною лінією тренду (наприклад: показники радіоактивного Вуглецю-14 залежно від віку органічного об'єкта).

Під час коливання значень даних застосовують змінне середнє (наприклад, коливання курсу валют — на найближчі дні неможливо здійснити прогноз).

Точність вибору типу лінії тренду описується спеціальним показником — числом від 0 до 1, яке відображає ступінь відповідності очікуваних значень лінії тренду фактичним даним (а математично: ступінь апроксимації). Кращим вважають показник, наближений до значення 1. Якщо цей показник має невелике значення (менше за 0,5–0,7), краще застосувати інший тип лінії тренду, поки не буде знайдено найбільше наближений тип.

Є такі типи лінії тренду: лінійна, поліноміальна, логарифмічна, степенева, експонентна, змінне середнє.

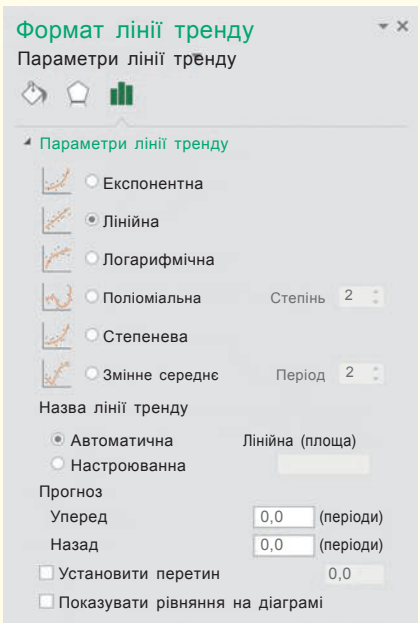


Рис. 11. Вікно **Формат лінії тренду**. До алгоритму на с. 176

Розглянемо *алгоритм додавання лінії тренду* до будь-якого ряду даних на діаграмі чи графіку.

Крок 1	Виділіть потрібний ряд даних.
Крок 2	Викличте контекстне меню та виберіть команду Додати лінію тренду.
Крок 3	У вікні Формат лінії тренду (рис. 11) виберіть тип лінії тренду та формат її зображення; за бажанням поставте галочку для запуску команди розрахунку величини достовірності апроксимації R^2.

Команда додавання лінії тренду є також у меню Діаграма — меню розташоване поруч із правим верхнім кутом діаграми під знаком «+».

Запитання для перевірки знань

1. Для чого використовують інфографіку?
2. Поясніть, у яких випадках зручно для ряду даних створити гістограму, а в яких — кругову діаграму. Наведіть приклади.
3. Як ви розумієте поняття лінії тренду?
4. На що вказує лінія тренду?
5. Як додати на діаграму лінію тренду?
6. За яким параметром визначають відповідність лінії тренду значенням діаграми?

Завдання для самостійного виконання

1. Знайдіть у підручнику з географії інтернеті та порівняйте чисельність населення країн або регіонів, площі, приросту населення тощо. Побудуйте діаграму порівняння даних.
2. Заповніть діапазон **A1:E7** даними таблиці (рис. 12).
3. Додайте стовпець для обчислення щільності населення за формулою

   = Населення/Площа.

   (цей стовпець потрібен для візуалізації даних таблиці, оскільки різниця в даних щодо населення в 100 разів, а щодо щільності — не така велика).
4. Побудуйте діаграму для різних регіонів:

1) виділіть діапазон **E2:E7**;

2) перейдіть до побудови точкового графіка й виберіть його вигляд **Об'ємна бульбашкова діаграма**;

3) налаштуйте параметри діаграми рис. 13.

5. Самостійно додайте до діаграми новий ряд — щільність населення.

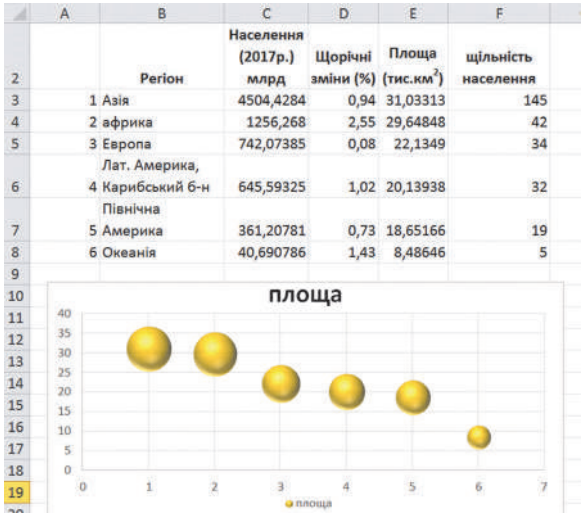


Рис. 12. Опрацювання завдання з географії у MS Excel

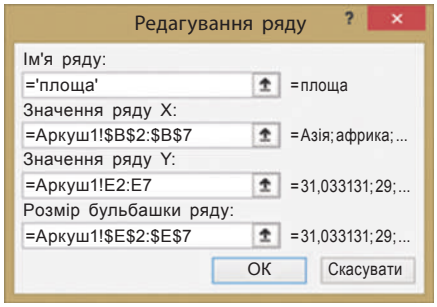


Рис. 13. Налаштування бульбашкової діаграми

3.7. Розв'язування задач із різних предметних галузей. Табличний процесор як засіб для фінансових розрахунків

Пригадайте, як можна розв'язати рівняння графічно. Як побудувати графік у MS Excel? Як викликати контекстне меню діаграми?

Розглянемо можливості MS Excel для розв'язування задач із різних галузей. Пригадайте з уроків математики, як графічно розв'язати систему рівнянь.

- **Графічне розв'язування системи рівнянь**

Приклад 1.

Розглянемо систему двох лінійних рівнянь:

наприклад, $\begin{cases} 3x + 2y = 5; \\ 2x - y = 8. \end{cases}$

Якщо побудуємо графік кожного рівняння, то на їх перетині отримаємо розв'язок (рис. 1).

Перетворимо рівняння у вигляд, зручний для побудови графіків: $y = (5 - 3x) : 2$; $y = 2x - 8$ і виконаємо такі дії.

1. У стовпець A вводимо значення аргументів функцій (x).
2. У стовпцях B і C обчислюємо значення кожної функції відповідно.
3. Виділяємо діапазон зі значеннями y-ків.
4. Переходимо до стрічки ВСТАВЛЕННЯ, до групи Діаграми.
5. Вибираємо тип діаграми — Точкова.
6. Для правильної та зрозумілої побудови графіка переходимо до контекстного меню діаграми.
7. Вибираємо команду Вибрати дані й у вікні, що відкриється, виділяємо ряд і для нього вибираємо команду Редагувати.
8. Заповнюємо поля для кожного ряду за зразком (рис. 2).

Як бачимо, графіки на рис. 1 перетинаються у точці (3; –2). Отже, розв'язком є $x = 3$; $y = -2$.

Якщо виділити діаграму, з'являється можливість перейти до додаткової стрічки ЗНАРЯДДЯ ДЛЯ ДІАГРАМ — Конструктор. Першою групою команд є Додати елемент діаграми. Якщо на діаграмі не вистачає ліній сітки, користуються саме цією групою. Також можна налаштувати вигляд діаграм за запропонованими зразками.

	A	B	C
1	x	y=(5-3x)/2	y=2x-8
2	-5	10	-18
3	-4	8,5	-16
4	-3	7	-14
5	-2	5,5	-12
6	-1	4	-10
7	0	2,5	-8
8	1	1	-6
9	2	-0,5	-4
10	3	-2	-2
11	4	-3,5	0
12	5	-5	2
13			
14			

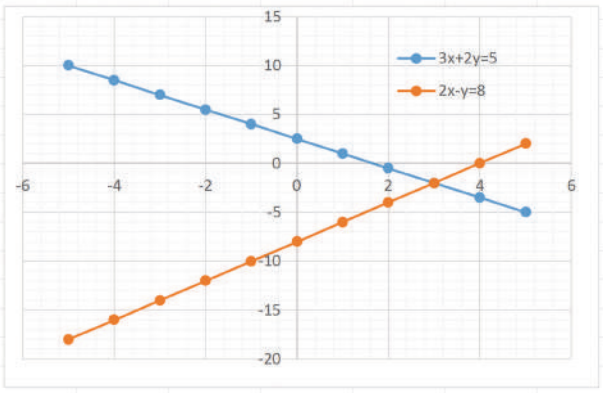


Рис. 1. Приклад графічного розв'язання системи лінійних рівнянь

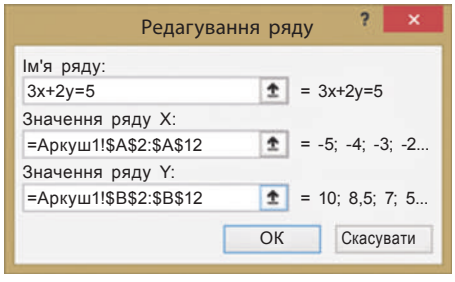


Рис. 2. Вікно редагування ряду даних діаграми

Приклад 1

Змінемо друге рівняння в системі на $2x^2 - y = 8$ та самостійно знайдіть розв'язання графічно. Перевіремо правильність розв'язання за рис. 3. Поміркуємо, чому ми отримали два розв'язки.

	A	B	C
1	x	y=(5-3x)/2	y=2x*x+8
2	-5	10	42
3	-4	8,5	24
4	-3	7	10
5	-2	5,5	0
6	-1	4	-6
7	0	2,5	-8
8	1	1	-6
9	2	-0,5	0
10	3	-2	10
11	4	-3,5	24
12	5	-5	42
13			
14			

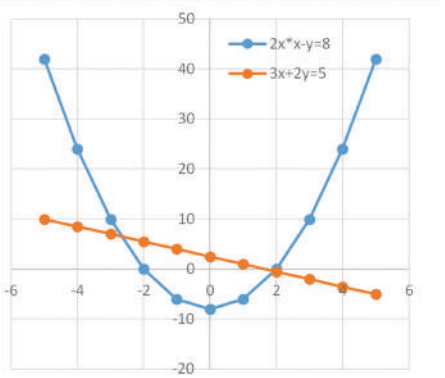


Рис. 3. Приклад графічного розв'язання системи рівнянь

Хід міркувань під час фінансових розрахунків

- У разі кредитування в банку виникають такі питання: скільки років необхідно повертати кредит; скільки грошових одиниць виплачено за користування кредитом; яку суму можна взяти в кредит, щоб повернути її за визначений термін
- У разі внесення вкладу в банк виникають такі питання: якою буде сума через визначений час; на який час слід вкласти гроші для отримання певної суми; який вклад необхідно зробити для отримання певної суми через визначений термін

Приклад 2.

Власник кладе в банк депозит у розмірі 5000 грн, ставка 20% річних. Простий відсоток приноситиме прибуток у розмірі 1000 грн щороку, незалежно від того, яка сума накопичилася на рахунку за цей час і незалежно від того, чи залишає власник відсотки в банку, чи регулярно їх знімає.

- **Розв'язування фінансових задач засобами MS Excel**

Функції для розв'язування фінансових задач передбачено в усіх пакетах для фінансових розрахунків. Ознайомимося з ними на прикладі засобів MS Excel.

Щоб використовувати фінансові функції у MS Excel для розрахунків величин, познайомимося з відповідного термінологією (у дужках термінологія MS Excel).

- **Відсоткова ставка** (RATE (Ставка)) — відносна величина доходу за фіксований інтервал часу, вимірюється як у відсотках, так і числом, може нараховуватися як добова, місячна, річна тощо.

Нарахування відсоткової ставки здійснюється по-різному, тому розрізняють прості й складні відсотки.

- Під **простим відсотком** розуміють прибуток, який нараховується тільки на початкову суму за кожен певний проміжок часу (приклад 2).
- **Складний відсоток** — складна форма нарахування відсотків за вкладом.
- Прибуток нараховується на суму разом із нарахованими відсотками, яка в цей момент наявна на рахунку. Тобто після закінчення кожного періоду сума, на яку нараховується прибуток, відсотково збільшується.

  Повернемося до прикладу 2: у перший рік відсотки будуть нараховуватися з 5000 грн і прибуток складе 1000 грн. У наступному році відсоток вже нараховуватиметься з 6000 грн (5000 + 1000) і так щороку, поки вкладник не припинить дію депозиту.
- **Період нарахування** — інтервал часу для відсоткової ставки (може бути день, місяць або рік).
- **Кількість періодів** (NPTR (КПер)) визначає загальну кількість періодів виплат кредиту, кожен тривалістю добу, місяць, рік тощо.
- **Періодична виплата** (PMT (Плт)) — платіж, який виплачує кожного періоду (це від'ємне число) або сума, яку отримує клієнт кожного періоду (це додатне число).

- **Капіталізація відсотків** — приєднання нарахованих відсотків до основної суми; нарощення — збільшення початкової суми у зв'язку з капіталізацією (складні відсотки).

Для розрахунків величин раніше у групі фінансових функцій MS Excel є такі функції:

Назва функції (англ.)	**Назва функції** (рос.)	**Пояснення**	**Аргументи функції**
RATE	СТАВКА	Відсоткова ставка за один період	кпер; плт; пс; [бс]; [тип]
VF	БС	Майбутня вартість інвестицій	ставка; кпер; плт; [пс]; [тип]
NPER	КПер	Кількість періодів	ставка; плт; пс; [бс]; [тип]
PV	ПС	Вартість інвестиції на поточний момент (початковий внесок)	ставка; кпер; плт; [бс]; [тип]
PMT	Плт	Величина виплати за один період	ставка; кпер; пс; [бс]; тип]

Як бачимо, усі величини взаємопов'язані, кожна розраховується через значення величин з умови. Але повинна справджуватися рівність:

$$PMT * NPERF + PV + VF = 0$$
або Плт * КПер + ПС + БС = 0.

Якщо якийсь аргумент можна не вказувати, то в таблиці він міститься у квадратних дужках. Тип у деяких функціях має значення 0 або 1, визначає час виплати: 0 (або якщо значення не вказано) — кінці періоду, 1 — на початку.

Запитання для перевірки знань

1. Як використати можливості MS Excel для розв'язування рівняння графічним способом?
2. Як у MS Excel обчислити кількість і вартість шпалер для обклеювання кімнати розмірами 4×6,5×2,7 м? Ширина рулонів 55 см, довжина — 10 м.
3. Яка різниця між простими та складними відсотками?
4. Назвіть параметри фінансових розрахунків.
5. Наведіть приклад фінансової задачі, у якій використовують складний відсоток.
6. Що б ви порадили особі, яка з нового року відкрила депозит і не знає, коли додавати гроші: на початку чи наприкінці року?

Відсотки нараховують щорічно. Як би ви аргументували свою відповідь?

Завдання для самостійного виконання

Для ознайомлення з наведеними прикладами перейдіть у MS Excel, заповніть таблиці початковими даними умови та скористайтеся фінансовими функціями для розв'язування задач. У кожному прикладі наведено підказки.

1. Для прикладу з розрахунком, яка сума виявиться на рахунку через три роки, якщо кожен рік роблять внесок на рахунок 10 000 грн і банк дає 10 % річних (раз на рік), функція VF (БС) матиме такий вигляд:

VF(0,1;3;-10000;;1) для внесків на початку року, результат **36 410**

VF(0,1;3;-10000;;0) для внесків наприкінці року, результат **33 744**.

• Якщо внесли 10 000 грн і щороку додають по 500 грн, функція матиме такий вигляд:

VF(0,1;3;–500;–10000;;0)
або **VF(0,1;3;–500;–10000;;1)**.

• Якщо банк нараховує відсотки раз на квартал, то перший аргумент функції (ставка) буде **0,1/4** (0,1 — це 10 % річних, а ділимо на 4, оскільки квартал — четверта частина року).

2 Визначте суму початкового внеску, який забезпечує клієнтові щорічні виплати в сумі 10 млн грн протягом 5 років (складні відсотки, 20 % річних).

PV(0,20;5;–10 000 000;;0)

PV(0,20;5;–10 000 000;;1)

3 У борг беруть 300 000 грн під річну ставку 6 %. Зза рік виплачується по 34 000 грн. Скільки років триватимуть ці виплати?

IVPer(6 %;–34 000;300 000) = 13 PV

3.8. Електронна таблиця як засіб подання відомостей про однотипні об'єкти. Операції з однотабличною базою даних

Які типи функцій у MS Excel ви знаєте?

Ви вже створювали в MS Excel таблиці, вносили дані й розподіляли ряди даних по рядках і стовпцях таблиці на свій розсуд. Зазвичай під час опрацювання дані, які мають багато властивостей, вносять у таблицю як список.

Список складається з рядків, які в Excel називають **записами,** стовпці списку — **полями**. Поля містять дані одного типу.

Список в Excel — таблиця, оформлення якої відповідає певним вимогам.

Ознайомимося з вимогами до списку:

- верхній рядок списку, заголовок, містить мітки (імена) відповідних полів, а його формат (шрифт, колір фону тощо) має відрізнятися від формату записів;
- усередині списку не повинно бути порожніх записів і полів, якими список відокремлюється від іншої частини робочого аркуша;
- не рекомендується на робочому аркуші розміщувати будь-що, крім списку, а робочий аркуш рекомендується іменувати назвою списку.

Списки можна сортувати і фільтрувати, до списків можна застосовувати функції баз даних.

- **Сортування списків**

Поле, за яким здійснюється сортування, називають **ключовим полем**, або **ключем сортування**.

Сортування списку — розташування його записів у певному порядку.

Записи можна розташовувати таким чином:

- у порядку зростання (зменшення) значень числових полів;
- в алфавітному (або зворотному алфавітному) порядку значень текстових полів;
- у хронологічному порядку полів типу Дата і Час.

Можливості сортування реалізуються за допомогою кнопок Сортування за зростанням і Сортування за спаданням панелі інструментів ОСНОВНЕ або скористатися командою меню ДАНІ — група Сортування і фільтр — Сортування, яка дозволяє

сортувати список за один прийом максимум за трьома полями (первинний ключ, вторинний тощо).

У разі потреби сортування можна провести більш ніж за трьома полями. У такому випадку список сортується послідовно: за первинним ключем, потім відсортовані списки в рамках первинних ключів відсортовуються за вторинним ключем і т. д.

Сортувати можна й частину списку, попередньо її виділивши.

Приклад 1.

Для проведення ІІІ етапу олімпіади з інформатики для 9–11 класів підготували список учасників. Завдання для всіх учасників однакові. Список містить поля: ПІБ, Клас, Область, Населений пункт, Освітній заклад, Бали за кожну задачу, Підсумковий бал.

Прізвища в списку розташовані так, як учні розмістилися по приміщеннях. Журі впорядковує список за класами, потім — за балами: від більших до менших. Одразу видно, скільки учасників від кожної паралелі й скільки буде в них переможців (50 %). Далі в списку додадуть поле Диплом. Адміністрація області впорядкує список за областями, населеним пунктом, щоб бачити, як виступили їх вихованці.

- **Фільтрування списків**

Фільтрування списку — це процес, у результаті якого для перегляду доступні записи, які задовольняють критерії фільтрації. Решта записів приховані.

Повернемося до прикладу 1: адміністрація району переглядатиме дані про учасників олімпіади тільки свого району, решту даних приховає, застосувавши фільтр.

У MS Excel команди фільтрування містяться на стрічці ДАНІ в групі Сортування і фільтр. Командою Фільтр біля назви кожного поля можна встановити розкривний список, який містить значення цього поля. Щоб вибрати значення для перегляду, потрібно встановити умови фільтрування шляхом скасуванням або проставлянням прапорця біля значення.

У зазначеному списку також можна вибрати команду для встановлення умов фільтрування. Для відфільтрованих значень можна вибрати команду сортування (рис. 1).

Розрізняють простий і розширений фільтри.

За допомогою команди Фільтр здійснюється фільтрування тільки за умовою для одного поля. Такий фільтр називають простим. Відфільтрований список теж можна фільтрувати. Так створюється ієрархічне фільтрування.

Щоб зняти режим фільтрування і перегляду всього списку, слід повторно скористатися командою Фільтр або командою Очистити.

Розширений фільтр дозволяє виконувати умови фільтрування практично будь-якої складності:

- створювати критерії з умовами за кількома полями з кількома умовами;
- створювати обчислювані критерії;
- переміщувати копію отриманої внаслідок фільтрації вибірки в інше місце книги MS Excel.

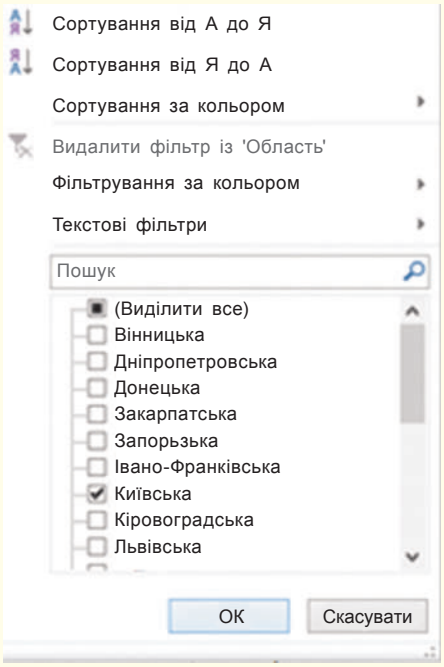


Рис. 1. Приклад списку фільтрування

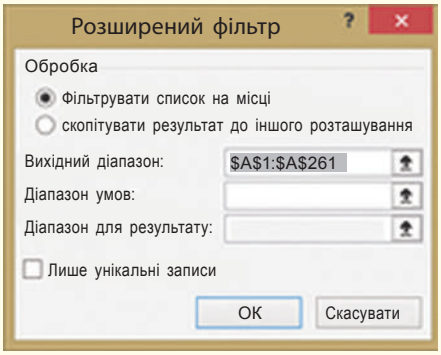


Рис. 2. Вікно налаштування розширеного фільтру

Розширений фільтр можна викликати командою ДАНІ — група Сортування і фільтр — Додатково. Відкривається вікно введення діапазонів списку та умов фільтрування (рис. 2).

Як бачимо, умови (критерії) фільтрування потрібно розмістити в діапазоні, тобто спочатку підготовити клітинки з умовами фільтрування, а потім викликати розширений фільтр.

Створюючи діапазон критеріїв, необхідно пам'ятати, що:

- діапазон умов має складатися не менше ніж із двох рядків (перший рядок — заголовки, які рекомендується просто копіювати із заголовків стовпців списку, наступні — відповідні критерії);
- якщо умови розташовуються в одному рядку, то це означає одночасність їх виконання, тобто вважається, що між ними поставлена логічна операція І;
- для істинності критерію, що складається з умов, розташованих у різних рядках, має виконатися хоча б один із них, тобто вважається, що вони з'єднані логічною операцією АБО;
- інтервал критеріїв повинен розташовуватися вище або нижче списку, або в іншому місці, або на іншому аркуші; критерії від списку мають відокремлюватися хоча б одним порожнім рядком, якщо його розміщено вище або нижче списку;
- в інтервалі критеріїв не повинно бути порожніх рядків.

Приклад 2.

Повернемося до прикладу 1. Потрібно переглянути, як виступають учнів 10 класу з Полтавської області та учасники зі Львівської області. На рис. 3 показано фрагмент таблиці. Над списком вставлено кілька рядків, у яких продубльовано заголовки полів і введено критерії.

У вікні розширеного списку в поле Вихідний діапазон вводимо діапазон розміщення даних таблиці, а в Діапазон умов — C1:D3, налаштуємо перегляд відфільтрованого списку.

	B	C	D	E	F
	Прізвище, ім'я, по батькові	**Клас**	**Область**	**Місто**	**Місце**
		10	Полтавська		
			Львівська		
	Прізвище, ім'я, по батькові	**Клас**	**Область**	**Місто**	**Місце**

Рис. 3. Приклад уведення умов для аналізу результатів олімпіади

- **Опрацювання списку за допомогою функцій MS Excel**

Для аналізу списків часто використовують такі функції:

Назва функції (англ.)	**Назва функції** (рос.)	**Опис**
COUNTIF (інтервал; критерій)	СЧЁТЕСЛИ (інтервал; критерій)	Повертає кількість значень в інтервалі, які відповідають критеріям.
SUMIF (інтервал; критерій; інтервал додавання)	СУММЕСЛИ (інтервал; критерій; інтервал додавання)	Повертає суму значень у клітинках з інтервалу, які відповідають критеріям.

Оскільки список має такі самі властивості, як і таблиця БД, то функції опрацювання даних списку містяться в групі

База даних. Вибір групи, функції групи слід викликати через команду Вставити функцію стрічки ФОРМУЛИ (рис. 4):

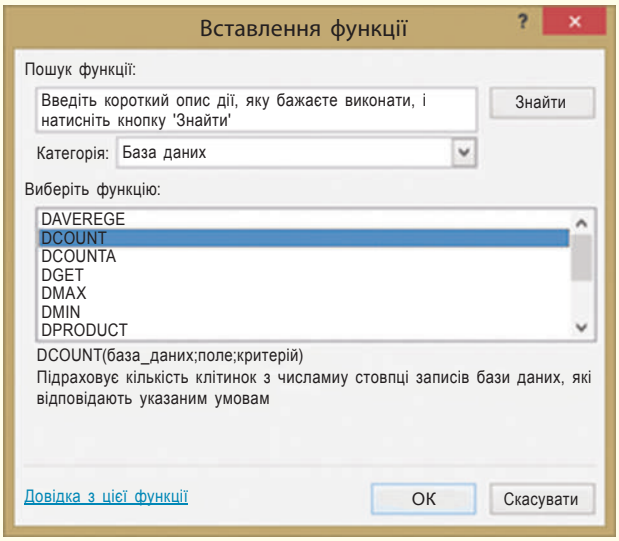


Рис. 4. Вікно вставлення функцій

Завдяки критеріям функції опрацьовують тільки ті записи, значення яких відповідають критеріям, а потім здійснюють операції зі вказаними полями.

Найчастіше використовують такі функції:

- DAVERAGE (база_даних; поле; критерій) або (ДСРЗНАЧ) (база_даних; поле; критерій) — повертає середнє значення полів записів БД, які задовольняють критерії.
- DCOUNT (база_даних; поле; критерій) або (БСЧЕТ) (база_даних; поле; критерій) — повертає кількість числових комірок у полях записів БД, які відповідають заданому критерію.
- DMAX/DMIN (база_даних; поле; критерій) або (ДМАКС/ДМІН) (база_даних; поле; критерій) — повертає максимальне/мінімальне значення поля серед записів БД, які задовольняють критерії.
- COUNTIF (інтервал; критерій) або (СЧЕТЕСЛИ) (інтервал; критерій) — повертає кількість значень в інтервалі, які відповідають критеріям.

Функції для аналізу списків однотабличних БД мають такі аргументи (база_даних; поле; критерій).

Правила звернення до функцій БД:

- перший аргумент задає весь список із записом заголовку полів, а не окремий запис;
- другим аргументом може бути заголовок поля у вигляді текстової константи (ім'я поля) або порядковий номер поля в списку;
- третій аргумент задає інтервал критеріїв аналогічно інтервалу критеріїв розширеного фільтра.

Запитання для перевірки знань

1. Які таблиці в MS Excel називають списком?
2. Як у MS Excel упорядкувати дані за кількома параметрами?
3. Що відбувається з даними списку внаслідок фільтрування?
4. Яка різниця між простим і розширеним фільтром? Наведіть приклади.
5. Наведіть приклади використання функцій бази даних.
6. Чи можна опрацювати дані списку без використання функцій бази даних?

Виконайте тестове завдання до розділу 3 з автоматичною перевіркою результату на сайті interactive.ranok.com.ua

Розділ 4. ЕЛЕКТРОННІ ПУБЛІКАЦІЇ

4.1. Багатосторінкові текстові документи. Налаштування параметрів сторінок. Розділи документа

Назвіть можливості текстового процесора Word щодо форматування символів, абзаців, сторінок.

У процесі опанування можливостей текстового процесора Word ви вже створювали невеликі за розміром документи. Але реферати, роботи для подання на захист науково-дослідницьких робіт МАН України, курсові роботи тощо містять велику кількість сторінок, які потребують оформлення за певними правилами. Пригадаємо, які властивості має сторінка як об'єкт текстового документа (рис. 1).

Під час форматування сторінки потрібно враховувати

- вигляд сторінки (альбомний або книжковий)
- розмір сторінки
- поля відступів між текстом і краєм сторінки
- наявність або відсутність переносів тексту
- розташування тексту в одній або кількох колонках
- колонтитули (верхні, нижні)

• Налаштування параметрів сторінок

Для налаштування параметрів сторінок слід звернутися до команд групи Параметри сторінок стрічки МАКЕТ — буде запропоновано готові зразки. Можна також самостійно встановити параметри у вікні Параметри сторінки (рис. 2).

Рис. 1. Сторінка документа

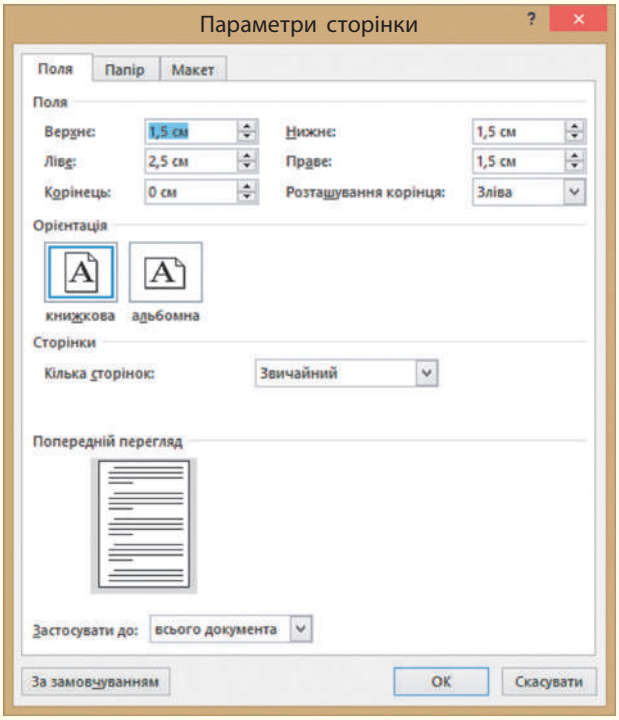


Рис. 2. Вікно **Параметри сторінки**

Далі розглянемо докладніше властивості сторінки та способи їх налаштування.

Поля — це відстань, на якій розташовується текст на сторінці від країв сторінки: указують значення (зазвичай у сантиметрах) для верхнього, нижнього, лівого та правого полів.

На рис. 3 подано типи полів. Перед тим як задати розміри полів, слід визначити, до яких сторінок встановлюються поля: для всіх чи тільки від поточної до останньої (список Застосувати).

У готовому документі сторінки можуть скріплюватися. Щоб передбачити місце для скріплення, слід вибрати команду Розташування корінця й зазначити розмір смуги корінця, а також місце розташування (зліва або зверху).

Орієнтація — це спосіб розміщення сторінки. Розрізняють *книжкову* (висота сторінки більша за її ширину) й *альбомну* (висота сторінки менша за її ширину) орієнтацію.

Досить часто виникає потреба у встановленні окремої орієнтації для деяких сторінок, наприклад, для сторінок, які містять таблиці, схеми чи інші зображення. Такі сторінки краще подавати з альбомною орієнтацією.

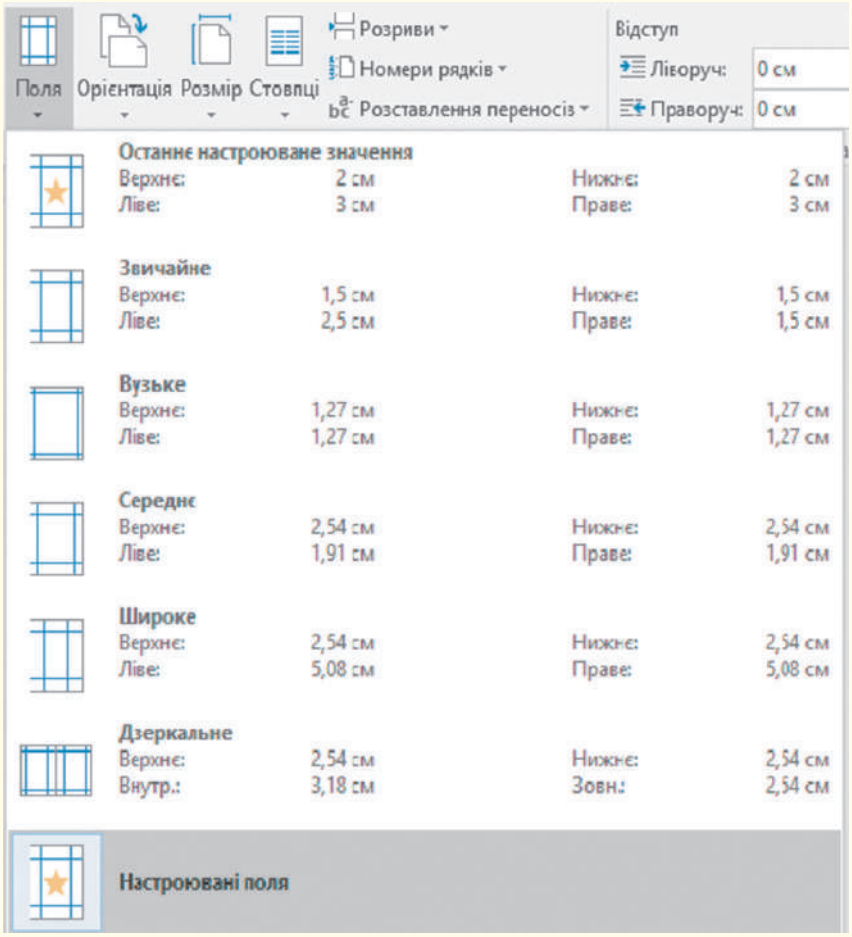


Рис. 3. Вікно вибору типу полів сторінок

Розмір — це висота й ширина сторінки документа. Аркуші паперу мають стандартизацію за розмірами. За потреби користувач може сам установити розмір аркушів для свого документа — це передбачено командою Інші розміри.

Аркуші паперу мають стандартизацію за розмірами: аркуш А4 є прямокутником зі сторонами 21 і 29,7 см. Аркуш А3 удвічі більший (42 × 29,7), а А5 — удвічі менший (21 × 14,8).

• Розділи в документі

Якщо змінити хоча б один параметр складової документа, визначається новий розділ (не плутайте з розділом, що є змістовою частиною документа, наприклад, параграф).

Уявимо, що на певній сторінці документа текст розташовано у дві колонки, а решта параметрів — однакові. Такий документ складається з трьох розділів: розділ до тексту, розташованого в дві колонки, розділ із текстом у дві колонки і решта тексту до кінця документа. Його можна поділити на розділи, установивши розриви розділів.

Розділ — це частина документа з одним і тим самим значенням параметрів форматування.

Види розривів розділів
- наступна сторінка
- поточна сторінка
- парна сторінка
- непарна сторінка

Розрив являє собою недрукований символ і зазначає місце в документі, до якого буде застосовано форматування.

У програмі Word передбачено вилучення розриву розділу. Якщо увімкнути режим перегляду прихованих символів, то видно, що розрив розділу подано як пунктирну лінію з текстом **Розрив розділу** (вид розриву).

Щоб вилучити недрукований символ, слід на початку поставити тестовий вказівник і натиснути клавішу Delete.

До сторінок можна застосувати різні види розривів (рис. 4).

Різні види розривів застосовуються в підручниках та посібниках: нові тематичні розділи, зазвичай починаються з нової сторінки.

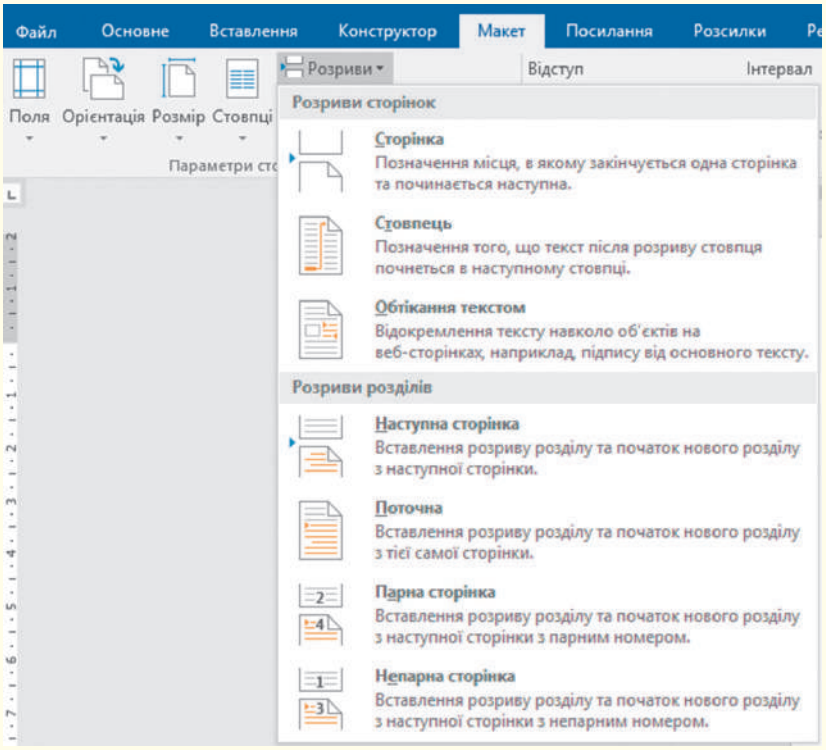


Рис. 4. Види розривів сторінок та розділів

Команду Наступна сторінка продубльовано командою ВСТАВЛЕННЯ — Розрив сторінки, її використовують для розміщення тексту після вказівника з нової сторінки.

Запитання для перевірки знань

1. Назвіть параметри форматування аркуша в текстовому документі.
2. Як викликати вікно Параметри сторінок?
3. Поясніть термін «розрив сторінки».
4. Поясніть сутність динамічної типізації змінних.
5. Що таке «розділ документа» з точки зору форматування?
6. Які види розривів розділів пропонується у Word?

4.2. Колонтитули

Як вставити нумерацію сторінок у документ Word?

У багатосторінкових текстових документах використовують колонтитули, у яких зазначають назву видання, назву розділів, номери сторінок та ін.

Колонтитул — (від франц. *colonne* — стовпець і латин. *titulus* — напис, заголовок) — напис, який розміщують у верхньому або нижньому полі кожної сторінки документа.

Наявність колонтитулів полегшує пошук необхідних розділів під час перегляду документа.

Відповідно до розміщення на сторінці розрізняють *верхні* та *нижні* колонтитули.

Розглянемо *алгоритм додавання колонтитула на сторінку*.

Крок 1	Викличте стрічкове меню **ВСТАВЛЕННЯ**
Крок 2	Виберіть команди з групи **Колонтитули**. Текстовий вказівник з'явиться в області колонтитула, основний текст буде недоступний для роботи з ним, з'явиться тимчасове стрічкове меню **ЗНАРЯДДЯ ДЛЯ КОЛОНТИТУЛІВ — КОНСТРУКТОР**
Крок 3	Уведіть текст у потрібне поле
Крок 4	Щоб повернутися у звичайний режим, достатньо розмістити текстовий вказівник за межами колонтитула й клацнути ЛКМ або скористатися командою (червоний хрестик) на стрічці **ЗНАРЯДДЯ ДЛЯ КОЛОНТИТУЛІВ — КОНСТРУКТОР**

Використання колонтитулів як номерів сторінок

- номерами сторінок не обов'язково числа
- номери сторінок можна як вставляти, так і вилучати:

— номери з кількох або усіх сторінок документа
— номер на першій сторінці
— номер із будь-якої сторінки

Для додавання номера сторінки або зміни його вигляду (рис. 1) слід скористатися командами вікна Формат номера сторінки зі списку Номер сторінки стрічки ВСТАВЛЕННЯ або із стрічки ЗНАРЯДДЯ ДЛЯ КОЛОНТИТУЛІВ — КОНСТРУКТОР, якщо ви вже працюєте з колонтитулами.

Це вікно містить команди встановлення вигляду колонтитулів (числа, літери, можливість використовувати в нумерації сторінок нумерацію розділів тощо), установлення початкових номерів для сторінок (не обов'язково нумерувати з першої, наприклад, якщо розділи розташовані в різних файлах).

Для вилучення колонтитулів призначена команда Видалити верхній (нижній) колонтитул: необхідно перейти до області колонтитула, вибрати список Верній (Нижній) колонтитул і запустити команду Видалити верхній (Нижній) колонтитул. Номери сторінок будуть вилучені.

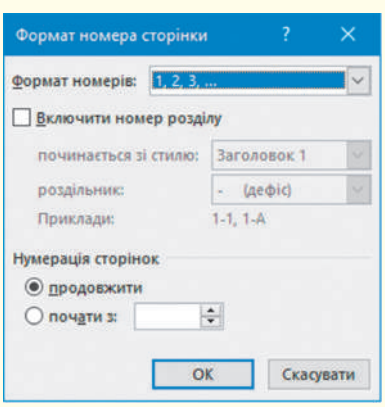


Рис. 1. Вікно **Формат номера сторінки**

Сторінки з нумерацією і без мають належати різним розділам, адже вони вже відрізняються параметром **номер сторінки**.

Для роботи з колонтитулами передбачено команди вилучення нумерації тільки із деяких сторінок.

Зазвичай на першій сторінці номер не проставляють. Щоб колонтитул не відображався на першій сторінці, потрібно встановити вказівник миші в область колонтитула і двічі клацнути ЛКМ — з'явиться стрічка конструктора колонтитулів. Далі слід встановити прапорець команди Інші для першої сторінки у групі Параметри.

Розглянемо *алгоритм видалення номера не на першій сторінці*.

Крок 1	Установіть перед такою сторінкою розрив розділу (не розрив сторінки!): перейдіть до стрічки **МАКЕТ**, зі списку **Розриви** виберіть команду **Наступна сторінка**
Крок 2	Перейдіть до області колонтитула на потрібній сторінці і двічі клацніть ЛКМ
Крок 3	У стрічці вимкніть команду **Як у попередньому** з групи **Навігація** — використання цієї команди порушить зв'язок між розділами
Крок 4	Зробіть активною команду **Інші** для першої сторінки. В алгоритмі **вилучення номерів на кількох сторінках** пункти приблизно такі самі. Розрив розділу необхідно встановити на початку першої сторінки з тих, що не будуть мати номери й зміниться крок 4
Крок 5	У стрічці активізуйте команду **Верхній колонтитул** (чи нижній — залежить, де проставлені номери сторінок) і зі списку, що розкриється, виберіть команду **Видалити верхній колонтитул** (чи нижній) — будуть видалені номери з усіх сторінок розділу

Розглянемо *алгоритм отримання колонтитулів різного вигляду на різних сторінках*.

Крок 1	На стрічці **КОНСТРУКТОРА** колонтитулів, коли вказівник уже в області колонтитула, є команда **Різні колонтитули для непарних і парних сторінок**. Біля цієї команди проставте прапорець
Крок 2	Перейдіть у область колонтитула непарної, запустіть команду **Верхній** або **Нижній колонтитул**; виберіть на свій розсуд зразок вигляду колонтитула; уведіть напис
Крок 3	Повторіть ці самі дії для парної сторінки
Крок 4	Оскільки Word пропонує різні зразки для різних сторінок, відслідковуйте, які зразки краще вибрати. Поверніться до основного тексту документа

Найбільш використовуваними є такі поля, як номери сторінок, дата, час, підписи під малюнками.

Якщо додавати номери сторінок у наявний колонтитул, то колонтитули замінюються номерами сторінок. У такому випадку процес нумерації сторінок здійснюється вставкою полів.

Нерідко різні організації мають потребу в створенні документів, які містять постійні дані та дані, які змінюються (оновлюються) автоматично. Такі дані проставляються в полях (не плутати з полями сторінки).

Поля — важливі об'єкти документа MS Word, які утримують змінні дані. Значення поля залежить від ключових слів і параметрів.

Поля дають змогу організувати автоматичне оновлення даних у документі, виконувати обчислення, установлювати зв'язки з іншими документами та об'єктами, створювати перехресні посилання.

Для вставлення поля в текст необхідно натиснути сполучення клавіш Ctrl + F9 або вибрати зі стрічки ВСТАВЛЕННЯ команду Експрес блоки — Поле (рис. 2), відкриється вікно вибору полів (рис. 3).

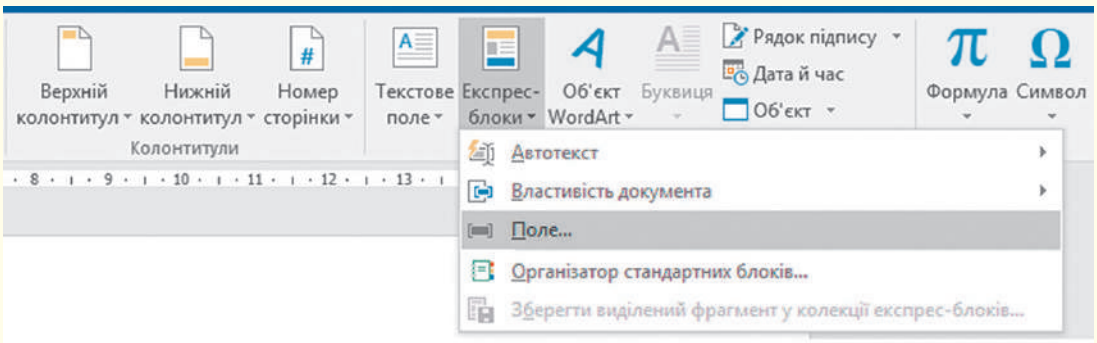


Рис. 2. Група **Експрес блоки**

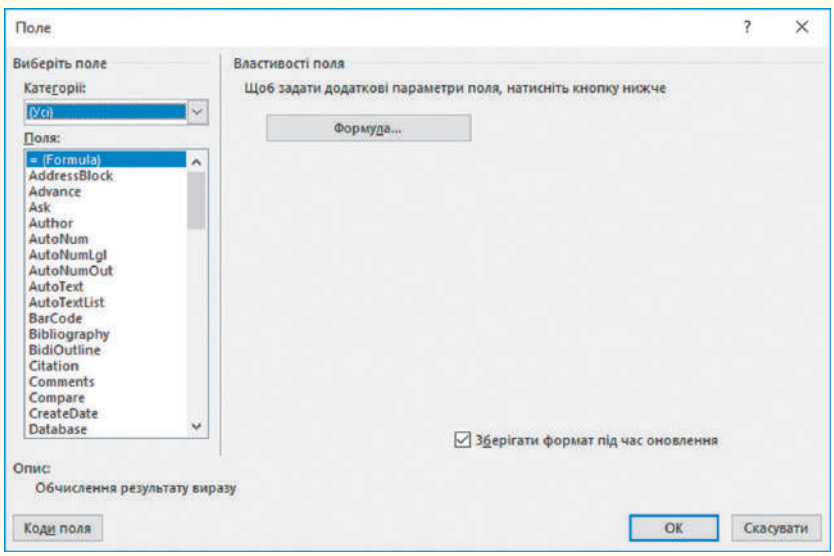


Рис. 3. Вікно **Поле**

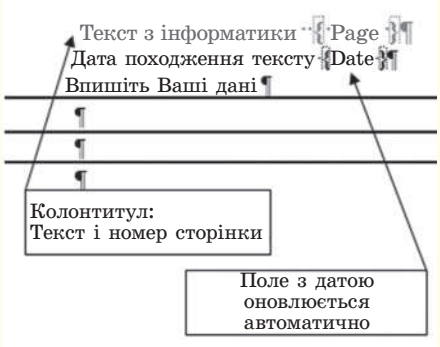
Коди полів відображаються всередині фігурних дужок ({}). Поля працюють як формули в Microsoft Office Excel: код поля відповідає формулі, а значення поля — значенням цієї формули.

Отже, об'єкт Поле може відображатися в документі за кодом (Date, Page тощо) та значенням (конкретна дата або сторінка, як у наведених прикладах).

Якщо на текст поля навести вказівник і клацнути ЛКМ, вмикається затінення — візуальна ознака поля (під час друку документа колір тла на аркуш не виводиться).

Синтаксис поля такий:
{ **ІМ'Я ПОЛЯ** Властивість **Ключі** (необов'язково) }

- Ім'я поля: Практично назва поля, яке вибирається з вікна **Поле**
- Властивість. Деякі поля використовуються з обов'язковим введенням додаткових параметрів. Вони й уточнюють властивість поля.



Щоб *вставити та опрацьовувати поле* **Дата**, необхідно виконати такі дії.

Крок 1	На клавіатурі натисніть сполучення клавіш **Ctrl + F9** — у місці розташування вказівника з'являються фігурні дужки { } з текстовим указівником на затіненому просторі
Крок 2	Уведіть назву поля **Date** (назви полів указують англійською мовою) — після введення з'являється напис **оновлення** (рис. 4)
Крок 3	Активізуйте команду **оновлення** — на місці поля буде встановлено поточну дату. Формат її регулюється операційною системою
Крок 4	Якщо вставити поле **Page** — сторінка, то натискання клавіші **F9** (виклик команди оновлення поля) відгукнеться номером поточної сторінки у місці поля

Рис. 4. Вставлення поля ДАТА

Гарячі клавіші для опрацювання полів

- Ctrl + F9 — вставка порожнього поля
- Alt + Shift + D — вставка поля DATE
- Alt + Shift + T — вставка поля TIME
- Alt + Shift + P — вставка поля PAGE
- F9 — оновлення виділених полів
- Shift + F9 — перемикання коду поля на його значення й навпаки
- Alt + F9 — перемикання кодів усіх полів на їх значення й навпаки

Команди контекстного меню для виділеного поля дозволяють оновлювати поля, передивлятися їх значення або коди та змінювати їх (рис. 5). Затінення полів регулюється командами Файл — Параметри — Додатково — Показувати вміст документа — Вибрати команду зі списку Затіняти поля (команди: Ніколи, Завжди, У разі виділення) (рис. 6).

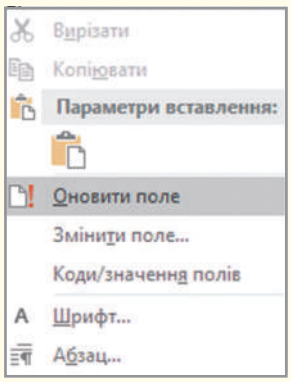


Рис. 5. Контекстне меню поля

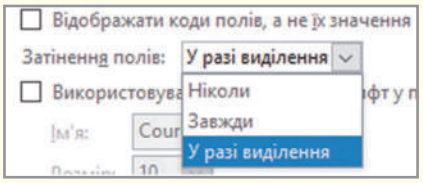


Рис. 6. Налаштування параметрів поля

Додатково ознайомтеся з алгоритмом додавання номерів сторінок за допомогою коду поля сторінки.

1. Перейдіть у область колонтитула — двічі клацніть область верхнього або нижнього колонтитула.
2. Перейдіть до команди ВСТАВЛЕННЯ Експрес блоки — Поле: відкриється вікно Поле.
3. У списку Поля виберіть пункт Page і натисніть кнопку ОК.

Запитання для перевірки знань

1. Що таке колонтитули?
2. Які команди опрацювання колонтитулів ви знаєте?
3. Як прибрати номер на першій сторінці?
4. Як отримати на різних сторінках різні написи в області колонтитулів?
5. Які типи посилань використовуються в документах Word?
6. Які типи розриву розділу пропонується у Word?

4.3. Схема документа

Пригадайте визначення шаблона документа. Як створити шаблон документа у Word? Що є ознакою кінця абзацу?

У текстовому процесорі існують різні режими перегляду документа (рис. 1).

Режими перегляду документа містяться на стрічці Подання в групі Подання. У режимі підготовки документа до друку додається режим попереднього перегляду сторінок.

Режими перегляду документа

- режим читання
- режим розмітки сторінки
- режим веб-документа
- режим структури документа
- режим чернетки

Рис. 1. Різні режими перегляду документа

Структура документа — це схема розміщення складових документа. Під час перегляду документа в режимі структури його вміст показано у вигляді маркованого списку.

Приклад.

Перегляд у режимі структури зазвичай використовують у процесі планування створення багатосторінкового документа та роботи з ним. Особливо це важливо для документа, у створенні якого бере участь авторський колектив. Після того, як автори домовляться про структуру документа, має сенс визначити частини для кожного з них. Таким чином буде відомо, про що йде мова в попередніх розділах, на які знання й уміння потрібно спиратися у певній частині посібника.

Шаблон (від франц. *echantillon*) означає зразок. За готовим зразком (шаблоном) створюють документи певного виду.

Щоб усі частини документа були відформатовані однаково, доцільно перед уведенням тексту створити шаблон. Документ із назвами тематичних розділів, із зазначенням режимів форматування його об'єктів необхідно зберегти у файлі з типом Шаблон Word (документ буде мати розширення .dotx). Таким чином, кожен автор працюватиме зі своєю частиною документа на основі попередньо створеного шаблона, збереже свою роботу командою Зберегти як... із зазначенням типу документа як документ Word.

Розглянемо *алгоритм створення структури*.

Крок 1	Перейдіть до стрічки **Подання,** у групі з таким самим іменем виберіть команду **Структура** — зміниться вигляд документа, а в меню буде показана стрічка **СТРУКТУРА** (рис. 2)
Крок 2	Перейдіть у робочу зону і почніть набирати заголовки розділів, параграфів як окремі абзаци
Крок 3	Під час уведеня назви параграфа розділу зверніться до команди **Рівень** у стрічці та зменшіть його. І так із кожним заголовком — то зменшіть, то збільшіть або залишіть на тому самому рівні. Є і другий варіант — уведіть усі заголовки, потім виберіть кілька і перемістіть їх до іншого рівня
Крок 4	Після створення структури командами стрічки перегляньте документ (якщо він є) або закрийте режим структури

Заголовок — назва стилю структурного елемента документа.

У режимі розмітки сторінки видно, що текст кожного рівня структури оформлено окремим стилем (рис. 2). Зазвичай текст рівня 1 має стиль Заголовок 1, нижчого — відповідно Заголовок 2 і т. д.

Форматування вигляду заголовків здійснюється командами групи Стилі стрічки ОСНОВНЕ.

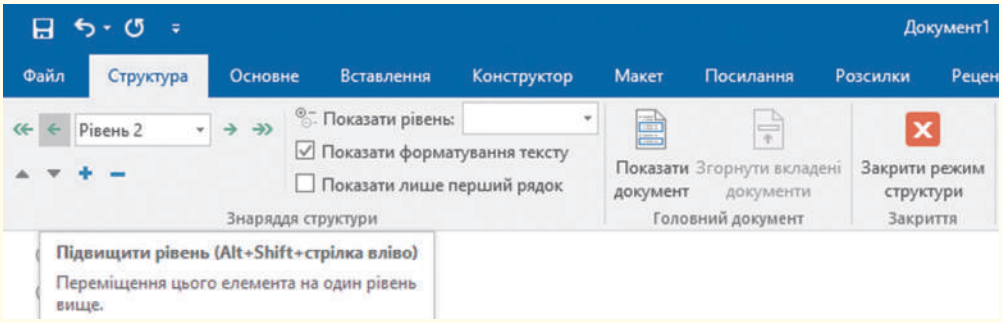


Рис. 2. Команди стрічки **СТРУКТУРА**

Зміст — перелік структурних елементів документа із зазначенням сторінок їх розміщення в документі.

На основі складових структури можна автоматично створити зміст документа командою ПОСИЛАННЯ → → Зміст вибрати зміст. На рис. 3 показано вигляд змісту з ієрархічним поданням заголовків різних рівнів.

Після додавання чи вилучення тексту в документі потрібно виділити зміст і запустити його оновлення: оновити все чи тільки номери сторінок.

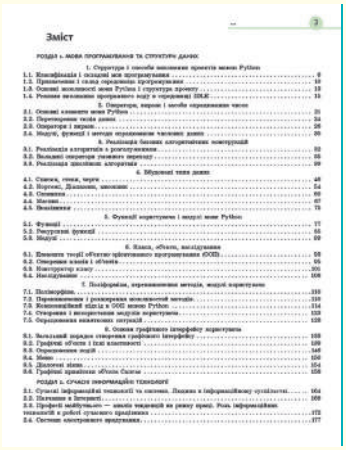


Рис. 3. Зміст із заголовками різних рівнів

Запитання для перевірки знань

1. Що таке структура документа?
2. На основі чого створюють структуру документа?
3. Чим зміст документа відрізняється від його структури?
4. Як в області колонтитула вказувати назву розділу?
5. Чому команда вставлення змісту розташована в стрічці **Посилання**?

Завдання для самостійного виконання

1. Відкрийте Закон України про освіту за посиланням: http://zakon3.rada.gov.ua/laws/show/2145-19.
2. Перейдіть до перегляду документа в режимі **Стисло** — використовуйте кнопки в правому верхньому куті. Як виглядає документ.
3. Перейдіть до перегляду документа в режимі **Розгорнуто** — використовуйте кнопки в правому верхньому куті. Зверніть увагу на вигляд документа.
4. Який із двох режимів дає змогу переглянути структуру документа?

4.4. Використання полів злиття

Проаналізуємо приклад 1. Текст (запрошення для всіх батьків однакове) загальний, а звертання — персональне, на ім'я та по батькові. Кожній особі оформлювати персональне запрошення складно. Щоб автоматизувати цю працю, можна скористатися такою можливістю текстового процесора, як робота з полями злиття.

Щоб не користуватися вікном поля і не прописувати коди поля вручну, можна скористатися стрічкою РОЗСИЛКИ.

Злиття — процес об'єднання основного документа з базою даних для створення серії однотипних документів, зміст яких складають однакові текстові або графічні елементи й унікальні.

Приклад 1.
Як надіслати всім батькам повідомлення про святкування та графік навчання 1 вересня? Якщо в класі 30 учнів, то необхідно створити таку саму чи більшу кількість документів із персональним запрошенням на свято. Документ можна оформити як лист-запрошення або електронний лист-повідомлення.

Приклад 2.

На рис. 1 наведено схему процесу злиття, а також типи документів, які можна в результаті отримати.

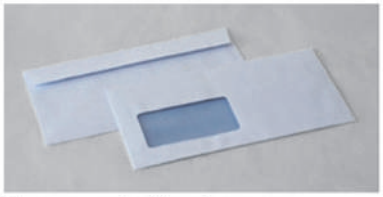
Як бачимо, на конвертах *загальною* ознакою є зворотна адреса, *унікальною* — адреса кожного отримувача. Листи, листівки, запрошення, повідомлення мають однаковий текст, але звернення до одержувачів персональне. Такі документи можуть містити додатковий текст для деяких одержувачів.

Отже, у результаті злиття отримуємо складений документ. Він містить однакові дані (текстові, графічні тощо) для всіх примірників документів та поля підстановки, у яких буде розміщено унікальні дані (у нашому прикладі — звернення до батьків). Його ще називають основним документом. База даних є джерелом даних для заповнення полів основного документа. Укажемо, значення якого поля з бази даних необхідно ввести в місце розташування поля підстановки (тут слово «поле» має різні значення).

Рис. 1. Схематичне зображення процесу злиття

У процесі злиття відбувається об'єднання файла основного документа та файла з джерелом даних — створюється файл розсилки.

Наводимо *алгоритм створення документа розсилки* (рис. 2), побудований на базі кроків, які пропонуються текстовим процесором MS Word.



В документі в блоці адреси:
[ПІПб одержувача]
[Вулиця], б.[будинок], кв.[квартира]
[місто]

Рис. 2. Приклад розсилки однотипних документів

Крок 1	Визначте тип документа розсилки
Крок 2	Визначте, на основі чого буде створено основний документ
Крок 3	Налаштуйте джерела даних
Крок 4	Створіть основний документ
Крок 5	Перегляньте документи розсилки
Крок 6	Завершіть злиття

У MS Word є стрічкове меню РОЗСИЛКИ, команди якого призначено для злиття (рис. 3).

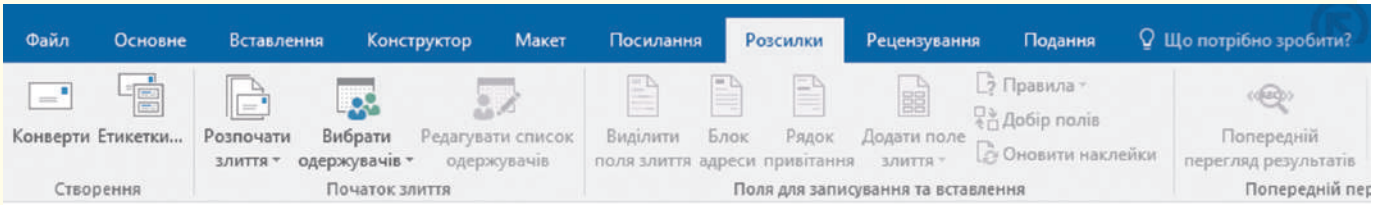


Рис. 3. Вигляд стрічкового меню **РОЗСИЛКИ**

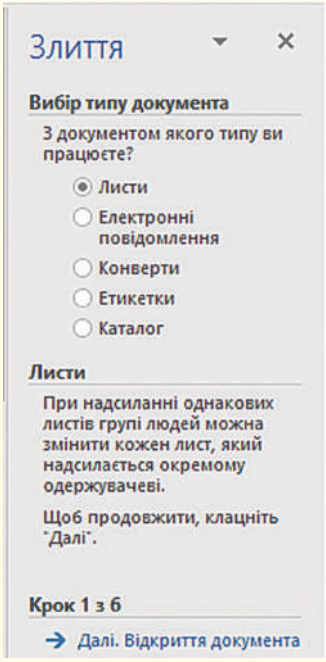


Рис. 5. Вікно області задач **Злиття**

Для злиття зручно користуватися майстром злиття: Розпочати злиття — Покроковий майстер злиття — рис. 4.

Результатом вибору цієї команди є відкриття області задач Злиття (рис. 5), у якій слід вибирати команди відповідного кроку. Область містить команди переміщення до наступного кроку або повернення до попереднього.

Звернемося знову до прикладу 1 (розсилка запрошень) та розглянемо *алгоритм злиття детальніше*.

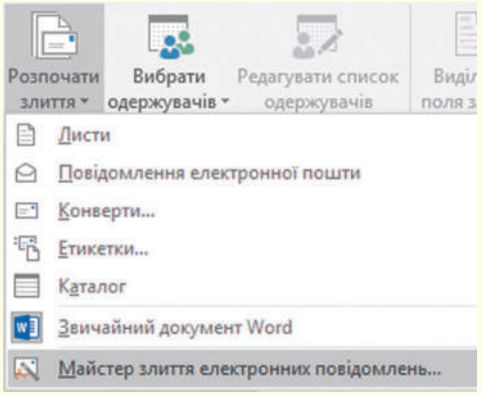


Рис. 4. Режими початку злиття

Крок 1

Виберіть тип документа — наприклад, лист або електронне звернення. Особливості роботи з різними типами файла розсилки розглянемо на наступних кроках

Крок 2

Визначте, на основі чого створити документ із загальними відомостями, поки що без полів з унікальними даними.

- Команду **Поточний документ** використовують, якщо у MS Word спочатку створили файл для розсилки без унікальних значень, а потім звернулися до команд стрічки **РОЗСИЛКИ**.
- Команда **Шаблон** призначена для багаторазового використання текстового документа. Якщо вибирають вказану команду, необхідно запустити команду **Вибір шаблону** для відкриття файла.
- Команда **Існуючий документ** призначена для використання вже збереженого на диску текстового документа. Після активізації цієї команди пропонується, як і в попередньому випадку, виконати команду **Відкрити**, щоб завантажити у вікно Word необхідний файл

Крок 3

Виберіть джерело даних та за потреби відредагуйте його вміст, наприклад, для створення листів має бути набір адресатів із зазначеними адресами.

- Контакти Outlook використовують для створення розсилки електронних листів, але Outlook має бути основною (за замовчуванням) поштовою програмою, установленою на вашому комп'ютері. Тоді переходять до списку контактів і вибирають із нього необхідні.
- Якщо файл із джерелом даних попередньо не створювали, вибирають команди **Створення списку — Створити**. У вікні створення списку адрес отримувачів файлів розсилки пропонується готовий шаблон з уведеними назвами стовпців. Стовпці у списку розсилки і є полями злиття. Користувач сам може додати необхідне поле чи вилучити командою **Настроїти стовпці** (рис. 6). Файл зберігають у форматі файла бази даних

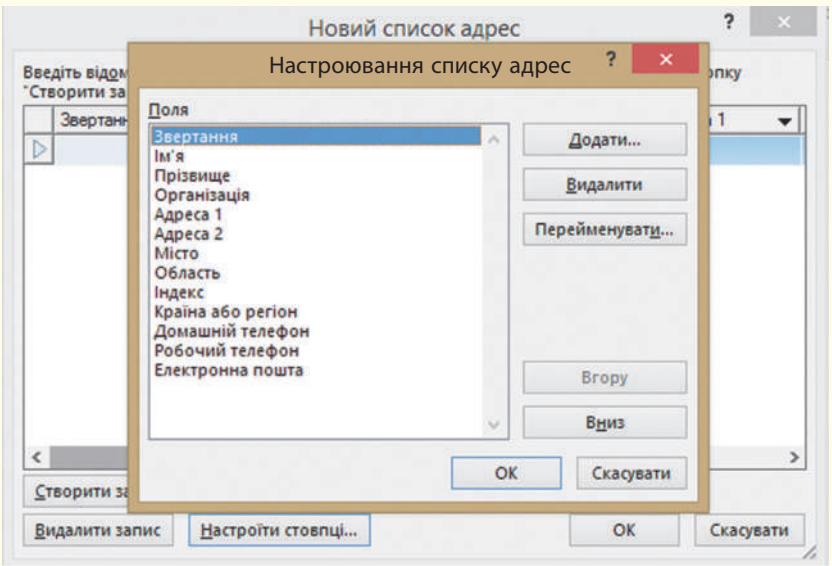


Рис. 6. Створення таблиці одержувачів розсилки

- Команду **Використання списку** запускають, якщо вже є база даних, підготовлена для створення розсилки. Також як джерело даних часто використовують файли електронних таблиць. Таблиця має містити заголовки стовпців із даними. Дані — поштові індекси, телефони — краще відформатувати як текст

Крок 4	• Створіть документ розсилки. Для цього заповніть документ Word унікальними даними. Принцип заповнення полягає у вставленні в документ полів підстановки. У документі текстовий вказівник розміщують у місце для вставлення унікальних даних — у місце вставлення поля підстановки, у вікні **Злиття вибирають команди відповідно до типу документа**: • Для конвертів — **Блок адреси**. • Для додавання електронної марки необхідно, щоб було встановлене відповідне програмне забезпечення. • Вибір команди **Рядок привітання** супроводжується відкриттям вікна з готовими шаблонами рядка. • Команда **Інші елементи** призначена для самостійного вставлення полів підстановки користувачем: у вікні **Вставка поля злиття** перелічені всі поля бази даних. Поле вибирається подвійним клацанням вказівника миші — у текстовому документі обране поле буде в лапках, після завершення злиття замість назви поля буде вставлено його значення
Крок 5	Перегляньте, користуючись панеллю навігації, кожен документ створеної серії розсилки
Крок 6	Завершіть злиття й виберіть режим опрацювання документа розсилки. • Команда **Друкувати документи...** дає змогу роздрукувати документи для кожного зі списку джерела даних. • Якщо створювалися електронні листи, запускають команду **Надіслати повідомлення** електронної пошти. У вікні, що відкриється, є можливість цей документ використати як вкладення до листа або як основний текст листа; також можна відправити всім обраним попередньо адресатам або вказати порядковий номер адресатів у списку. • Результатом команди **Змінити окремі документи** є створення складеного документа, який містить стільки окремих документів з унікальними значеннями, скільки є записів у базі даних (якщо текст не перебільшує сторінку, а в базі даних 300 записів, то в результаті отримаєте документ із 300 сторінками: усі сторінки будуть мати загальний текст, а поля будуть заповнені унікальними значеннями з бази)

Усі команди, запропоновані майстром злиття, наявні на стрічці РОЗСИЛКИ (рис. 7). У групі Поля для записування та вставлення є список Правила (рис. 8), який містить функції керування унікальними даними основного документа.

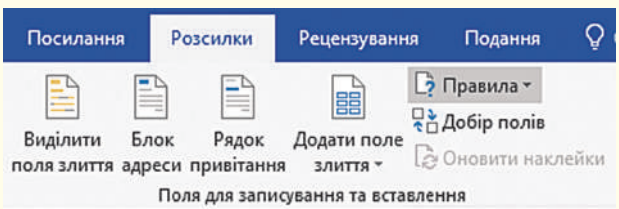


Рис. 7. Вигляд команд стрічки **РОЗСИЛКИ**

Рис. 8. Список функцій для вибору унікальних даних розсилки

У листах має бути звернення «*Шановний*» до чоловіків і «*Шановна*» до жінок, а далі ім'я. Аналізуємо: звідки взяти ім'я — з бази даних, а на основі чого додати «*ий*» або «*а*» до «*Шановн*» — на основі поля, значеннями якого є «*ж*» (жіноча) чи «*ч*» (чоловіча), букви, які визначають стать людини. Якщо «*ч*» — додається «*ий*», інакше — «*а*».

Ознайомимося з *алгоритмом оформлення звернення*.

Крок 1	В основному тексті введіть *Шановн* та розмістіть указівник після *Шановн*
Крок 2	Вставте поле з функцією: із правил викличте функцію **IF...THEN...ELSE** (якщо...то... інакше) — відкриється вікно заповнення вказаної функції (рис. 9)
Крок 3	Заповніть вікно згідно зі зразком на рисунку, підтвердьте дію кнопкою **ОК**
Крок 4	Поставте пропуск у документі. Указівник нікуди не переміщуйте. Із групи **Поля для записування та вставлення** викличте команду **Додати поле злиття**
Крок 5	У вікні, що відкриється, виберіть поле **Ім'я По батькові** (таке поле має бути в базі даних). Підтвердьте дію кнопкою **ОК**

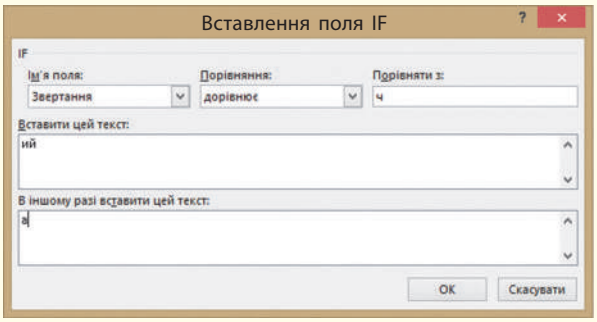


Рис. 9. Вікно заповнення поля з правилом IF...THEN...ELSE

Запитання для перевірки знань

1. Що означає поняття «злиття» для документа Word?
2. Які складові дають змогу створити розсилку? Назвіть кроки.
3. Які є режими опрацювання документа розсилки?
4. Як створити звертання до кожної особи за її іменем у розсилці?
5. Скільки сторінок може мати документ після завершення злиття?
6. Скільки файлів створюється після завершення злиття? Відповідь обґрунтуйте.

4.5. Комп'ютерні публікації. Видавничі системи. Електронні книги

Наведіть означення поняття «публікація». Наведіть приклади публікацій. Чи можна твір із літератури назвати публікацією?

Публікація — це процес видання твору.

Залежно від способу збереження розрізняють:

- друковані публікації (книжки, брошури, журнали, газети, бюлетені, буклети, листівки тощо),
- електронні публікації, їх також називають комп'ютерними (збережені на носіях, розміщені в Інтернеті, зазвичай на сайтах). У таких публікаціях (електронних книжках) завдяки налагодженим зв'язкам між частинами зручно здійснювати переходи.

Єдину за змістом і оформленням опубліковану роботу (твір, видання) теж називають **публікацією**.

Прикладами професійних програмних засобів верстки є QuarkXPress, Adobe PageMaker, Adobe InDesign тощо.
Є й інші настільні видавничі системи: Page Plus, Avery DesignPro, Paraben's Label Builder тощо.

До складу Microsoft Office входить відома вам настільна видавнича система Microsoft Publisher, яка порівнянно з професійними системами має обмежену кількість функцій. Натомість Microsoft Publisher містить інтуїтивно зрозумілий інтерфейс, значну кількість шаблонів і макетів публікацій, що значно спрощує створення.

Завдяки полям у публікації організовують автоматичне додавання: вставлення оновлення дати й часу; вставлення підписів до таблиць та малюнків; створення виносок, посилань на частини тексту, інші об'єкти з можливостями переходів до них; закладок тощо.

Пригадаємо, що видавнича система — комплекс апаратних та програмних складових для підготовки публікацій. *Апаратними* засобами є пристрої уведення та виведення, опрацювання, збереження та передавання (комп'ютер, принтер, копіювальний апарат, сканер, графічний планшет, фото- та відеотехніка). *Програмними* засобами є спеціальні програми для верстання публікації.

Процес створення публікації складається з кількох етапів. На етапі макетування та верстання визначаються параметри форматування тексту та графічних зображень, їх взаємного розміщення на сторінках для поєднання функціональних і естетичних якостей усіх складових публікації. Сучасні програмні засоби завдяки своїм можливостям автоматизують роботу.

Макет сторінки публікації — це зразок для точного розміщення текстових та ілюстративних об'єктів публікації. Макет не несе інформаційного навантаження.

Верстка — процес компонування текстового й ілюстративного матеріалу відповідно до розробленого макета сторінки (шпальти) публікації з дотриманням принципів дизайну та технічних вимог.

Пригадаємо, із яких об'єктів складається публікація. Розглянемо можливості MS Word для організації зв'язків між окремими об'єктами в публікації. Зі створенням змісту та нумерації сторінок ви вже ознайомилися.

Не обов'язково користуватися кодами полів. У програмі MS Word велика кількість полів уставляється за допомогою вбудованих функцій, які викликаються командами меню:

- у стрічковому меню ВСТАВЛЕННЯ в групі Посилання містяться команди: Закладка, Перехресне посилання;
- у групі Текст (список Експрес-блоки) містяться команди: Автотекст, Властивість документа, Поле;
- у стрічковому меню ПОСИЛАННЯ в групі Підписи містяться команди для встановлення підписів під різними об'єктами документа.

- Додаткові можливості опрацювання полів

Ми розглянули можливість уставлення номера сторінки, як верхнього або нижнього колонтитула. У верхньому колонтитулі бажано розмістити назву параграфа. Щоб це здійснити, кожен параграф має бути новим розділом, а для його назви вибрано стиль заголовка, наприклад, Заголовок 1.

Розглянемо *алгоритм вставлення у верхній колонтитул назви параграфа*.

Крок 1	Перейдіть до верхнього колонтитула на сторінці з першим параграфом
Крок 2	Відкрийте вікно **Поля** командою **ВСТАВЛЕННЯ — Експрес блоки — Поле**

Крок 3	У розділі **Категорії** виберіть **Зв'язки й посилання**
Крок 4	Зі списку полів, який відкриється, виберіть **StyleRef** (рис. 1).

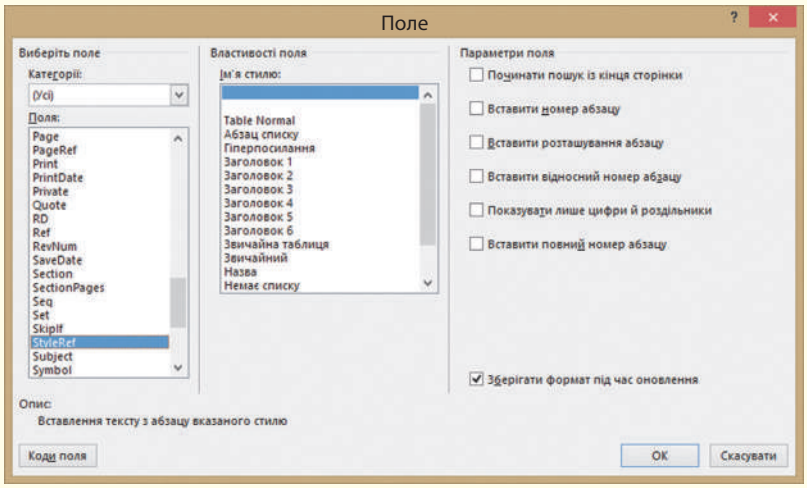


Рис. 1. Вікно вибору полів

Крок 5	В області **Властивостей поля** виберіть стиль заголовка, яким установлено назви параграфів (для нашого прикладу **Заголовок 1)**
Крок 6	Підтвердіть свій вибір кнопкою **OK**

Уставлення цього поля краще робити в шаблоні, оскільки, наприклад, у кожного автора публікації у полі StyleRef будуть відображені заголовки написаних ним параграфів.

- Підписи до об'єктів документа

Як ви помітили, у посібнику автори посилаються на рисунки або таблиці за їх номером. MS Word надає можливість автоматизувати таку роботу.

Розглянемо *алгоритм додавання назв до наявних об'єктів*.

Крок 1	Виділіть елемент, до якого слід додати назву
Крок 2	Запустіть команду **Вставити назву** зі стрічки **Посилання** з групи **Підписи** — відкриється вікно **Назва** (рис. 2). Незмінною частиною підписів є, наприклад, **Рисунок** або **Таблиця**, **Формула** — саме цю категорію й вибирають із розкривного списку підписів, а для власної назви користуються командою **Створити**

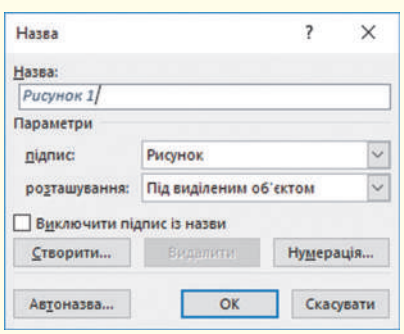


Рис. 2. Вікно вставлення підписів під рисунками

Крок 3	У списку **Розташування** зазначте, де розташовуватиметься підпис. Зазвичай рисунок підписують знизу, а таблиці — над ними
Крок 4	Для встановлення нумерації назв скористайтеся однойменою командою — відкривається вікно, у якому регулюють номери об'єктів (рис. 3)
Крок 5	Кнопка з командою **Автоназва** дозволяє задати типи об'єктів, назва до яких додається автоматично в разі додавання в документ нового об'єкта
Крок 6	Підтвердіть вибір форми підписів кнопкою **OK**

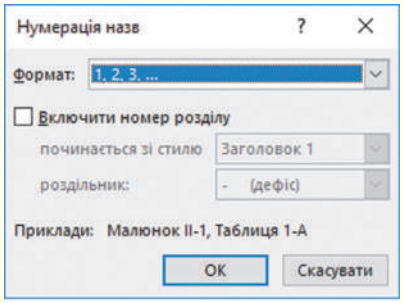


Рис. 3. Вставлення підписів під рисунками з номером розділу

Приклад.
Заголовки — стилями **Заголовок** і **номер Заголовка**, назва таблиць, малюнків — стилем **Назва** (стилі вибирають з однойменної групи стрічки **ОСНОВНЕ**).

Сполучення клавіш для вставлення виносок
- Ctrl + Alt + F — вставка звичайної виноски
- Ctrl + Alt + D — вставка кінцевої виноски

• Посилання

У звітах, доповідях часто доводиться посилатися на розділи, таблиці й ін. У групі Підписи для цього є команда Перехресне посилання. Можна зробити текстове посилання на назву теми, заголовок таблиці, підпис до рисунка; можна зробити посилання на номер сторінки, де починається тема або міститься таблиця, рисунок тощо. Зазвичай для посилання використовують текст «Див. розділ (сторінку, таблицю тощо)».

Щоб посилання працювало правильно, назви об'єктів посилання необхідно здійснити форматувати стандартними стилями (приклад).

У разі запуску команди відкривається вікно Перехресні посилання, у якому слід вибрати зі списку тип посилання і вказати, на що встановлюється посилання. Вікно містить поле для уточнення формату посилання.

Щоб уставити посилання на розділ із назвою зі стилем Заголовок 1, зі списку Тип посилання вибирають Заголовок, уточнення вставляють у полі Для якого заголовка. У нашому прикладі необхідно вказати Заголовок 1, а в полі Вставити посилання на вибрати, наприклад, текст заголовка або сторінку. Якщо посилання вставлялося на Рисунок, поле для уточнення буде називатися Для якої назви, і вписати назву.

• Виноски в документі

Іноді для пояснення деяких слів або понять використовують виноски (роз'яснення), які найчастіше розташовані або наприкінці сторінки, або наприкінці розділу або книги.

Виноски можна створити автоматично в групі Виноски стрічки ПОСИЛАННЯ:

Крок 1	Зазначте місце в документі для вставлення позначки виноски
Крок 2	Скористайтеся командою **Вставити виноску** — вказівник автоматично переміститься у нижню частину сторінки для вставлення тексту виноски
Крок 3	Форматування виносок, місце їх розташування (знизу чи згори сторінки), формат номерів виносок або встановлення символу для виноски на розсуд користувача здійснюється у вікні **Виноски**, яке викликається з однойменної групи команд стрічки **ПОСИЛАННЯ** (рис. 4).

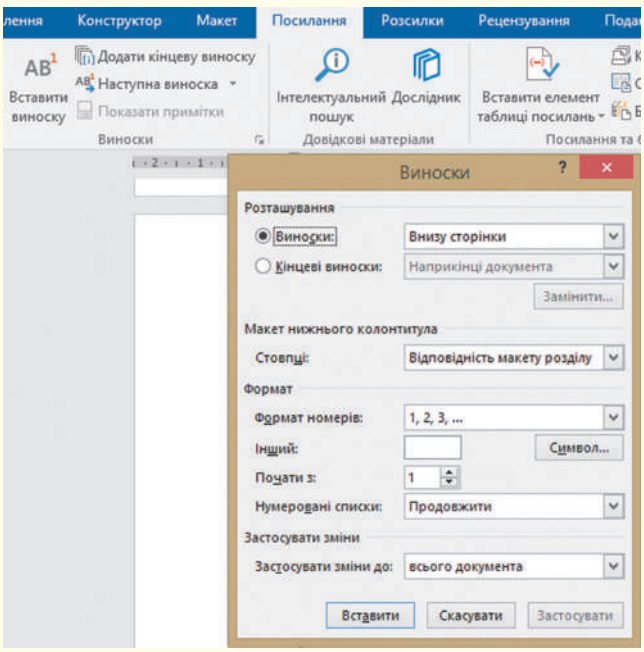


Рис. 4. Вікно форматування виносок

- Закладки

У процесі роботи з великими за обсягом документами може виникнути необхідність перейти з поточного в інше місце документа. Для зручності переходів використовують закладки. Закладки у Word є кодами. У документі їх не видно, на друк вони не виводяться.

Для вставлення закладки слід установити в потрібному місці вказівник або виділити текст і клацнути вкладку Вставити. У вікні слід ввести ім'я закладки, і клацнути кнопку Додати.

Щоб із будь-якого місця документа перейти до потрібної закладки, запускають команду стрічка ВСТАВЛЕННЯ — група Посилання — Закладка. У вікні, що відкриється, з поля закладок вибирають потрібну і користуються кнопкою Перейти.

Редагування виносок (переміщення, копіювання, вилучення) здійснюють зі знаком виноски в тексті документа (номером чи іншим символом), а не в області виноски. В останній змінюють тільки текст виноски. Якщо для виносок використовувалася нумерація й одну з них вилучили, автоматично змінюється нумерація решти виносок.

Запитання для перевірки знань

1. Що таке виноска?
2. Який алгоритм додавання заголовків параграфів як колонтитулів?
3. Назвіть особливості створення закладок.
4. Як у Word автоматизувати вставлення підписів таблиць?

Виконайте тестове завдання до розділу 4 з автоматичною перевіркою результату на сайті interactive.ranok.com.ua

Розділ 5. ГРАФІКА. МУЛЬТИМЕДІА

5.1. Комп'ютерна графіка та сучасні напрями її використання. Види комп'ютерної графіки.

Поясніть, що таке «піксель». Які види цифрових зображень ви знаєте? Наведіть приклади програмних засобів для створення та опрацювання графічних зображень.

Сучасне життя не можна уявити без застосування мультимедійних технологій — комплексу апаратних та програмних засобів, які забезпечують одночасну роботу з текстом, звуком, статичними та динамічними графічними зображеннями, відеосюжетами.

Мультимедіа важко уявити без комп'ютерної графіки. Сьогодні комп'ютерна графіка як сукупність методів і способів перетворення даних у графічне зображення і графічного зображення у дані за допомогою комп'ютера використовується в усіх сферах: у науці, освіті, медицині, на виробництві.

Комп'ютерну графіку використовують у кіноматографії, рекламі, поліграфії, з розвитком мереж — у Інтернеті, у першу чергу, службі **World Wide Web**, в електронних виданнях — веб-сторінку важко уявити без малюнків.

Комп'ютерна графіка — розділ інформатики, який вивчає методи і засоби створення та опрацювання графічних зображень із використанням комп'ютерних технологій.

Як ви знаєте, найпростішими програмними засобами, призначеними для створення нового графічного зображення та редагування вже готового, є графічні редактори. Вони перетворюють комп'ютер на віртуальну майстерню досвідченого користувача — художника-ілюстратора. Редактори містять різноманітні інструменти, та віртуальну палітру, що дає змогу отримувати фарби будь-якого типу й кольору.

Графічні редактори класифікують за принципами формування, збереження та відтворення графічного зображення на екрані монітора.

Розрізняють такі види комп'ютерної графіки: *растрова, векторна, фрактальна, тривимірна, анімаційна* (*динамічна*).

Растрові зображення являють собою прямокутну область (растр), що складається з пікселів, кожен з яких має певний колір. Піксель (від англ. PICture'S ELement) — найменший елемент зображення. Пікселі мають такий розмір, що зображення здається цілісним, хоча в разі збільшення видно зернисту структуру (рис. 1).

а

б

Рис. 1. Маніпуляції з растровим зображенням

Зміна розміру растрового зображення суттєво впливає на розмір файла, у якому воно зберігається.

Сфера застосування: розробка електронних (мультимедійних) і поліграфічних видань з ілюстраціями високої якості. Останнім часом для введення растрових зображень у комп'ютер, окрім сканера, використовують цифрові фото- та відеокамери. У мережі Інтернет нині застосовуються здебільшого растрові ілюстрації.

У векторній графіці збереження зображень засноване на математичному описі елементарних об'єктів — геометричних примітивів.

Розмір файла майже не залежить від зміни розміру зображення (рис. 2).

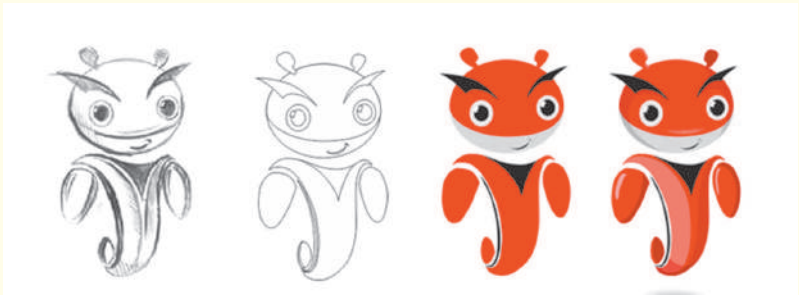


Рис. 2. Маніпуляції з векторним зображенням

Сфера застосування: виготовлення ескізів, технічних рисунків, креслень, схем, створення рекламних проектів, оформлення проектно-конструкторської документації різного рівня складності, виконання завдань з архітектурного проектування, композиції, дизайну.

Фрактальна графіка базується на автоматичній генерації зображень шляхом математичних розрахунків.

В основу методу побудови зображень покладено *принцип самоподібності* — таку назву він отримав завдяки наслідування елементами–«спадкоємцями» властивостей від так званих елементів–«батьків» (рис. 3).

Створення фрактальної композиції має основою не малювання, а формальний опис алгебраїчних або геометричних властивостей самоподібного елемента.

Рис. 3. Приклад фрактального зображення

Сфера застосування: абстрактні фрактальні композиції часто використовують у розважальних програмах, а також як тло листівок, слайдів та ін.

Найбільш поширеними растровими графічними редакторами є: **ACDSee Photo Editor, Corel PHOTO–PAINT, Microsoft Paint** (останнім часом **Paint. NET**), який входить до складу всіх версій **ОС Windows**, а також **Adobe Photoshop** і його аналог **GIMP** (GNU Image Manipulation Program).

До числа найвідоміших векторних редакторів належать **Adobe Illustrator**, **Macromedia FreeHand**, **Corel Draw**, **Inkscape**, **Corel Xara**.

Для створення фрактала необхідне середовище програмування або спеціальні програмні засоби, наприклад, **Ultra Fractal**, **Fractracer** тощо.

Створення малюнка у тривимірних графічних редакторах, наприклад, **Maya**, **SoftImage**, **3D-Studio Max**, **LightWave3D**, схоже на побудову з окремих цеглинок, як у конструкторі. Найпотужнішим редактором є **Maya**, а найпопулярнішим — **3D-Studio Max**.

Для створення анімаційних зображень існує багато різноманітних програм: **Adobe Photoshop**, **Autodesk Maya**, **GIMP**, **Blender**, **Synfig**, **Visual GIF**, **Animator** тощо.

Тривимірна графіка (3D-графіка, від англ. 3D — three dimensions — три виміри) оперує об'єктами у тривимірній системі координат, результатом є плоска картинка, проекція (рис. 4).

Рис. 4. Приклад тривимірної графіки

Початкове зображення будується як векторне, моделюються умови освітленості та встановлення точок спостереження, а вихідне зображення генерується як растрове на основі заданих об'єктів і параметрів візуалізації.

Сфера застосування: тривимірна графіка має значні можливості для здійснення технічного креслення.

Анімаційною графікою, анімацією, або мультиплікацією, називають штучне відтворення руху в кіно, на телебаченні чи в комп'ютерній графіці шляхом послідовного відображення малюнків або кадрів з частотою, що забезпечує цілісне зорове сприйняття образів.

Мультиплікація (від латин. «мульти» — багато) відповідає традиційній технології розмноження малюнка, адже для того, щоб зображення «ожило», треба повторити його рух із частотою від 10 до 30 мальованих кадрів у секунду.

Перші мультфільми з'явилися майже сторіччя тому, а сьогодні широко застосовуються нові програмні засоби, доступні для створення рухомих об'єктів не тільки фахівцям, а й учням.

Сфера застосування: напрям анімації, який дає змогу передавати природні рухи в реальному часі. Датчики прикріплюють до актора в місцях, які відповідають контрольним точкам комп'ютерної моделі (у місцях згину рук, ніг тощо). Коли актор рухається, за допомогою спеціальної програми його рухи оцифровуються — координати та орієнтація у просторі передаються на графічну станцію, і анімаційні моделі «оживають».

Запитання для перевірки знань

1. Як ви розумієте поняття «комп'ютерна графіка»?
2. Який програмний засіб називають графічним редактором?
3. Назвіть види комп'ютерної графіки.
4. Наведіть приклади використання графічних редакторів у вашому повсякденному житті.
5. Як ви гадаєте, чи є зображення сніжинки прикладом фрактала?
6. Який графічний редактор: векторний чи растровий, на вашу думку, обере користувач для створення картини, а який — для створення реклами?

5.2. Моделі відображення кольору

Поміркуйте, як формується кольорове зображення на екрані монітора. Пригадайте, що таке «піксель».

З курсу фізики вам відомо, що в призмі біле світло розкладається на кольори веселки (рис. 1). Їх сім (пригадайте приказку «Чарівниця Осінь Жар-птаха Закликає Бабин Сад Фарбувати»).

Рис. 1. Розкладання білого світла на кольори веселки

Розглянемо, як формується багатоколірне зображення на екрані монітора чи аркуші паперу. Зображення на екрані ми бачимо завдяки випромінюванню світла точками екрана, а на папері — завдяки світлу, що відбивається від поверхні аркуша. Для опису кольорів, утворюваних у різний спосіб, створено різні колірні моделі.

Колірна модель — математична модель подання кольорів у вигляді послідовності чисел: колірних компонентів або колірних координат.

Колірні моделі задають певні системи координат, які дозволяють однозначно визначити колір. Усі можливі значення кольорів, які можна описати моделлю, визначають її колірний простір. Є різні колірні моделі, основними є RGB, CMYK, Lab, HSB.

Розглянемо ці моделі та принципи кодування кольору.

- У моделі RGB кольори отримають змішуванням трьох базових кольорів, це червоний, зелений і синій (англійською мовою Red, Green, Blue).

Модель використовується для зображення, яке ми бачимо у випромінюваному світлі. Вона називається *адитивною*, оскільки колір пікселя формується поєднанням зазначених кольорів (рис. 1). Кількість кольорів, які може мати піксель, залежить від рівня яскравості того чи іншого основного кольору (приклади 1, 2).

- У моделі CMYK кольорове зображення на папері формується за іншими правилами. Точки на папері не світяться, бо папір поглинає світло, і ми бачимо кольорові зображення у відбитому світлі.

Якщо від білого кольору відокремити три кольори моделі RGB, отримаємо кольори моделі CMYK: блакитний (Cyan), пурпуровий (Magenta) і жовтий (Yellow). Модель CMYK називається *субтрактивною* (від англ. Subtract — віднімати), або *додатковою* (доповнює кольори адитивної моделі до всього спектра). Діапазон насиченості кольорів змінюється в межах від 0 до 255.

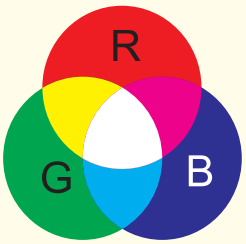


Рис. 1. Графічне зображення моделі **RGB**

Приклад 1.
Чистий зелений колір у моделі **RGB** буде описано так: 000 255 000. Тут 000 означають відсутність червоного та синього кольорів, а зелений (255) має максимальне насичення.

Приклад 2.
000 000 125 — такі значення говорять про ненасичений синій колір. Усі нулі означають відсутність будь-якого кольору, тобто чорний, а 255 255 255 навпаки — білий. У разі такого кодування можна отримати 256^3 (або 2^{24}) різних кольорів.

У моделі **CMY** (Cyan, Magenta, Yellow) всі нулі дають біле світло (усі кольорові компоненти моделі **RGB** відбились). Якщо поверхня поглинула світло повністю, то ми бачимо її чорною. Якщо на папері бачимо зображення жовтого кольору, то поглинувся синій колір моделі **RGB**, а зелений і червоний, що відбилися, у поєднанні дали жовтий. Аналогічно: бачимо на папері блакитний колір, то поглинувся червоний моделі **RGB**, а якщо пурпуровий, то зелений.

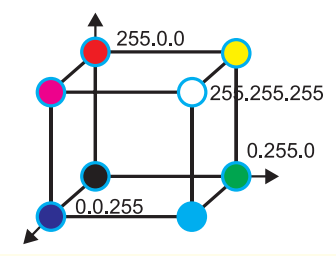


Рис. 2. Схематичне зображення кольору в моделі **RGB**

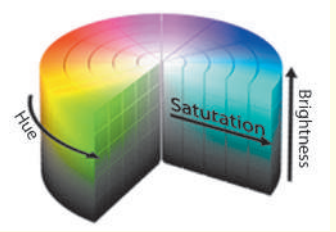


Рис. 3. Схематичне зображення кольору в моделі **HSB**

Під час друку використовують модель CMYK. Оскільки поєднання CMY-кольорів не дає глибокого чорного кольору, до основних фарб додають чорну (англ. blacK (букву «B» не використовують, щоб не плутати з Blue моделі RGB). Таким чином, у поліграфії друк здійснюється за допомогою блакитної, пурпурової, жовтої та чорної фарби, що, власне, і становить палітру CMYK.

На рис. 2 наведено модель RGB із зазначенням значень колірних складових та кольорів моделі CMYK.

- Модель Lab базується на сприйнятті кольору людиною.

На екрані монітора ми бачимо кольори в режимі випромінювання, а після друку на папері тих самих кольорів не отримуємо. Проблема розв'язується так: у процесі вибору кольору програма пропонує вибрати ще й модель, щоб приблизно дібрати кольори для зображення, призначеного на друк. При цьому найяскравіші кольори моделі RGB неможливо передати за допомогою CMYK, а для найтемніших кольорів моделі CMYK немає аналогів у RGB. Модель Lab однозначно визначає колір.

У моделі Lab використовується можливість окремо впливати на яскравість, контрастність і колір зображення. Особливо важливим це є для процесу додрукарської підготовки видання. Lab надає можливість вибіркового впливу на окремі кольори в зображенні, посилення кольорового контрасту, а також покращує якість цифрових фотографій завдяки боротьбі з шумом.

Абревіатура LAB визначає колірний простір (L*a*b) і сьогодні є міжнародним стандартом. Колір визначається трьома координатами: L — освітленістю, a — діапазоном зміни від пурпурового до зеленого та b — діапазоном зміни від синього до жовтого. Освітленість змінюється в діапазоні від 0 до 100 %. Її максимальне значення відповідає максимальній яскравості кольору. Значення діапазонів зміни кольорів задаються числами від –128 до 127.

- Модель HSB побудована на характеристиках кольору: колірному тоні, насиченості, яскравості (рис. 3).

Колірний тон (англ. Hue) — колір світла, який ми бачимо. Якщо усі кольори розмістити на колірному колі, то позиція кольору буде змінюватись від 0° до 360°, що й дасть значення параметра тону.

Насиченість (англ. Saturation) характеризує чистоту кольору певного тону, її значення коливаються від 0 до 100 % у поєднанні з сірим кольором (тон кольору задається хвилею певної довжини, решта хвиль зливаються в сірий колір). Значення 0 насиченості відповідає сірому кольору.

Яскравість (англ. Brightness) означає наскільки світлим (100 %) чи темним (0 %) подано колір.

У процесі формування кольорів на екрані кожний базовий колір RGB передається по своєму каналу, тому графічні редактори дозволяють вибрати режим кольору зображення: *кольорове*, *напівтонове* (у відтінках сірого) або *чорно-біле*.

У зв'язку з цим виникає поняття напівтонової моделі: використовується єдиний канал із 256 відтінками сірого кольору, які відповідають лише за яскравість пікселя. Зображення з використанням монохромної моделі мають два кольори: білий і чорний. Крім того, колір має таку характеристику, як глибина, що вказує на якість його передавання.

Глибина кольору — кількість бітів (обсяг пам'яті), яка використовується для зберігання і представлення кольору при кодуванні одного пікселя растрової графіки або відеозображення.

Оскільки два кольори монохромної моделі можна закодувати 1 бітом, то такі зображення займають зовсім невеликий обсяг пам'яті. Для кодування кольорів напівтонової моделі потрібно 8 бітів, або 1 байт, решта кольорів кодуються 24 бітами.

Оскільки в сучасних комп'ютерах при звертанні до відеопам'яті використовуються 32-бітна адресація пам'яті та 32-бітові шини даних, то до 24-бітного методу подання та зберігання зображення Truecolor додається 8-бітний альфа-канал, який задає прозорість зображення в деяких пікселях. Отже, колір кодується 32 бітами, а реально має 2563 кольори.

У комп'ютерах 24-бітний **Truecolor** використовує по 8 бітів для червоної, синьої та зеленої складових. Усього 16 777 216 кольорів (256 × 256 × 256). Цей тип кольору найбільш придатний для сприйняття людським оком та опрацювання зображень.

Запитання для перевірки знань

1. Що означає поняття «колірна модель»?
2. Назвіть колірні моделі для опису кольорів.
3. Поясніть принцип **RGB**-кодування. Де він використовується?
4. Що таке **CMYK**-кодування? Де воно використовується?
5. Які моделі більш точно описують колір у процесі сканування зображення та виведення цього самого зображення з файла на друк?
6. Якими кольорами необхідно позначити області поєднання основних кольорів, наведених на рис. 2?

5.3. Формати графічних файлів

Що означає поняття стиснення файлів?

Розмір файла визначається способом кодування збереженого в ньому графічного зображення, велике значення має також метод стиснення зображення.

Стиснення найчастіше застосовується до растрових зображень, оскільки вони зазвичай мають великий обсяг.

Графічний формат — це спосіб збереження графічного зображення. Розглянемо деякі формати графічних файлів.

Існує низка графічних форматів: **TIFF** (Tagged Image File Format), **JPEG** (Joint Photographic Experts Group), **GIF** (CompuServe Graphics Interchange Format), **PNG** (Portable Network Graphics), **PDF** (Portable Document Format) та ін.

Растровий **формат CALS** (англ. Computer Aided Acquisition and Logistics Support) розроблено підрозділом міністерства оборони США для стандартизації обміну графічними даними в електронному вигляді та опрацювання зображень програмами.

У **форматі GIF** використовується технологія запису пікселів зображення через рядок, тому після отримання лише частини файла складається враження, що маємо все зображення. Якість при цьому досить низька.

Формат RAW (англ. Raw — сирий) зберігає дані в нестисненому стані, що дозволяє уникнути їх втрат. В основному дані отримані з матриці цифрового фотоапарата. Зображення чорно-біле, для його перегляду необхідні спеціальні програми, оскільки інформація про колір і інші параметри зображення зберігається в окремій матриці.

Розглянемо деякі формати графічних файлів.

- У форматі TIFF не використовується стиснення, тому його називають апаратно незалежним. Це єдиний формат, який використовується в професійному дизайні для зберігання зображень високої якості. Формат TIFF є найкращим вибором при передачі растрових зображень у векторні програми і видавничі системи.
- Файли формату JPG (JPEG) мають можливість стиснення зображення в десятки разів, при цьому ступінь стиснення користувач може вибирати самостійно. У поліграфії цей формат не використовується через істотні втрати якості зображення. У форматі JPG використовується алгоритм стиснення jpeg, більш придатний для зображень, які містять плавні переходи яскравості і кольору. Найбільшого поширення JPEG набув у цифровій фотографії для передавання мережею Інтернет. Оскільки форматом JPG використовується принцип стиснення із видаленням певної частини даних, то кінцевий варіант роботи краще зберігати у JPG-форматі, щоб не втратити якість.
- Формат GIF призначено для стиснення файлів, у яких міститься багато однорідних заливок (у логотипах, написах, схемах). Файл GIF може містити не одне, а кілька растрових зображень, які завантажуються по черзі із зазначеною у файлі частотою — так створюється GIF-анімація. Основне обмеження формату GIF полягає в тому, що кольорове зображення може бути записане тільки в режимі 256 кольорів.
- Формат PNG розроблено на заміну формату GIF і використовується для розміщення зображень в Інтернеті. Порівняно з аналогічними GIF-файлами файли мають менший розмір, глибина кольору може бути 8-бітною (PNG-8) і 24-бітною (PNG-24). Файли формату PNG однаково відображають зображення незалежно від оснащення комп'ютера.
- Формат PDF розроблено для виведення електронних публікацій на друк: файли містять інформацію про векторні зображення, шрифти, розбивку на сторінки, а документ на екрані ми бачимо в такому вигляді, який він матиме після друку.
- Графічні редактори мають внутрішні формати для збереження зображення у файлах: PSD (Adobe Photoshop Document) є форматом растрового редактора Photoshop; CDR (CorelDRAW Document) — векторного редактора CorelDRAW.
- Формат XCF — убудований формат растрового редактора GIMP. Він зручний як для збереження проміжних результатів, так і для подальшого відкриття зображення й опрацювання редактором. Оскільки файли формату XCF не підтримуються більшістю програм для перегляду зображень, то після остаточного опрацювання зображення

його краще зберегти в більш поширеному форматі, наприклад, JPEG, PNG, TIFF тощо.

- Формат SVG — формат файлів, створених у векторному графічному редакторі Inkscape. У форматі збережені всі форми і ефекти створеного зображення, а файли мають розширення .svg.

Коли виникає потреба в перетворенні з одного формату в інший використовують спеціальні програмні засоби — конвертори графічних файлів.

Основна відмінність таких конвертерів від програмних засобів для перегляду зображень і графічних редакторів полягає в можливості пакетного перетворення форматів файлі. Окремі програми можуть одночасно з конвертацією виконувати й додаткові операції (наприклад, перейменування, зміна розмірів, додавання водяних знаків тощо).

Є багато програмних засобів для конвертації, але на сьогодні зручно використовувати програми-конвертори в *онлайновому режимі*.

Формат SVG (Scalable Vector Graphics standard) також призначено для забезпечення векторної графічної підтримки браузерів, для об'єднання тексту, графіки, анімації та інтерактивних компонентів.

Формат SVG є описом **XML** та **CSS** — файли цього формату можна відкривати й опрацьовувати в будь-якому текстовому редакторі.

Запитання для перевірки знань

1. Як ви розумієте поняття графічного формату?
2. Який формат мають файли з нескладними анімаційними зображеннями?
3. Який формат обрати для зображень з метою подальшого використання у веб-сторінці?
4. Що таке конвертор? Яке його призначення?
5. У якому форматі зберігаються файли растрової комп'ютерної графіки? Наведіть приклади форматів відповідно до редакторів.
6. У якому форматі зберігаються файли векторної комп'ютерної графіки? Наведіть приклади форматів відповідно до редакторів.

5.4. Створення векторних зображень в офісних програмних засобах

Пригадайте типи графічних зображень, які можна додати до документів у пакеті MS Office.

Із векторним редактором ви познайомилися у процесі створення документів у MS Word — у MS Office вбудовано векторний графічний редактор. У його середовищі можна створити як прямі лінії (відрізки), так і лінії будь-якої форми (криві).

Лінії (відрізки) малюють «протяжкою» маніпулятора миші з натиснутою лівою кнопкою миші (ЛКМ). Малювання лінії подібне до малювання олівцем.

Основою роботи у векторних редакторах є **лінія** — її ще називають **контуром**. **Замкнений контур** — такий, у якому збігаються початкова та кінцева точки. Замкненим контуром створюють фігури, які в подальшому можна залити кольором.


Пьєр Безьє (на фото зліва) — французький інженер з компанії «Рено», який у 1962 р. запропонував типи кривих для проектування кузовів автомобілів. Незалежно від Безьє в 1959 р. криві відкрив Поль де Кастельжо з компанії «Сітроен», але його дослідження вважалися виробничою таємницею до кінця 1960-х.

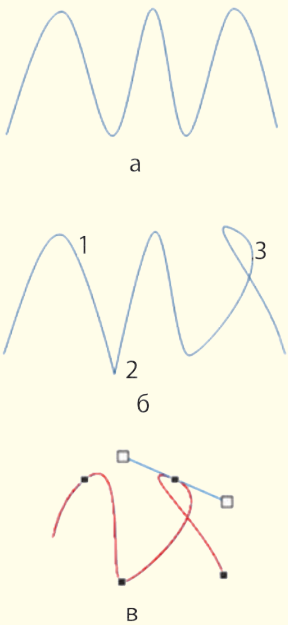


Рис. 1. Типи вузлів:
1 — гладкий; 2 — кутовий.

Криві також малюють за допомогою миші, але ЛКМ тримають натисненою тільки в точках перетину. Для завершення малювання кривої потрібно двічі клацнути ЛКМ.

Режим полілінії дозволяє створити криву й без клацання в точках перетину.

Форма кривої визначається складовими — так званими кривими Безьє. За їх допомогою можна створити зображення прямолінійного відрізка, дуги, еліпса, кола, прямокутника тощо. Якщо увімкнути режим редагування контуру, на ньому з'являються вузли, кожний з яких належить окремій складовій.

У векторній графіці розглядають різні типи вузлів: *кутові*, *прямі* та *гладкі* (плавні, згладжені). Тип вузла встановлюється напрямними.

Напрямні (або **керувальні**) **лінії** — дотичні до контуру у вузлі. Зазвичай під час редагування контуру їх показано синім кольором.

Точки, у яких з'єднуються криві Безьє, і керувальні лінії не роздруковуються. Вони використовуються для створення та редагування форми зображення.

Редагування контуру відбувається шляхом редагування наявних вузлів: додаванням, вилученням та опрацюванням (редагують напрямні). На рис. 1 зображено криву (випадок *а*) і ту саму криву зі змінами (випадок *б*). У випадку *в* наведено петлю, створену обертанням напрямної.

Зазвичай у графічному редакторі, убудованому в MS Word, створюють прості малюнки (для складних існують інші графічні редактори). Його використовують для ілюстрування тексту схемами, блок-схемами та іншими зображеннями, подібними до креслень.

На рис. 2 наведено приклад створення ілюстрації до умови або розв'язання задачі зі стереометрії. На цьому прикладі обзорно розглянено можливості MS Word для створення зображень.

Приклад. На рис. 2, а, використано фігуру Куб з групи Основні фігури, у випадках б і в — додано пунктирні лінії і вставлено написи з літерами. У процесі опрацювання фігур з'являється стрічка ФОРМАТ (Знаряддя для зображення), команди якої дозволяють змінити контур фігури, її заливку, згрупувати кілька фігур в одну, змінити положення фігури відносно решти тощо.

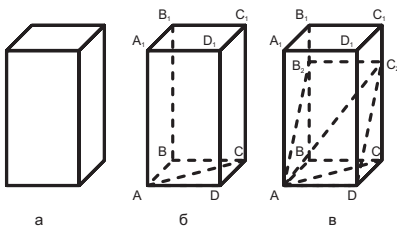


Рис. 2. Етапи створення ілюстрації до задачі зі стереометрії

Пригадаємо, як створити правильну геометричну фігуру. На стрічці ВСТАВЛЕННЯ у групі Ілюстрації списку Фігури міститься перелік фігур для створення зображення. Якщо зображення складається з кількох фігур, його зручно створювати після виклику команди ВСТАВЛЕННЯ → Фігури → Створити полотно — у документі буде створено прямокутну область для подальшого розміщення в ній фігур. Зміна розмірів зображення, його переміщення здійснюється однойменними маніпуляціями з полотном.

Якщо зображення створюється з кількох об'єктів, доцільно з утриманням клавіші Shift виділити кожний об'єкт і скористатися командою Групування. Надалі згруповане зображення сприймається Word як одне ціле. За потреби групу можна розгрупувати однойменною командою й перегрупувати.

Для створення організаційних діаграм у MS Word на стрічці ВСТАВЛЕННЯ у групі Ілюстрації є список СМАРТ АРТ, у якому подано шаблони різного виду діаграм.

Розглянемо на стрічці ПОДАННЯ вкладку Лінії сітки (рис. 3). Лінії сітки використовують в офісних програмних засобах для зручної роботи із зображеннями: відображення фігур здійснюється з прив'язкою до ліній сітки, а на друк лінії не виносяться.

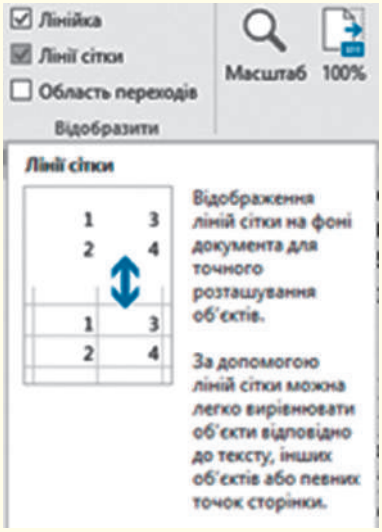


Рис. 3. Вкладка **Лінії**

Запитання для перевірки знань

1. З чого складаються контури зображень?
2. Які елементи використовують для опрацювання форми контуру?
3. Які типи вузлів є на кривих векторної графіки?
4. Для чого користуються полотном під час вставлення зображень у текстовий документ?
5. Які клавіші на клавіатурі допомагають створити правильні фігури (квадрат, коло тощо)?
6. Як вставити в документ зображення растрової графіки?

Завдання для самостійного виконання

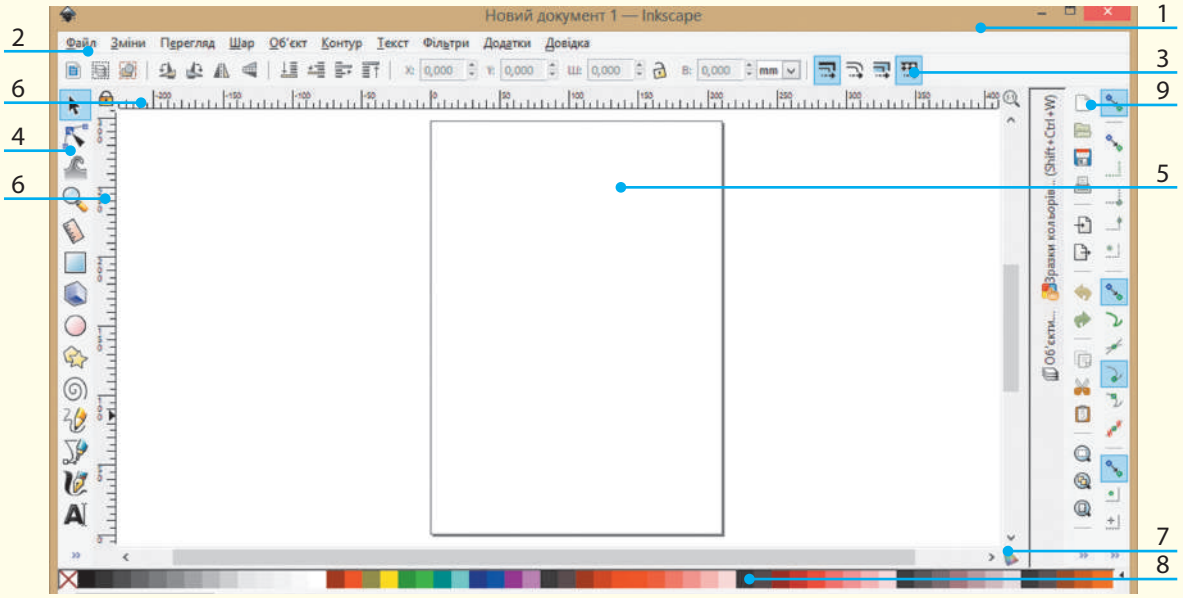
1. У відкритому документі MS Word із стрічки **ВСТАВЛЕННЯ**, групи **Ілюстрації** виберіть список **Фігури**, а в ньому інструмент **Крива**.
2. Попробуйте створити зображення першої букви вашого імені. Звичайно, одразу не вийде привабливе зображення — не змінюйте його, залиште першу спробу, на згадку.
3. Виділіть зображення й викличте контекстне меню.
4. Виберіть команду **Змінити точки** — на зображенні букви з'явились вузли саме в тих місцях, де створювали перегиб кривої.
5. Підведіть вказівник миші до будь-якого вузла і клацніть — з'являється пряма з маркерами на кінцях (див. рис. 1). Вузли показують на те, що загальна крива складається з контурів, а пряма є дотичною до контуру у точці — вузлі.
4. Якщо вказівник миші встановити на маркері дотичної, затиснути ЛКМ на ньому, то переміщенням маркера можна змінити й вигляд перетину, і, як наслідок, вигляд кривої. Так можна досягти того, щоб літера виглядала гарно.

5.5. Векторний графічний редактор Inkscape. Інтерфейс редактора

Що називають інтерфейсом? Які типи інтерфейсів ви знаєте?

Inkscape — багатофункціональний редактор векторної графіки, який має можливість зберігати створені графічні проекти в різних форматах. Програма також підтримує роботу з форматом SVG, що дає можливість використовувати градієнти, змінювати розташування шарів, створювати ілюстрації різного типу, застосовувати фільтри та ефекти.

Інтерфейс редактора має такі складові (див. рисунок).



Інтерфейс графічного редактора **Inkscape**

При наведенні вказівника миші на кнопку меню з'являється (спливає) невелике вікно підказки про її призначення.

Залежно від того, який інструмент вибраний у вікні інструментів, змінюється вигляд контекстної панелі. На ній відображаються налаштування і параметри активного інструмента.

У верхній частині вікна розташовано рядок заголовка (*1*), головне меню з низкою команд (*2*), контекстна панель параметрів активного інструмента (*3*).

Панель інструментів (*4*) утримує ряд кнопок — графічних інструментів для створення й опрацювання фігур.

Полотно (або Канва) (*5*) призначено для створення та опрацювання малюнків. Полотно не обмежує простір для малювання, засобами редактора (Параметри документа) рамку для нього можна встановити невидиму або з тінню.

У вікні за замовчуванням показано горизонтальну та вертикальні лінійки (*6*) для визначення координат у пікселях. Показ лінійок, одиниці вимірювання можна змінити в Параметрах документа (команда з меню ФАЙЛ). За трикутними маркерами на лінійках фіксуються координати вказівника миші. Ці координати (X і Y) показано в рядку стану (*7*) поруч із параметром масштабу (Z).

Над рядком стану розташована палітра кольорів Inkscape (*8*), яка дозволяє визначити колір фігур.

Вертикальна смуга (*9*) складається з двох панелей: *панелі інструментів* (дублює найчастіше використовувані команди головного меню) і *панелі прилипання* (її інструменти використовуються для правильного й точного розміщення об'єктів зображення).

Інтерфейс графічного редактора можуть доповнювати такі елементи, як напрямні або сітка. Напрямні регулюють місця розташування об'єктів малювання — об'єкти неначе притягуються до напрямних. Кількість напрямних визначає користувач. Щоб не встановлювати багато напрямних, користуються сіткою, її тип визначається у Параметрах документа меню ФАЙЛ.

Після запуску Inkscape властивості нового документа встановлюються за замовчуванням.

Опрацювання вигляду сторінки здійснюють за допомогою команди ФАЙЛ → Параметри документа. Відкриється вікно Параметри документа, на вкладці Сторінка якого можна встановити нові параметри. Наприклад, в області команди Тло встановлюється колір сторінки.

Особливістю Inkscape є інтерактивний характер застосування налаштувань користувача: будь-яка зміна параметра одразу виконується, у вікнах відсутні команди ОК і Відміна.

У **рядку стану** також показано кольори заливки й контуру, список шарів із можливістю переходів між ними, повідомлення.

Розрізняють сітку двох типів. **Прямокутна** сітка складається з вертикальних і горизонтальних ліній. **Аксонометрична** сітка містить додатково діагональні лінії, використовується в технічних або архітектурних кресленнях.

Запитання для перевірки знань

1. До якого типу графічних редакторів належить Inkscape?
2. Назвіть елементи вікна Inkscape.
3. Навіщо встановлюють показ сітки в процесі створення та опрацювання зображень?
4. Як визначити правильне розташування вказівника миші в середовищі редактора?
5. Як розташовані вісі координат у Inkscape?
6. Як ви вважаєте, якщо складові зображення розташовані за межами полотна, чи будуть вони виведені на друк?

Завдання для самостійного виконання

1. Відкрийте програму Inkscape.
2. У головному меню перейдіть до команди **Перегляд**.
3. Поставте прапорець біля команди **Напрямні**.
4. Установіть **вертикальну та горизонтальну напрямні**:
   1) розмістіть вказівник миші на горизонтальній або вертикальній лінійці;
   2) натисніть ЛКМ, перетягніть напрямну в необхідне місце полотна, відпустіть ЛКМ;
   3) так само встановіть другу напрямну.
5. Підведіть вказівник миші до напрямної — вона змінить колір; натисніть ЛКМ і перетягніть напрямну.
6. У головному меню перейдіть до команди **Перегляд**.
7. Поставте прапорець біля команди **Сітка** — у робочому полі буде показано сітку.
8. Установіть властивості сітки:
   1) перейдіть до команди головного меню **ФАЙЛ**, виберіть команду **Параметри документа** — відкриється однойменне вікно;
   2) перейдіть на вкладку **Сітка**; познайомтеся з параметрами сітки.
9. Познайомтеся з вмістом вкладок **Напрямні** та **Прилипання**.
10. Закрийте вікно Inkscape.

5.6. Інструменти векторного редактора Inkscape та їх налаштування

На основі яких математичних кривих будують контури у векторних редакторах?

Графічний примітив — мінімальний графічний об'єкт, який можна побудувати у векторному редакторі, та сукупність яких складає векторний малюнок.

Створення зображення у середовищі векторного редактора Inkscape починається зі знайомства з Панеллю інструментів. На ній містяться інструменти для побудови різноманітних кривих, основу яких складають криві Безьє, та спеціальні інструментальні засоби для створення простих об'єктів (графічних примітивів).

Щоб скористатись певним інструментом, його необхідно клацнути вказівником миші, або скористатися гарячими клавішами, — контекстна панель одразу зміниться. Малювання відбувається так само, як і в редакторі, убудованому в офісні програми. Це спрощує побудову складних об'єктів.

Якщо розмір вікна редактора не дозволяє розмістити панель інструментів повністю, то після останнього видимого інструмента з'являються стрілочки >>. Наведенням вказівника миші на стрілки відкривається список інструментів (див. таблицю), у дужках зазначено гарячі клавіші для вибору інструмента.

1 — Вибрати (Позначення та трансформація об'єктів) (F1)
2 — Редагування контурів за вузлами (F2)
3 — Коректор (Shift+F2)
4 — Масштаб перегляду (F3)
5 — Вимірювання (M)
6 — Прямокутник (Квадрат) (F4)
7 — Паралелограм (Тривимірні об'єкти) (Shift + F4)
8 — Еліпс (Коло, Дуга) (F5)
9 — Зірка, Багатокутник (*)
10 — Спіраль (F9)
11 — Олівець (Довільний контур) (F6)
12 — Перо Безьє (Малювання кривих Безьє) (Shift+F6)
13 — Каліграфічне перо, Пензель (Ctrl + F6)
14 — Текст (F8)
15 — Розпилювач (Shift + F3)
16 — Ластик (Shift + E)
17 — Заповнення (Shift + F7)
18 — Градієнт (Ctrl + F1)
19 — Сітка
20 — Піпетка
21 — Лінія з'єднання

Пригадайте, як ви створювали контури з кривих у графічному редакторі офісних програм. Для побудови контурів в редакторі Inkscape користуються інструментом Перо Безьє. Ним можна намалювати як ламану, так і криву з гладкими вузлами.

Для малювання ламаної вказівник миші потрібно перемістити в точку перегину і клацнути ЛКМ. Якщо під час створення вузла не відпускати мишу, а перетягнути її в бік наступного вузла, то створиться гладкий вузол.

Властивості вузлів можна змінити шляхом налаштування його параметрів у контекстній панелі інструментів. Пригадайте, що називають напрямною вузла та як за її допомогою змінювати вигляд контуру.

У місці розташування вузла контур можна «розірвати» відповідними інструментом контекстної панелі, а потім перетягуванням вузла в інше місце отримати два нез'єднаних контури. Можна здійснити і зворотню дію: виділити два вузли і скористатись інструментом з'єднання вузлів — два контури зіллються в один.

Пригадайте, як користуватись інструментами графічних редакторів для створення геометричних фігур. У Inkscape для цього використовують інструменти Прямокутник, Паралелограм, Еліпс, Зірка, Спіраль. Форму і розмір об'єктів можна змінити переміщенням маркерів.

На рис. 1, *а* відповідно зверху вниз наведено об'єкти, побудовані інструментами, розташованими на контекстній панелі:

- Прямокутник;
- Еліпс (вибрали форму цілий еліпс);
- Зірка, Багатокутник (вибрали багатокутник із кількістю кутів — 5);
- Зірка (Багатокутник) (вибрали зірку з тією самою кількістю кутів — 5);
- Спіраль (вибрали кількість витків — 3).

У процесі побудови на таких об'єктах з'являються круглі та квадратні маркери, які мають різне призначення. Круглі маркери змінюють форму об'єкта (у прямокутника змінюють вигляд кутів (отримують прямі чи скруглені кути), з еліпса створюють сектор чи дугу. *Квадратні маркери* змінюють розмір об'єкта. Перетягування квадратних маркерів зірки змінює її форму; у маркерів багатокутника і спіралі інше призначення. Багатокутник має тільки квадратний маркер для зміни розміру та обертання; маркером спіралі регулюється кількість її витків.

На рис. 1, *б*, показано результати маніпулювання маркерами: круглими (б) та квадратними (*в*).

Розглянемо особливості роботи з інструментом Вибрати (будемо вважати його активним).

Об'єкт виділяють клацанням. Якщо ЛКМ натиснути у лівому верхньому куті довільної прямокутної ділянки, а відпустити у правому нижньому, буде виділено всі об'єкти, що до неї потрапили.

Якщо виділений об'єкт клацнути ще раз, то зміниться вигляд маркерів прямокутної області виділення. Слід пам'ятати, що кутові маркери призначено для обертання об'єкта, а горизонтальні або вертикальні — для зсуву по відповідних сторонах (маркерами зсуву прямокутник перетворюють на паралелограм).

Важливим об'єктом векторного зображення є **сплайн** — крива, за допомогою якої описується та чи інша геометрична фігура. В Inkscape вибір режиму сплайна здійснюється кнопкою контекстної панелі інструмента **Перо Безьє**. На сплайнах побудовано сучасні шрифти **TrueType** і **PostScript**.

За допомогою маркерів змінюють розмір і вигляд об'єктів, але не форму контуру.

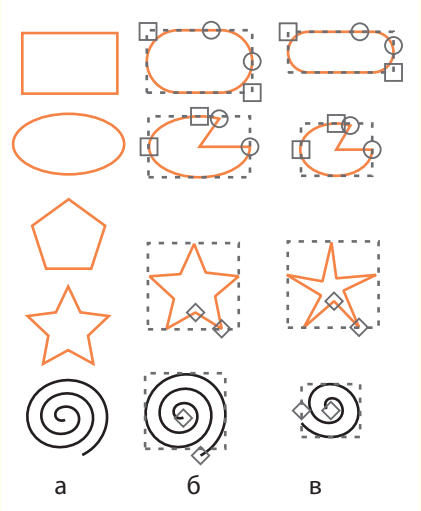


Рис. 1. Опрацювання основних графічних примітивів.

Запитання для перевірки знань

1. Що можуть означати стрілки в кінці панелі інструментів Inkscape?
2. Якими інструментами малюють контури?
3. Які режими використання інструмента **Перо** є в Inkscape?
4. Які інструменти призначені для створення фігур?
5. Як можна змінити розмір і форму об'єкта: назвіть кілька варіантів?
6. Які команди контекстного меню інструмента **Зірка** так змінюють початкову форму, наприклад, зірочки ✸✸❁✸?

Завдання для самостійного виконання

Ознайомитися з можливостями Inkscape щодо створення контурів.

1. Для зображення ламаної виберіть інструмент **Перо Безьє** (малювання кривих Безьє чи прямих ліній).
2. На контекстному меню по черзі вибирайте різні параметри малювання лінії, спробуйте «написати» малу прописну літеру г (від слова «графіка») — їх має бути 4 або 5, якщо скористатися параметрами малювання лінії з прямими кутами.

   Щоб намалювати ламану, виконайте такі дії:

   1) розмістіть вказівник миші в довільне місце полотна;
   2) клацніть ЛКМ для вибору початкової точки малювання;
   3) перемістіть вказівник у місце, де має бути перетин лінії; клацніть ЛКМ — для ламаної;
   4) продовжуйте, поки отримаєте щось подібне на літеру;
   5) у кінцевій точці двічі клацніть ЛКМ.
3. Розмістіть вказівник миші на зображенні літери, намальованої як ламана, та клацніть — об'єкт буде виділено пунктирною прямокутною рамкою з маркерами по її контуру, як на рис. 2.

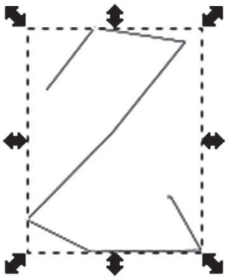


Рис. 2. Зразок «написання» літери г

4. Зробіть активним інструмент **Вибрати** (Позначення та трансформація об'єктів). Зверніть увагу на контекстну панель (більшість її команд вам вже знайомі з попередніх курсів).
5. Скористайтеся кутовими маркерами, щоб за допомогою миші змінити розміри об'єкта.
6. Змініть вигляд ламаної. Для цього:
   1) активізуйте інструмент редагування контурів — контекстне меню містить команди редагування вузлів; на ламаній з'являться позначки вузлів;
   2) за допомогою миші виберіть будь-який вузол (наприклад, другий) — вибраний вузол матиме інший колір;
   3) на контекстній панелі виберіть інструмент **Зробити позначені вузли гладкими** (користуйтесь спливаючими підказками) — форма лінії у зазначеному вузлі зміниться.
7. Із натиснутою клавішею Shift виберіть решту вузлів, вигляд яких необхідно змінити.
8. Якщо є потреба, виберіть вузол і відкоригуйте форму кривої за допомогою напрямних.
9. Виберіть інструмент **Олівець** і знову «напишіть» літеру «г». Зверніть увагу, що контекстна панель містить такі самі параметри, як і для інструмента **Перо Безьє**.
10. Виберіть інструмент **Пензель** і ще раз «напишіть» літеру «г» — робота з інструментом нагадує написання каліграфічним пером.

5.7. Колір в Inkscape. Фарбування градієнтом

Пригадайте команди для заливки фігур та малювання їх контурів в офісних редакторах. Чи можна замінити кольори вже побудованих фігур? Які кольорові моделі ви знаєте?

Ми ознайомилися з основними прийомами малювання геометричних фігур, але не звертали уваги на колір цих фігур. Розглянемо рис. 1. У верхньому рядку подано квадратні об'єкти різного кольору, а в нижньому — такі самі об'єкти, з контурами різних кольорів, але без зафарбовування внутрішньої області.

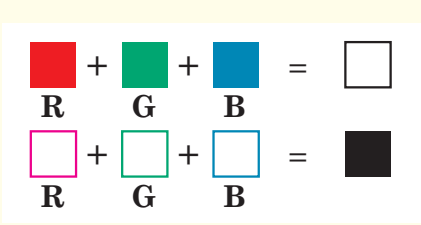


Рис. 1. Графічне зображення, подане в колірній моделі **RGB**

В Inkscape зафарбовування внутрішньої області об'єкта називають заповненням, а фарбування контура — штрихом.

Щоб вибрати колір та тип зафарбовування, потрібно скористатися вікном Заповнення та штрих (рис. 2), яке відкривається командою Заповнення та штрих у меню ОБ'ЄКТ.

Як бачимо, у ньому є три вкладки: Заповнення, Колір, штриха, Стиль штриха.

У вкладці Заповнення можна вибрати режим зафарбовування внутрішньої області вибраного об'єкта: *суцільне заповнення*, *лінійний* або *радіальний градієнт*.

Якщо фарбування відсутнє, інструмент матиме вигляд хрестика.

Під інструментами з режимом заповнення міститься область вибору кольору з кнопками RGB, CMYK, HSL, «колесо». «Колесом» зручно користуватися: на колі вибирається тон кольору, а в трикутнику встановлюється його насиченість та яскравість. Рядок альфа-канал дозволяє встановити прозорість кольору.

Якщо на вкладці Заповнення вибрати режим Зразок, то виділений об'єкт зафарбується кольором, вибраним на панелі кольорів. Щоразу, створюючи об'єкт, потрібно використовувати попередньо вибраний колір заповнення та штриха.

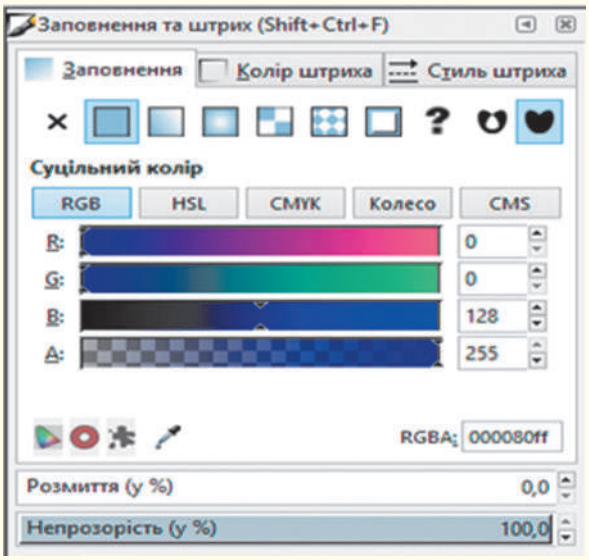


Рис. 2. Вікно **Заповнення та штрих**

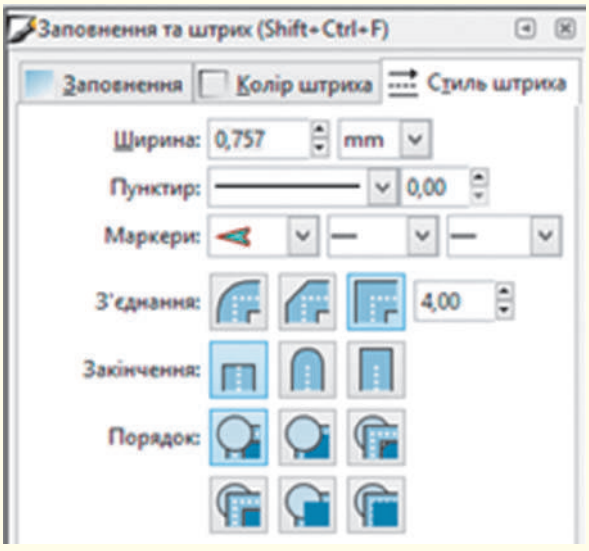
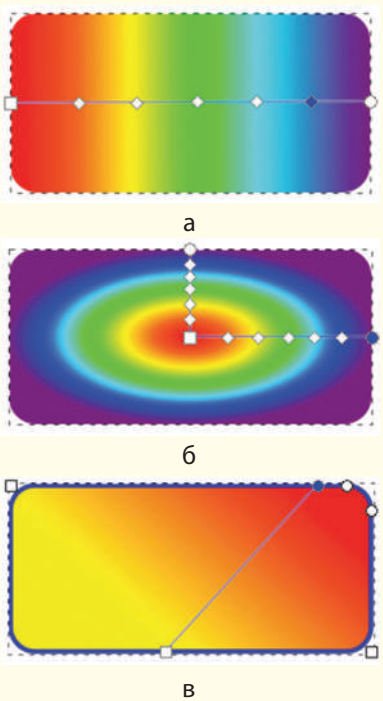


Рис. 3. Вікно вибору стилю штриха

Кожний об'єкт обов'язково має контур і заповнення. Якщо вибрано команду без зафарбовування, об'єкт все одно має зазначені елементи. Команди в Inkscape розподіляються на команди опрацювання об'єкта або його контуру.

Для фарбування також можна скористатися інструментом **Заповнення**, а колір вибрати на колірній панелі (навести вказівник і клацнути квадратик із кольором) або за допомогою інструмента **Піпетка**.



а

б

в

Рис. 4. Переміщення важелів градієнта

Вкладка Колір штриха використовується для встановлення кольору та прозорості, штриха та їх вилучення.

На вкладці Стиль штриха (рис. 3) містяться параметри штриха, позначки для закінчення несуцільного штриха та порядоку розташування певного штриха серед решти.

Розглянемо *алгоритм заповнення внутрішньої області поточного об'єкта.*

Крок 1	Виберіть команду ОБ'ЄКТ → **Заповнення та штрих**
Крок 2	Перейдіть до вкладки **Заповнення**
Крок 3	Налаштуйте режим заповнення та колір (для заповнення непрозорість не дорівнює 0)
Крок 4	Розмістіть вказівник миші всередині області фарбування
Крок 5	Підтвердьте клацанням ЛКМ

Вибраний колір буде поточним, поки його не змінити. Щоб змінити колір об'єкта, його слід виділити і вибрати колір на колірній панелі (клацнути ЛКМ квадратик з кольором).

Зафарбувати внутрішню область об'єкта можна як суцільним кольором, так і з використанням градієнта.

Градієнтом називають спосіб фарбування, у разі застосування якого відбувається плавний перехід від одного кольору до іншого.

Градієнт використовують як для заповнення внутрішньої області об'єкта, так і для фарбування штриха.

Під час вибору параметра градієнт використовується поточний колір — він змінюється від повної насиченості до прозорості. На об'єкті з'являється зображення так званого *важеля градієнта*.

Його опорні точки призначено для встановлення межі градієнтного заповнення, її колірної насиченості, зміни спрямованості та кольорів градієнта.

Керувати параметрами градієнта можна за допомогою команд контекстного меню інструмента Градієнт.

Градієнти розподіляються на два типи: *лінійний* і *радіальний*. На рис. 4 наведено прямокутники, зафарбовані різними типами градієнтного заповнення з кольорами веселки. Як бачимо, радіальний градієнт має два важеля: вертикальний і горизонтальний (*а*, *б*). Важелі можна переміщувати й таким чином створювати напрямок переходу кольорів (*в*).

Запитання для перевірки знань

1. Назвіть команди заповнення внутрішньої області та штриха об'єкта.
2. Як відкрити вікно налаштування кольорів об'єкта?
3. Які вкладки містить вікно **Заповнення та штрих** та яке їх призначення?
4. Які типи заповнення є в Inkscape?
5. Наведіть алгоритм заповнення внутрішньої області об'єкта.
6. Поясніть механізм керування параметрами градієнта.

Завдання для самостійного виконання

1. Намалюйте коло, виберіть його за допомогою інструмента **Вибір**.
2. У вікні **Заповнення та штрих** виберіть режим **Суцільний колір** та установіть колір фарбування, наприклад, червоний.
3. Виберіть тип заповнення **лінійний градієнт** — на зображенні показано важіль градієнта.
4. Поекспериментуйте з переміщенням опорних точок. Для цього:
   1) виберіть круглу опорну точку на важелі градієнта — точка підсвітиться кольором;
   2) виберіть у палітрі кольорів колір, наприклад, синій, — отримаємо перехід кольору від червоного до синього;
   3) аналогічно змініть перехід кольору від синього до зеленого.
5. Двічі клацніть на важелі градієнта — з'явиться нова опорна точка. Після цього:
   1) виберіть для цієї точки колір, наприклад, жовтий — отримаємо синьо-жовто-зелений градієнт;
   2) додайте ще одну точку й виберіть колір на власний розсуд.
6. Переміщуйте одну з опорних точок — з'ясуйте, як змінюється напрямок градієнтного заповнення.

5.8. Складені векторні зображення в Inkscape

Згадайте можливості офісного графічного редактора. Як ви опрацьовували зображення, що складаються з багатьох об'єктів?

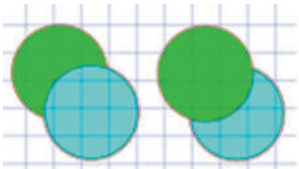
Зображення, для побудови яких використовували кілька примітивів, називають складеними.

У таких зображеннях об'єкти можуть перекривати один одного (приклад 1). Взаємне розташування об'єктів регулюється командами контекстного меню інструмента Вибрати: Опустити на задній план або Опустити на рівень, Підняти на передній план або Підняти на рівень.

Для роботи з кількома об'єктами як одним цілим у Inkscape передбачено групи команд для групування об'єктів, об'єднання об'єктів, створення складних контурів. Команди опрацювання виділених об'єктів містяться в меню ОБ'ЄКТ.

Операція групування полягає в об'єднанні двох або більше об'єктів. Далі отриманий об'єкт опрацьовується як графічний примітив. Його можна переміщувати, обертати, змінювати

Приклад 1.
На рис. 1 зображено еліпси із заповненням. Існує два варіанти їх розташування: а — верхній еліпс міститься на рівень нижче від іншого; б — міститься на рівень вище.



а б

Рис. 1. Перекриття об'єктів

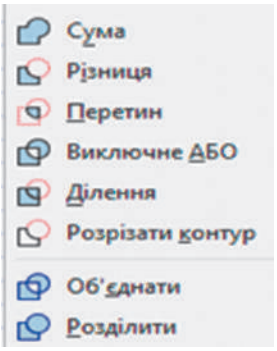


Рис. 2. Команди меню **КОНТУР**

розміри без спотворення взаємного розташування і пропорцій його складових.

Операція об'єднання призначена для опрацювання контурів виділених об'єктів (команди Об'єднати і Розділити в меню КОНТУР). У результаті об'єднання утворюється новий об'єкт, який має властивості об'єкта, який розміщується на верхньому порівняно з іншими об'єктами рівні.

На рис. 2 наведено команди меню КОНТУР. Меню КОНТУР містить також команди для створення складених контурів на основі логічних операцій (Та, Ні, Або, Виключне Або тощо).

Розглянемо ці команди та нові об'єкти, створені на основі двох еліпсів.

Приклад 2.

На рис. 3 наведено приклади застосування команд у тому самому порядку, як вони подані в меню **КОНТУР** (перші малюнки в кожному рядку — зразки, до яких застосовували команди).

У випадку *а* опрацьовували контури еліпсів без заповнення (перший малюнок в ряді) за допомогою команди **Сума**, **Різниця**, **Перетин**.

Для наочності наступні команди застосовували до зафарбованих еліпсів (випадок *б*): перший малюнок — зразок, надалі — результат дії команд **Виключне АБО**, **Ділення**, **Розрізати контур.** Наступна група команд об'єднання: **Об'єднати** та до утвореного об'єкта застосували команду **Розділити** (зверніть увагу на колір).

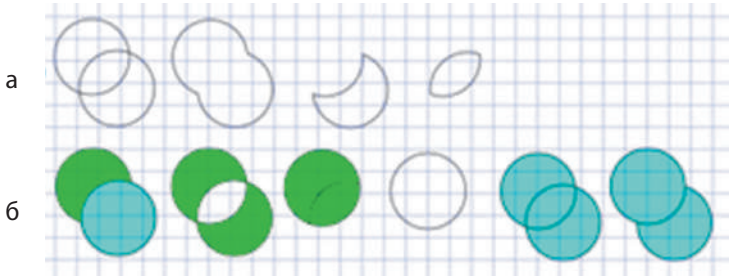


Рис. 3. Приклади використання команд меню **КОНТУР**

У меню **ЗМІНИ** в разі виклику команди **Клонувати** розкривається список команд, серед яких є команда створення одиничного клона та команда створення мозаїки клонів.

Якщо зображення складається з об'єктів однакової форми, їх можна не малювати окремо, а застосувати команди меню ЗМІНИ: копіювати, вставити, дублювати, клонувати.

У разі виконання операції копіювання або дублювання нові об'єкти створюються повністю самостійними і незалежними від оригінала. Зазначені операції зручно виконувати за допомогою миші. Вказівник слід навести на об'єкт і почати його переміщувати — у місці, де має бути копія, натиснути клавішу Пропуск.

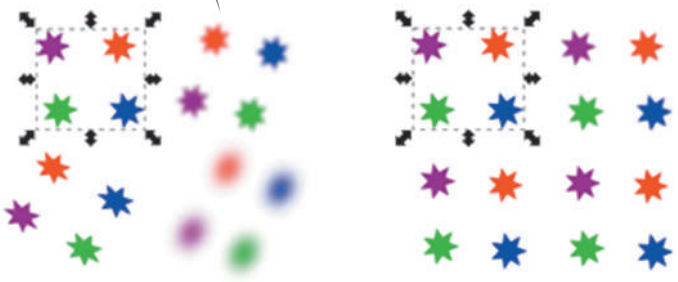
Клон об'єкта має тільки ті властивості, які є в оригіналі, і змінити їх у клоні не дозволено. Контекстне меню клону навіть не містить команд змінення заповнення або кольору контуру.

Властивості клону змінюються синхронно із змінами властивостей оригіналу.

Окремо для клону можна змінювати його просторове розміщення. Та якщо виконати команду ЗМІНИ — Клонувати —

Приклад 3.

Для зображення феєрверка достатньо створити кілька різнокольорових зірочок, а далі кожну клонувати в мозаїку і у вікні, що розкриється, задати кількісні параметри мозаїки (рис. 4).



а б

Рис. 4. Створення мозаїки клонів об'єкта (4 зірочки) з різними режимами клонування

— Від'єднати клон, клонований об'єкт стає самостійним із доступними для зміни властивостями.

Будь-які об'єкти можуть бути перетворені на контур. При цьому візуально не відбувається жодних змін, але змінюються властивості об'єктів. До контурів можна застосувати ефекти, які суттєво змінюють вигляд об'єкта.

У об'єкта прямокутник можна змінити заокругленість кутів, але після його перетворення в контур ця операція стає недоступною. Проте з'являється можливість редагувати його вузли.

Запитання для перевірки знань

1. Які зображення називаються складеними?
2. Як редагують рівні розміщення кількох об'єктів?
3. Як швидко створити кілька об'єктів на основі існуючого?
4. Які режими клонування є в Inkscape?
5. Навіщо об'єкт перетворюють на контур?
6. Які команди наявні у вікні налаштування мозаїки клонів та яке їх призначення?

Завдання для самостійного виконання

1. У вікні Inkscape відкрийте файл із заготовкою орнаменту до практичної роботи №30.
2. Зробіть активним інструмент **Вибрати**.
3. Опрацюйте з першою п'ятіркою кругів — фрагментів орнаменту. Для цього:
   1) виділіть один із бокових кругів і центральний, утримуючи клавішу shift;
   2) до виділених об'єктів застосуйте команду **КОНТУР — Перетин**: зверніть увагу на колір отриманого об'єкта.
4. Повторіть попередню дію до другої п'ятірки кругів — фрагментів орнаменту. Який колір має створений фрагмент?
5. Створіть орнамент за зразком (рис. 5). Для цього:
   1) створіть клони для кожного з отриманих об'єктів: ЗМІНИ — Клонувати — Мозаїка клонів — відкриється вікно Створити мозаїку клонів.
   2) у вікні, що відкриється, встановіть кількість рядків та стовпців — по 2;
   3) застосуйте команди дзеркального відображення деяких клонованих об'єктів, додайте нові та перемістіть їх, як показано на рис. 5;
   4) за бажанням зафарбуйте внутрішню область об'єктів.
   5) збережіть роботу у файлі з іменем **Орнамент2**.

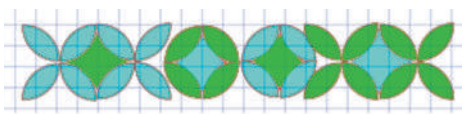


Рис. 5. Зразок орнаменту

6. Переміщуйте одну з опорних точок, з'ясуйте, як змінюється напрямок градієнтного заповнення?

5.9. Опрацювання тексту в Inkscape

Що таке напис у текстовому документі? Як форматують текст у написах?

Для створення текстових об'єктів у Inkscape використовують інструмент Створювати і Правити текстові об'єкти. В Inkscape розрізняють такі типи тексту: простий і фігурний (художній).

Тип тексту

Простий текст	Фігурний (художній) текст
Використовують для великих за обсягом текстових фрагментів; має такі самі структурні елементи, як у текстових процесорах: абзаци, колонки тощо. Опрацювання і форматування здійснюється, як у написах текстових документів (рис. 2, а)	Використовують для невеликих за обсягом текстових фрагментів, наприклад, заголовків. Вибравши інструмент, вказівник потрібно розмістити в місце введення тексту і набрати текст з клавіатури (рис. 2, б)

Рис. 1. Класифікація типів тексту

а

б

Рис. 2. Маніпуляції з текстом:
а — простим; б — фігурним

Створення текстових написів здійснюється інструментом Текст. Для створення простого тексту потрібно: розмістити вказівник у місці створення рамки, намалювати рамку і ввести текст. У рамку можна вставити вже скопійований текст. Простий текст можна розмістити у фігурний контейнер або вздовж контуру: виділити текст і фігуру, наприклад, еліпс, і запустити команду ТЕКСТ — Розмістити по контуру.

Щоб визначити тип тексту, його необхідно виділити інструментом **Текст** — простий текст містить ромбоподібний маркер у правому нижньому куті контейнера.

Опрацьовують виділений текст за допомогою панелі контекстного меню. Як і інші інструменти, Текст може вибирати об'єкти свого типу — текстові об'єкти. Отже, клацанням миші вибирають текстовий об'єкт і, у разі потреби, починають змінювати текст.

Який шрифт вибрати для тексту, що вводиться, залежить від наявності шрифту в системі, де буде відкрито SVG-файл. У меню ТЕКСТ містяться команди для перетворення типів текстів.

Пригадайте, як регулюється міжсимвольний інтервал у текстовому процесорі. Найпростішими маніпуляціями з виглядом тексту є регулювання *міжсимвольного* та *міжрядкового* інтервалів: гарячі клавіші Alt та клавіші переміщення вказівника змінюють існуючий інтервал на відстань один піксель.

Якщо текстовий об'єкт перетворити на контур, з'являється можливість переміщення літер як звичайних контурів. Для підготовки зображення з текстом до друку, літери тексту теж перетворюють на криві — і зображення, яке було текстом, можна «прочитати» навіть за відсутності у друкарні шрифтів.

Клавіші Alt+[або], застосовані до виділеної літери, здійснюють її обертання. Зміною інтервалів та обертанням літер створюють красиві текстові зображення.

Запитання для перевірки знань

1. Які типи тексту є в Inkscape?
2. Як уводять простий текст у робоче поле?
3. Чим фігурний текст відрізняється від простого?
4. Як візуально визначити тип тексту в Inkscape?
5. Які маніпуляції можна здійснити з літерами тексту?
6. Навіщо текст перетворюють на контур на етапі підготовки зображення до друку у друкарні?

5.10. Художні ефекти в Inkscape

Під час виконання практичної роботи, ви познайомилися з деякими ефектами, які застосовували до тексту. Операції з контуром також можна вважати ефектами роботи з контуром. Деякі з них вам вже відомі, тож розглянемо ще кілька.

Цікавим є результат взаємодії тексту та контуру. Якщо в робочому полі виділити спочатку текст, потім контур і виконати команду ТЕКСТ — Розмістити по контуру, то отримаємо текст, який буде розміщено вздовж контуру. В отриманому зображенні можна вилучити контур — залишиться тільки текст (рис. 1).

Ефекти опрацювання контуру можна застосувати не тільки до тексту, а й до об'єктів (рис. 2).

Меню КОНТУР містить і інші команди встановлення ефектів до контурів. Команда Спростити найчастіше застосовується для контурів, створених інструментом Олівець. Оскільки такі контури мають багато вузлів, то, використовуючи команду, можна згладити форму кривої шляхом їх вилучення.

Розглянемо *алгоритм вибору ефектів контуру.*

Крок 1	У меню **КОНТУР** викличте команду **Ефекти контурів** — відкривається вікно ефектів
Крок 2	Виберіть ефект — вигляд вікна зміниться
Крок 3	Виберіть необхідні параметри ефекту, клацніть кнопку **ОК**

Якщо побудувати зображення контуру гілки, а в ефектах контурів вибрати *фон Кох*, то отримаємо верхнє зображення на рис. 5: намальована еталонна гілочка та дві знизу як результат ефекту для кількості рівнів 1.

У разі збільшення кількості рівнів кожна гілочка наступного рівня породжує дві зменшені. Напрямні з маркерами з'являються після роботи з вузлами еталонного або створеного зображення

Приклад 1.
На рис. 1 наведено приклад побудови тексту, який розміщено вздовж контуру: введення тексту (А); малювання контуру (Б); результат виконання команди **Розмістити по контуру** (В); результат після вилучення контуру (Г).

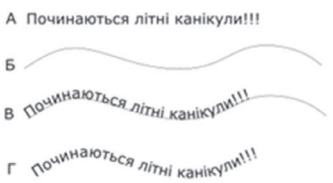


Рис. 1. Етапи побудови тексту, розташованого вздовж контуру

Приклад 2.
Уздовж контура можна розмістити об'єкт **зірочка** (рис. 2)

Рис. 2. Результат ефекту розміщення зірочки вздовж контуру

Приклад 3.

Як вам відомо, одним із видів графіки є фрактальна графіка. В Inkscape фракталізацію об'єкта дозволяє здійснити ефект *фон Кон*. На рис. 3 подано фрактальне зображення гілочки, яку створено без додавання ефектів.

На рис. 4 також маємо фрактальне зображення гілочки, тільки в кольорі. Уважно придивіться до зображення — на ньому повторюється зменшене зображення гілки з чотирма листками. Кількість повторень — 4.

На рис. 5 видно пошкоджені контури. Початковий вигляд гілки не мав пошкоджень, але під час фракталізації відбуваються математичні розрахунки вигляду контурів і можуть з'являтися дефекти. Якщо виділити складову початкового зображення, то виділяється область, яка охоплює такі самі складові всіх рівнів.

Не можна просто так перемістити зображення рівнів, адже вони неначе прив'язані до еталона. Але завдяки команді вузли з рядку Створення контуру з'являються напрямні з маркерами, за допомогою яких переміщують зображення рівня.

Рис. 3. Зображення фрактальної гілочки

Рис. 4. Результат опрацювання вже створеної фрактальної гілочки

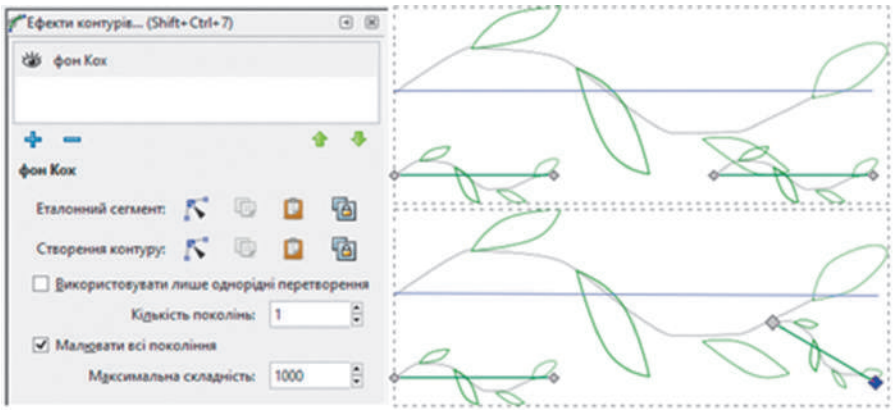


Рис. 5. Вигляд результату застосування ефекта фон Коха

На рис. 6 показано результати застосування ефектів Розсіювання та Інтерполяція для двох контурів еліпса і чотирикутника. Спочатку створили об'єкти, потім перетворили їх

на контури, виділили обидва та застосували ефект. Кількість розсіяних чотирикутників по еліпсу встановлюється відповідним параметром.

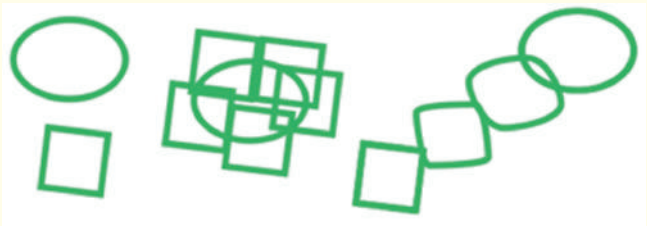


Рис. 6. Результати застосування ефектів **Розсіювання** та **Інтерполяції** до двох контурів

У меню ДОДАТКИ командою Втягування/Розтягування ореолу... створюють цікаві маніпуляції з контуром.

Ефекти застосовуються до окремих об'єктів (із замкненим і незамкненим контуром) і до кількох, але незгрупованих командою Групувати об'єкти. Налаштування ефекту здійснюється через меню ДОДАТКИ.

Інтерполяція створює ефект плавного переходу контуру одного об'єкта в контур іншого за встановленою кількістю проміжних складових.

Ореол являє собою набір контурів, які називають еквівалентними та за формою подібні до основного контуру об'єкта. Ореоли є збільшеними та зменшеними контурами зразкового і поширюються від нього ззовні або всередині нього.

Запитання для перевірки знань

1. Які ефекти можна застосувати до тексту в Inkscape?
2. Яке призначення команди **Спростити** в меню **КОНТУРИ**?
3. Поясніть на прикладі ефект фракталізації.
4. Скільки об'єктів має бути для ефектів **Розсіювання**, **Інтерполяція?**
5. Як запустити та налагодити параметри ефектів з контурами?
6. Чи можливо застосувати ефект **Інтерполяція** для перетікання кольорів?

Завдання для самостійного виконання

1. Створіть зображення зірочки та контур.
2. Інструментом **Вибрати** спочатку виділіть зірочку, потім контур.
3. Запустіть команду **ДОДАТКИ — Використання контуру — Візерунок уздовж контуру**. Відкриється вікно (рис. 7), у якому налагоджені параметри для зображення.
4. Уведіть параметри як на рис. 7 і проставте прапорець біля **Перегляд у дії**, яке ви отримали зображення.
5. Проставте прапорець біля команди **Візерунок** є вертикальним. Як змінилося зображення?
6. Зніміть попередньо встановлені прапорці та запустіть команду **Застосувати**.

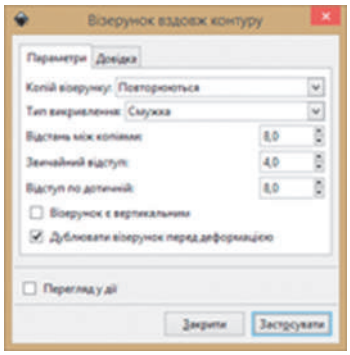


Рис. 7. Вікно налагодження вигляду візерунка вздовж контуру

5.11. Растровий редактор GIMP

Яке зображення називають растровим? Що означають поняття «піксель», «растр», «кольорова модель», «глибина кольору»? Що означає «інструмент графічного редактора»?

Растрове комп'ютерне зображення складається з пікселів, розташованих по рядках і стовпцях. Для опрацювання таких зображень використовують растровий графічний редактор GIMP (GNU Image Manipulation Program).

Програмний засіб GIMP багатофункціональний: він призначений не лише для створення растрових зображень, а також служить додатком для ретуші фотографій, мережевих систем пакетної обробки зображень, програмою для відтворення зображень, конвертором форматів зображень тощо.

До можливостей і основних функцій редактора належать такі: наявність повного набору інструментів для створення та опрацювання зображень; раціональне використання пам'яті; підтримка альфа-каналу для корекції прозорості; можливість роботи з шарами й каналами та багаторазової відміни й повтору операцій.

Розглянемо інтерфейс графічного редактора GIMP. Інтерфейс має два режими: *багатовіконний* і *одновіконний*. Під час відкриття вперше спрацьовує багатовіконний режим.

Інтерфейс редактора має такі складові (рис. 1).

Перша версія GIMP написана Пітером Маттісом і Спенсером Кімболом. Чимало програмістів внесло свій вклад у розробку програми, допомогло з підтримкою і тестуванням. GIMP випускається командою розробників під керівництвом Свена Нойманн і Мітча Наттерера.

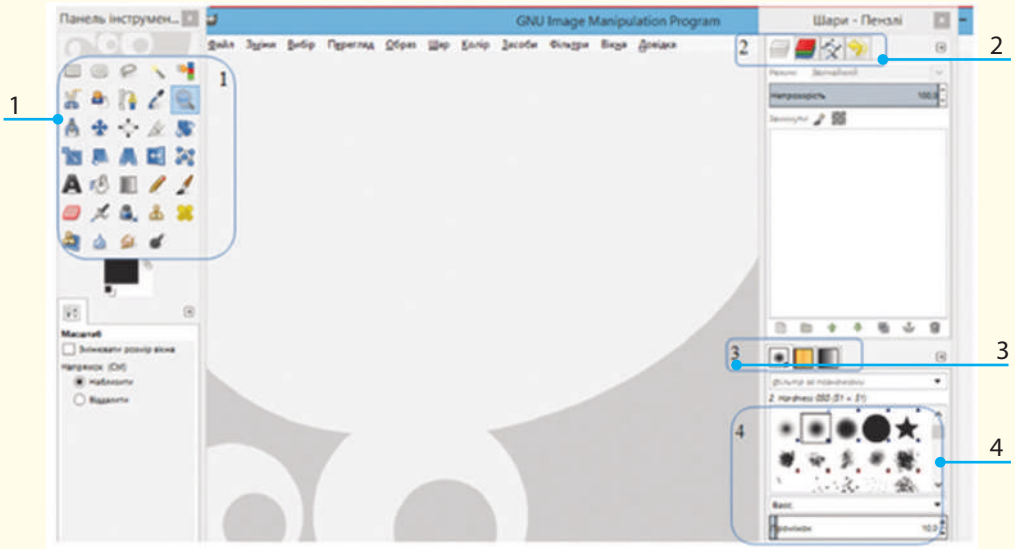


Рис. 1. Інтерфейс графічного редактора GIMP

1 — панель інструментів, під нею — кнопки налаштування кольорів і вікно налаштування параметрів обраного інструмента;

2 — область діалогу: шари, канали, контури та історія, під нею — вікно відображення шарів;

3 — панелі Пензель/Текстури/Градієнт, що показують діалоги налаштування пензля, текстур, градієнту. Панелі можна переміщувати у вікні редактора за бажанням користувача;

4 — типи пензлів.

GIMP підтримує такі формати файлів, як GIF, JPEG, PNG, XPM, TIFF, TGA, MPEG, PS, PDF, PCX, BMP тощо.

Крім того, є рядок заголовка, рядок меню, вікно для зображень (розташоване посередині), унизу вікна може бути рядок стану.

Додаткові вікна встановлюють з набору ВІКНА — Діалоги з підтримкою прикріплення. Меню ВІКНА також містить назви всіх відкритих вікон.

У GIMP не можна відкрити або почати створювати зображення без вікна для нього. Разом з тим, можна відкрити багато вікон із зображеннями.

Зображення в GIMP може бути достатньо складним: уявіть стопку прозорих аркушів, на кожному з яких щось намальовано. Такі аркуші є шарами в редакторі (зона 2 на рис. 1). Зображення створюють з використанням шарів. Малювати можна на кожному шарі. Шари можуть бути прозорими й покривати не весь простір зображення, тому можна бачити не тільки верхній шар, а й частину інших, під ним. Шари можна міняти місцями. Досвідчені користувачі GIMP працюють з багатошаровими зображеннями.

У GIMP є три колірні режими зображення: *RGB*, *градації сірого* та *індексоване*.

Кольорове зображення в GIMP має три канали; колір точки на екрані, принтері або іншому пристрої виведення є результатом об'єднання кольору кожного каналу. У градаціях сірого працює один канал (колір змінюється в діапазоні 256 градацій сірого: від 0 — чорний до 255 — білий).

Індексоване зображення має обмежений набір кольорів: 256 або менше. Режим Індексований використовується у випадках, коли важливий розмір файлу або коли зображення має невелику кольорову гаму.

Щоб змінити колірний режим зображення, потрібно виконати команду ЗОБРАЖЕННЯ — Режими і вибрати необхідний режим. Колірні канали можна бачити в діалозі Канали (зона 2 на рис. 1).

Цифрове зображення складається з сітки квадратних елементів різного кольору, так званих точок (пікселів). У кожного зображення є розмір у точках, наприклад 900 точок у ширину і 600 точок у висоту. Але пікселі — точки зображення — не мають фізичного розміру.

Екран монітору, принтери, сканери мають **роздільну здатність** — характеристику, яка визначається в кількості точок на дюйм ppi (pixel per inch — точки на дюйм).

Точки пристроїв мають фізичні розміри (приклад).

Канал є одним компонентом кольору точки. Весь прямокутний масив одного з компонентів кольору для всіх точок зображення теж називається *каналом*.

Існує ще альфа-канал — додатковий канал, який відповідає за непрозорість. Діапазон значень альфа-каналу від 0 до 100%. Коли значення альфа-каналу певного місця шару дорівнює 0, це означає повну прозорість і тоді видно колір нижнього шару. Проміжні значення альфа-каналу дають колір як поєднання кольорів поточного і нижнього шарів.

Приклад.
Під час друку роздільна здатність принтера визначає фізичний розмір зображення на папері й, відповідно, розмір пікселів. Те саме зображення 900 на 600 точок можна роздрукувати з розміром 3 на 2 дюйма з ледь помітними точками або на великому плакаті з великими квадратними точками.

Зображення, узяті з цифрових фотоапаратів і мобільних пристроїв, зазвичай містять значення 72 або 96 точок на дюйм.

Запитання для перевірки знань

1. До якого типу графічних редакторів відносять GIMP?
2. Назвіть елементи вікна GIMP.
3. Як Ви розумієте «шар зображення» в GIMP?
4. Дайте визначення поняття «канал зображення».
5. Назвіть кольорові режими зображення в GIMP.
6. Чому піксель не має фізичних розмірів?

5.12. Інструменти малювання в GIMP та їх налаштування

Які команди містяться в меню ФАЙЛ будь-якого програмного засобу?

Для створення нового зображення в GIMP використовують команду ФАЙЛ — Створити: відкриється вікно налаштування параметрів зображення (рис. 1).

Вікно містить список Шаблони для зручного встановлення ширини й висоти зображення. Також їх значення можна встановити в області Розмір зображення із зазначенням одиниць вимірювання. Унизу містяться кнопки вибору орієнтації зображення, у GIMP цей параметр має назву Портрет/Ланшафт.

Якщо в меню ФАЙЛ вибрати підкоманду Створити (вона вказана без гарячих клавіш), відкривається вікно створення зображення з переліком команд, які залежать від операційної системи (команди використовують системні функції) (рис. 2).

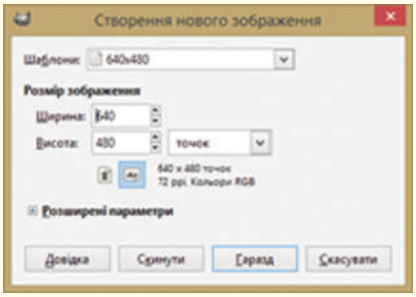


Рис. 1. Вікно налаштування параметрів зображення

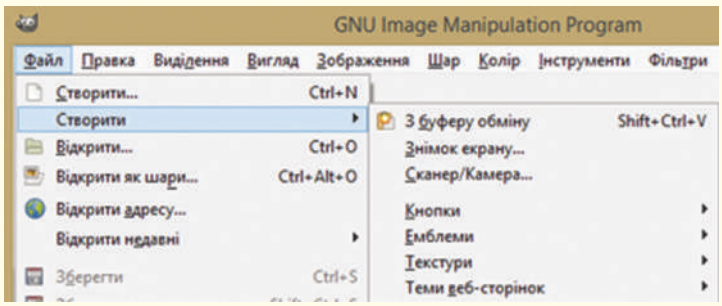


Рис. 2. Додаткова команда створення зображення

Панель інструментів GIMP містить 13 інструментів для малювання (рис. 3). Усі вони створюють зображення шляхом нанесення на тло кольорових крапок. Ці інструменти згруповані в меню Інструменти за призначенням у створенні загального зображення.

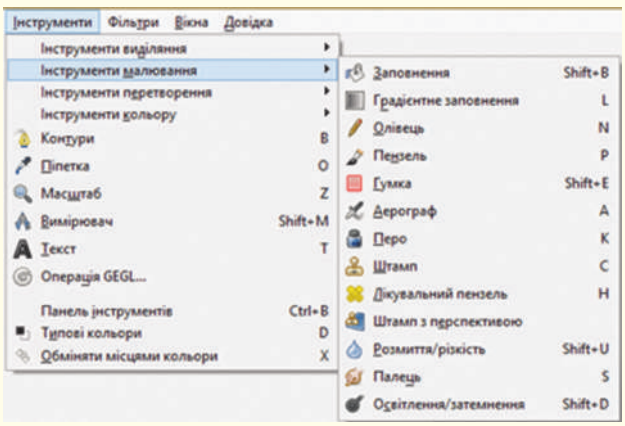


Рис. 3. Інструменти малювання редактора

Якщо ширину й висоту встановили однаковими, кнопки орієнтації будуть недоступні.

Розглянемо призначення інструментів малювання.

- Інструменти Олівець, Пензель, Аерограф і Перо створюють лінії під час переміщення миші з натиснутою лівою кнопкою. Решта використовується для модифікації наявних (уже намальованих або відкритих) зображень.
- Інструмент Заповнення заповнює замкнуту ділянку кольором або будь-якою текстурою. Для створення колірного переходу в межах замкненої ділянки є інструмент Градієнт.

Замкнена ділянка в растровому редакторі — виділена спеціальним інструментом область, або область, яка має суцільний колірний кордон.

- Інструмент Гумка стирає намальоване кольором тло.
- Інструменти Штамп, Штамп з перспективою, Лікувальний пензель виконують клонування фрагмента зображення і переміщують у вибране місце на зображенні. Використовуються для усунення невеликих дефектів зображення.
- Інструменти Розмиття/Різкість, Палець і Освітлення/Затемнення використовуються для створення ефектів переходу кольору або інших ефектів художнього перетворення вже намальованого.

Розглянемо *алгоритм малювання відрізка*.

Крок 1	Перемістіть вказівник миші в початкову точку відрізка
Крок 2	Натисніть клавішу Shift і з натиснутою ЛКМ перемістіть вказівник по ходу відрізка
Крок 3	Відпустіть мишу в кінцевій точці

Лінії малюють інструментами **Олівець**, **Пензель**, **Гумка**, **Аерограф**, **Перо**, **Штамп**, **Розмиття/Різкість**, **Палець** і **Освітлення**

Вигляд лінії залежить від налаштувань інструмента, яким її малюють. Так само створюють і ламані. Клавіша Ctrl дозволяє створювати відрізки під кутами, кратними 15. Після вибору інструменту вказівник миші має вигляд активного інструмента. Щоб намалювати лінію потрібного вигляду, слід вибрати в параметрах інструмент **Пензель** тощо.

Запитання для перевірки знань

1. Назвіть способи створення зображення в GIMP.
2. Що означає «замкнена область зображення»?
3. Назвіть інструменти малювання в GIMP.
4. Як вибрати колір малювання; тла?
5. Яким кольором *відбудеться* малювання після відкриття GIMP?
6. Спробуйте намалювати відрізок одразу після відкриття GIMP. Чому редактор не реагує на ваші дії?

Завдання для самостійного виконання

1. Запустіть редактор GIMP.
2. Перейдіть до значка вибору кольору під панеллю інструментів.
3. Клацніть верхній прямокутник — відкриється вікно вибору кольорів.
4. Виберіть колір малювання, наприклад, червоний, підтвердьте — **Гаразд**.
5. Клацніть нижній прямокутник і у вікні кольорів виберіть колір тла.
6. Клацніть подвійну стрілку — змінилися кольори малювання та тла.

5.13. Інструменти виділення в GIMP та їх налаштування

Пригадайте, які режими виділення об'єктів має Inkscape.

Часто в зображенні необхідно виокремити об'єкт, щоб зміни стосувалися тільки його і жодним чином не впливали на частину зображення.

Інструменти виділення призначено для відокремлення необхідної області зображення активного шару. Таких інструментів у GIMP сім (рис. 1).

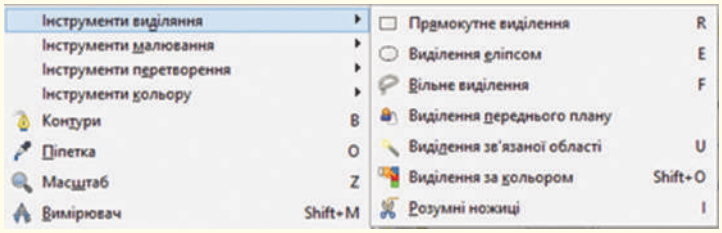


Рис. 1. Додаткова команда створення зображення

- Інструменти Прямокутне виділення і Еліпс виділяють ділянки, які відповідають формі інструмента. Щоб зробити кути прямокутника непрямими, до виділеної ділянки слід застосувати команду ВИДІЛЕННЯ — Закруглений прямокутник і налаштувати ступінь заокругленості.
- Інструмент Вільне виділення, або Ласо, нагадує обведення певної ділянки олівцем. Виділення переднього плану створює навколо виділеної області *маску* — напівпрозоре заповнення синього кольору. Края маски можна точно відкоригувати Пензлем та Гумкою.
- Інструменти Виділення зв'язної області, або Чарівна паличка, та Виділення за кольором виділяють області схожого кольору. Застосування інстумента Чарівна паличка надає розмите виділення. Інструмент Виділення за кольором від Чарівної палички відрізняється тим, що виділяють не лише суміжні області, а й всі області в зображенні кольором, на який наведено вказівник.
- Інструмент Розумні ножиці має таку назву завдяки розпізнаванню інструментом кольорових країв об'єкта зображення: за його допомогою виділяють область вручну, а інструмент визначає межі переходу одного кольору в інший.

Після того як інструмент виділення вибрано, змінюються команди області параметрів. Для зручності всі параметри виділення однакові для всіх інструментів, просто деякі з них застосовують тільки для окремих випадків. Загальними параметрами є режим згладжування та розмиття меж (рис. 2).

Інструменти перетворення та кольору розглянемо у процесі створення зображень у GIMP.

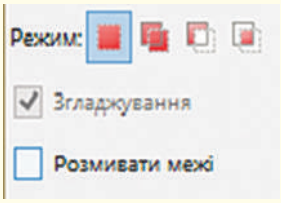


Рис. 2. Загальні параметри налаштування інструментів виділення

Команди **Згладжування** та **Розмивати межі** мають:
- режим заміни (замінює попереднє виділення на нове);
- режим додавання (додає нове виділення до існуючого);
- режим віднімання (віднімає нове виділення від існуючого);
- режим перетину (залишає загальну частину обох виділень).

Якщо зображення виявляється більшим за вікно редактора, GIMP відображає його у зменшеному масштабі: масштаб показано в рядку стану. Якщо встановити масштаб 100 % (масштаб можна вибрати у списку масштабів рядка стану), з'являться смуги прокрутки для повного перегляду зображення.

Зображення можна відкрити командою ЗОБРАЖЕННЯ — ФАЙЛ. Відкрите зображення часто заповнює все вікно редактора.

Розмір зображення можна змінити командою ЗОБРАЖЕННЯ — Розмір зображення. Відкривається вікно, у якому й налагоджують розмір зображення.

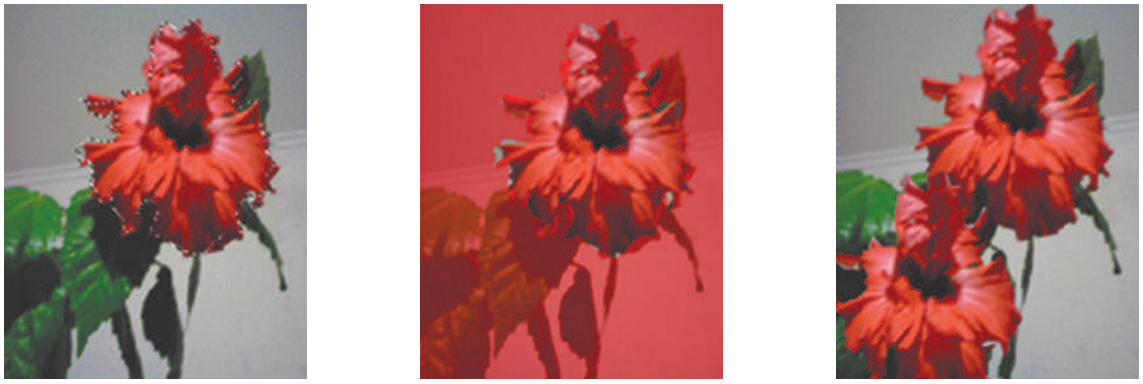
Після закінчення роботи в середовищі редактора необхідно зберегти зображення: ФАЙЛ — Зберегти або Зберегти як. Командою Вихід закриваються всі зображення і GIMP. Якщо є незбережені зображення, редактор їх покаже зі вказівкою Зберегти.

Приклад

Виділену ділянку зображення обведено пунктирним контуром (рис. 3, а). Для детального перегляду такої ділянки є кнопка швидкої маски, яка викликається командою **ВИДІЛЕННЯ — Перемкнути швидку маску** або клацанням кнопки в рядку стану.

Виділена область залишається такою, як і була в зображенні, решту вкрито червоним прозорим кольором (рис. 3, б). Зображення зі швидкою маскою можна зберегти для роботи, також інколи зручніше в режимі швидкої маски копіювати виділене в інше зображення. У режимі швидкої маски зручно коригувати виділену область: якщо по виділеному почати малювати будь-яким інструментом малювання, наприклад, **Пензлем**, чорним кольором, виділені місця малювання видаляться з виділеного, а білим можна додати до виділення інші ділянки.

Виділену ділянку зображення можна зробити прозорою (команди **ПРАВКА — Очистити**), переміщувати, змінювати розміри. Якщо під час переміщення такої ділянки утримувати сполучення клавіш Shift+Alt (рис. 3, в), то відбудеться додаткове переміщення, а якщо CTRL+ Alt — з переміщенням зникає початково виділена область, у різних версіях GIMP сполучення клавіш може бути іншою. В останніх випадках створюється так зване плаваюче виділення. Щоб працювати із зображенням і надалі, плаваюче виділення можна зняти клацанням у будь-якій невиділеній ділянці зображення.



а б в

Рис. 3. Опрацювання виділених областей

Запитання для перевірки знань

1. Для чого в GIMP є команди виділення?
2. Назвіть інструменти виділення в GIMP.
3. Поясніть, що таке швидка маска.
4. Чи може бути в одному зображенні кілька областей виділення?
5. Назвіть загальні параметри елементів виділення.
6. Який алгоритм виділення тла у зображенні з об'єктом?

5.14. Шари. Створення колажу

Поясніть поняття шарів у редакторі GIMP на прикладі прозорих аркушів.

Структура шарів зображення відслідковується в діалозі Шари. Під час відкриття GIMP діалоги Шари і Пензлі буде подано у вікні редактора.

Вікно Шари також можна викликати через меню **ВІКНА** командою Діалоги з підтримкою прикріплення і вибрати діалог Шари — на рис. 1 наведено вікно щойно відкритого порожнього зображення.

У вікні Шари відображаються шари зображення: маленькі мініатюри й назви. При створенні нового зображення, воно складається лише з одного шару — тла. Решта доданих шарів будуть прозорими. Опрацювання прозорості шарів здійснюється роботою з альфа-каналом.

Активним шаром є шар, у якому відбувається створення та опрацювання зображення. Він виділений кольором, активним стає після подвійного клацання ЛКМ.

Шари мають назву, автоматично надається назва: шар 1, шар 2 і т. д. Назву шару можна змінити. Особливо це важливо для складного зображення, кожному шару краще за зображенням на ньому.

Щоб змінити назву шару, слід виконати подвійне клацання ЛКМ і ввести нову, або з контекстного меню шару вибрати команду Правка ознак шару.

Якщо клацнути кнопку між зображеннями ока й мініатюрою шару, з'явиться зображення ланцюжка. Його призначення — згрупувати шари для операцій із кількома шарами.

У вікні налагоджують режими зображення шару (ми розглянемо їх далі), ступінь непрозорості, а також можна замкнути шар.

Назва активного шару відображена в рядку стану. У нижньому рядку вікна кнопки команд опрацювання шарів дозволяють:

- створити новий шар до зображення;
- підняти активний шар на рівень;
- опустити активний шар на рівень;
- створити копію активного шару та додати її до зображення;
- прикріпити плаваючий шар (ще одна можливість зняти плаваюче виділення);
- видалення активного шару (видалити шар ще можна перетягуванням його у область корзини).

Для опрацювання шарів призначено меню ШАР. У меню є команда Шар до розміру зображення. В одному зображенні розмір шарів може бути різний. Регулювання розміру і розміщення шару здійснюється командою Розміри меж шару, переміщення шарів і виділення інструментом Переміщення. Виконайте вправу для опрацювання зображень в шарах і операцій з шарами.

Зображення в GIMP зручно уявити як стос прозорих аркушів — **шарів**. На кількість шарів обмежень немає, усе залежить від доступної пам'яті в системі.

Рис. 1. Вигляд вікна Шари

Зображення ока означає можливість переглянути шар. Для зняття режиму перегляду шару (вилучення зображення ока) потрібно по цьому зображенню клацнути ЛКМ. Якщо клацнути з натиснутою клавішею Shift, для перегляду залишиться шар, який клацнули.

• Режими шарів у GIMP

Окрім характеристик шару, з якими ви ознайомилися, є такі, як прозорість і режим. Зазвичай під час роботи в полі Режим вибрано Звичайний. Це означає, що зображення в редакторі створюється як на папері (нижні шари не видно).

Щоб побачити зображення нижнього шару, потрібно регулювати прозорість поточного. Уявіть малювання на склі: його можна фарбувати, щоб малюнок був непрозорий або напівпрозорий, або ледь помітний. Але через малюнок певною мірою проглядається все, що є за склом.

Регулювання прозорості шару аналогічне малюванню на склі. Для регулювання позорості користуються бігунком значень непрозорості діалогу Шари: 100 % — шар непрозорий, а 0 % — повністю прозорий.

Зображення зазвичай складається з кількох шарів, і через частково непрозорий шар видно зображення нижніх шарів. Вони накладаються (верхнє на нижнє) і внаслідок змішування кольорів їх зображень отримуються цікаві малюнки. Растрові редактори мають режими змішування. У GIMP їх 21. Для зображення лише з одним шаром вибір режиму ні на що не впливає. Режими вибирають із поля режими діалогу Шари (див. рис. 1).

Колаж (від франц. Coller — приклеювання) — прийом в мистецтві, поєднання в одному творі різнорідних елементів (різних за походженням, контрастних за стилем тощо). Колажі характерні для мистецтва XX ст. Процес створення колажу полягає в створенні мальованих або графічних зображень шляхом наклеювання на будь-яку основу предметів і матеріалів, що відрізняються від основи за кольором і фактурою.

Запитання для перевірки знань

1. Що таке шар зображення?
2. Як переглянути шари зображення в GIMP?
3. Які властивості перегляду шарів є у вікні **Шари**?
4. Поясніть, що таке плаваюче виділення.
5. Які команди роботи з шарами наведені у вікні **Шари?**
6. Які команди з шарами здійснюють в межах стопки шарів?

Завдання для самостійного виконання

1. Відкрийте редактор і запустіть команду створення нового зображення.
2. Інструментом виділення виділіть прямокутну ділянку — вертикальну смужку.
3. Виберіть червоний колір і інструментом заповнення зафарбуйте смужку.
4. Створимо смуги з кольорами веселки різними способами:

   Спосіб 1: працюємо з копією зображення. Для цього виконайте такі дії.

   1) Скопіюйте виділену область. Створіть новий шар, зробіть його активним і вставте смужку на нього.
   2) Перефарбуйте смугу на помаранчевий колір.
   3) Смужку перемістіть праворуч від червоної. У вас з'явилось плаваюче виділення, яке розташоване на тимчасовому шарі поверх стопки шарів.
   4) Для плаваючого виділення прикріпляють шар одноіменною командою з меню **ШАР**. Зверніть увагу на наявність шарів після прикріплення. Також прикріпити можна створивши новий шар — плаваюче виділення буде на новому шарі.
   5) Змініть назву шару на помаранчевий.

   Спосіб 2: працюємо з копією шару. Для цього виконайте такі дії.

   1) Створіть копію попереднього шару, перефарбуйте смужку в жовтий колір, перемістіть її, прикріпіть шар. Шар має бути виділений синім кольором.
   2) Виконайте команду **ШАР — Об'єднати з попереднім** і надайте шару нову назву П Ж (помаранчевий і жовтий).

Завдання для самостійного виконання

5 Аналогічно способу 2 створіть шари З Б із зеленою та блакитною смужками та С Пурпуровий із синьою та пурпурною смужками. Якщо у вас увімкнутий перегляд усіх шарів, бачите кольори веселки (рис. 2).

6 Попрацюємо з шарами. Для цього виконайте такі дії.

1) Виконайте команду **ШАР — Стопка шарів — Зворотній порядок шарів**. Чому ви бачите тільки червону смугу? Відмініть команду з меню **ПРАВКА**.

2) Поекспериментуйте з командами **Стопки шарів**. Зверніть увагу на розташування шарів. Чого не відбуваються зміни в зображенні? Як прибрати показ якихось смуг?

3) Виконайте команду **Зображення — Об'єднати видимі шари**. Як називається єдиний шар створеного зображення?

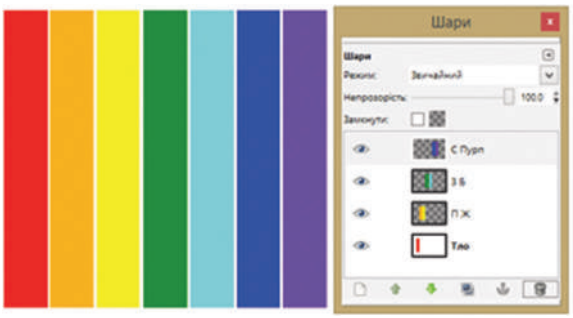


Рис. 2. Вигляд зображення і вікна **Шари** для нього

5.15. Редагування зображень у GIMP

Які можливості щодо редагувань зображень мають офісні програми, Inkscape?

У процесі створення зображень може виникнути потреба у перетворенні як всього зображення, так і його частини. Для перетворення зображень використовують інструменти перетворення (рис. 1).

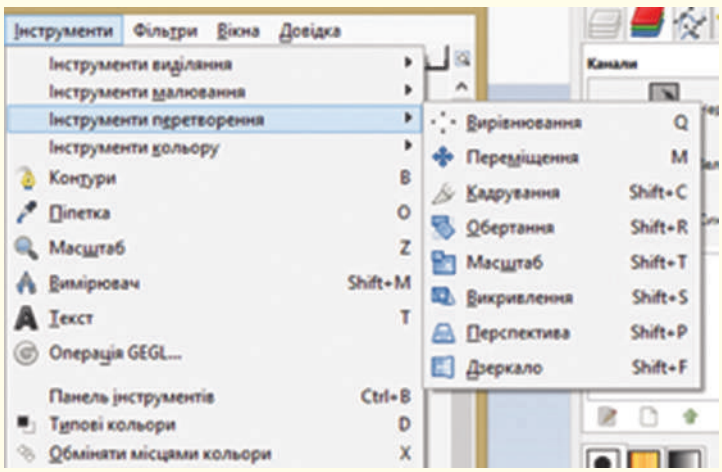


Рис. 1. Команди перетворення в GIMP

Виклик будь-якої команди супроводжується відкриттям відповідного їй вікна для налагодження параметрів перетворення. Вікно відкривається й у разі активації інструмента.

Із деякими командами і відповідними їм інструментами ви вже познайомились. Оскільки GIMP часто використовують для виправлення якості цифрових фотографій, розглянемо процес перетворення готових зображень.

- Обертання — здійснюється інструментом Обертання і при натисканні миші з'являється сітка, яка обертатиметься за рухом миші. Коли сітка виглядає так, як потрібно, слід натиснути клавішу Enter і зображення буде обернене. У процесі обертання з'являється вікно **Обертання** (рис. 2), у якому відображено маніпуляції із зображенням. Підтвердити результат можна кнопкою **Повернути**.

Після обертання зображення утворюються трикутні порожнини по кутах зображення. Їх можна залити нейтральним кольором, але якщо кут обертання невеликий, краще використати кадрування.

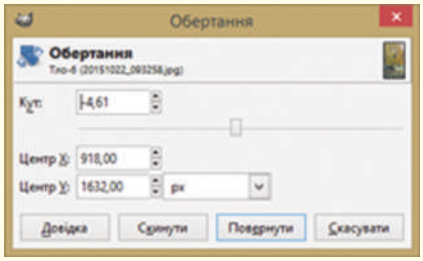


Рис. 2. Вікно налагодження обертання

Приклад

На рис. 4 показано зображення (А), його вигляд після обертання (Б). На якість фотографії не звертайте уваги — у подальшому попрацюємо над її покращенням.

а б в г

Рис. 4. Результат перетворення зображення (обертання і кадрування)

У наведеному прикладі розглянули обертання на довільний кут, але редактор містить команди віддзеркалення та обертання на визначений кут у меню ЗОБРАЖЕННЯ — Перетворення (рис. 3).

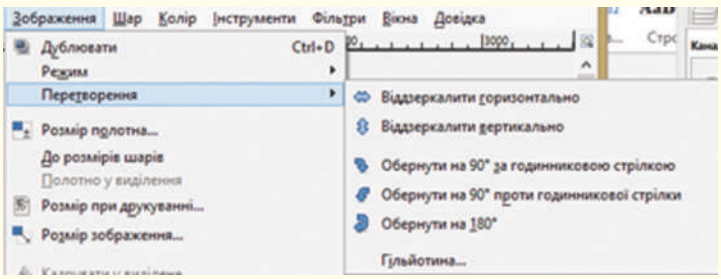


Рис. 3. Команди перетворення меню **ЗОБРАЖЕННЯ**

Режим кадрування встановлюють командою **Кадрування** в меню **ІНСТРУМЕНТИ — Інструменти перетворення** або активізацією інструмента кадрування.

Для обертання кількох шарів зображення їх попередньо необхідно зв'язати встановленням ланцюжка зліва від мініатюри шару у вікні **ШАРИ**.

- **Кадрування.** Кадруванням є процес вилучення (відрізання) областей зображення.

В обох випадках інструмент активний, а під панеллю інструментів з'являються команди налагодження його параметрів (рис. 5). Установлюють прапорці.

- Для кадрування активного шару придатний лише поточний шар.
- Дозволити зростання, якщо необхідно «наростити» результуюче зображення прозорою областю, оскільки рамка кадрування може вийти за межі зображення.
- Команду Малювати з центру застосовують, якщо необхідно встановити розміри кадрування. Початковою вважається точка встановлення вказівника в зображенні. Від неї розраховується розмір, як від центра.
- Командою **Фікс** встановлюють співвідношення сторін зображення, конкретне значення ширини або висоти зображення.Для кадрування зображення вказівник миші встановлюють в лівий кут зображення, затискують ЛКМ і переміщують вказівник до нижнього правого кута — так створюється виділена прямокутна область кадрування. Зображення береться в рамку і переміщенням кутів цієї рамки (вони виділені пунктиром) урізають зображення. Зміна області кадрування, розмір і пропорції фіксуються в рядку стану.

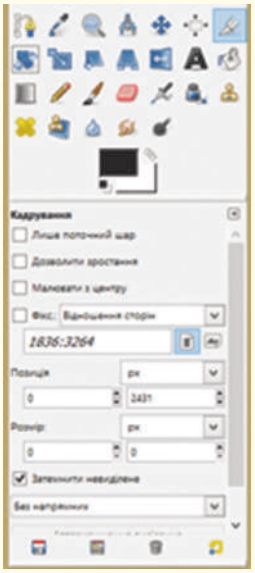


Рис. 5. Налаштування параметрів інструмента **Кадрування**

У меню Зображення вказано кілька режимів кадрування:

- Кадрувати у виділене — застосовується, якщо ви хочете мати тільки виділену область зображення як окреме зображення; команда урізає всі шари, які складають зображення.
- Автокадрування — урізання країв зображення активного шару: визначення кольору тла і обрізання до зображень, кольори яких відрізняються від фонового.
- Старанне кадрування — вирізання із зображення ділянок одного кольору, аналізується колір активного шару а урізання відбувається для всіх шарів.

Пригадайте призначення напрямних і сітки в Inkscape.

У меню є команда Гільйотина, яку використовують для розрізання зображення на частини вздовж направляючих, встановлених користувачем. У процесі розрізання напрямними встановлюють межі частин зображення, запускають команду Гільйотина і GIMP відкриває стільки вікон зображення, скільки було частин у зображенні. Надалі кожну частину можна зберегти в окремому файлі.

Змінити форму виділеної ділянки або шару дозволяють команди Викривлення, Перспектива і Дзеркало.

Завдяки параметрам налагодження інструмента кадрування можна не користуватись маніпулятором миша, а налагодити встановлення початкової позиції і розмір результуючого зображення.

Запитання для перевірки знань

1. Які команди застосовують для перетворення зображень?
2. Що означає операція кадрування зображення?
3. Як можна здійснити кадрування в GIMP?
4. Які проблеми можуть виникнути після застосування обертання до зображення?
5. Яке призначення і результат відпрацювання команди **Гільотина**?
6. Що потребує особливої уваги при використанні різних режимів кадрування?

5.16. Канали. Корекція кольору та тону

Як можна змінити кольорову гаму зображення в Inkscape?

Важливою характеристикою зображення є якість. Фотографії, наприклад, можуть бути занадто затемненими, нечіткими, неправильно передавати колір, мати дефекти (особливо старі фото) та ін.

Розрізняють *глобальні* та *локальні* дефекти зображень. Для **усунення глобальних** дефектів, таких як низька контрастність, колірні спотворення, зернистість, використовують *тонове* (налагоджують яскравість, контрастність тощо) та *колірне* (змінюють та налаштовують колірну гаму) коригування. **Локальні дефекти**, які може мати зображення, — подряпини, плями, «червоні очі», можуть частково зникнути під час усунення дефектів усього зобраення.

Розглянемо можливості GIMP для кадрування зображень.

Розглянемо **коригування за допомогою рівнів**. Інструмент **Рівні** використовують для освітлення й затемнення зображень, зміни контрасту та коригування відтінків, що переважають.

Вікно **Коригування рівнів кольорів** містить гістограму **Рівні на вході**. Тон кольору розташовано зліва (темні) направо (світлі). Якщо графік має вигляд зафарбованої ділянки, це вказує на наявність у зображенні майже всієї гами відтінків. Рівні входу використовуються для освітлення світлих областей (яскравих тонів), затемнення тіней (темних тонів), зміни балансу світлих і темних тонів.

Для регулювання освітлення в каналі слід установити **Яскравість** і відрегулювати її переміщенням маркерів: лівим (чорним) — темні тони, правим (білим) — світлі, середнім (сірим) — проміжні. Достатньо «захопити» вказівником миші маркер на горизонтальній вісі й із натиснутою ЛКМ перемістити. Відразу змінюється освітленість зображення.

Послідовність операцій для покращення якості зображення

1) за потреби зображення кадрують;
2) здійснюють тонове та колірне коригування;
3) за потреби ретушують фото;
4) установлюють розмір зображення, тип відповідно до його призначення;
5) після виконання операцій (п. 4) часто збільшують різкість зображення.

Гістограма є графіком розподілу кількості пікселів (вертикальна вісь) по колірних тонах (горизонтальна вісь).

Приклад

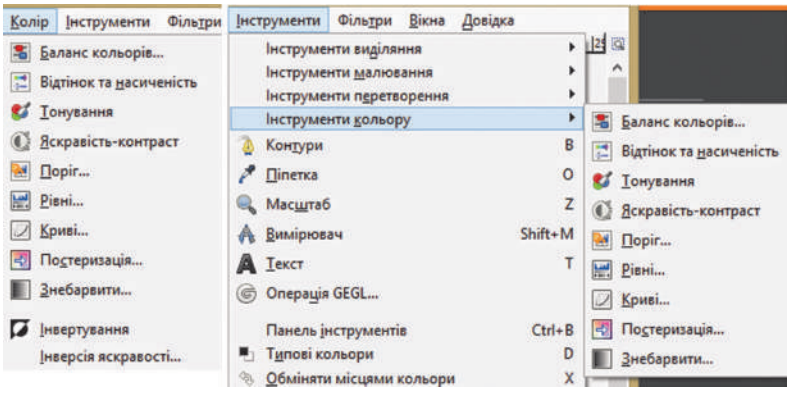


Рис. 1. Інструменти кольору в меню **КОЛІР** і **ІНСТРУМЕНТИ**

На РИС. 5.46 видно, що фотографія затемнена. Для коригування освітленості і кольорової гами зображення GIMP пропонує інструменти кольору в меню **КОЛІР** та **ІНСТРУМЕНТИ** (рис. 1).

Приклад

На рис. 2 показано вигляд гістограми та зображення після використання чорного й білого маркерів. Освітленість також регулюють переміщенням середнього маркера в бік темних або світлих тонів.

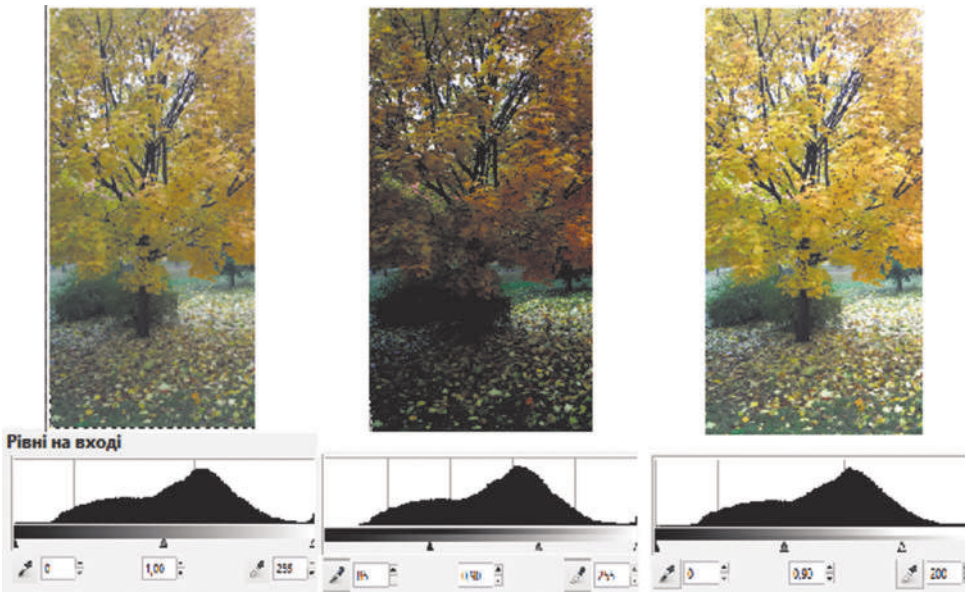


Рис. 2. Результат тонового коригування зображення ?

Значення можна вибирати маркерами, за допомогою пипеток і встановленням значень.

Рівні на виході здійснюють колірне коригування: можна також використовувати маркери або проставляти значення, й колірна гама зображення змінюватиметься. Окремі кольори можна скоригувати, якщо вибрати їх у списку **Канал**. Командою **Відновити канал** можна повернути зображення до попереднього стану.

Розглянемо **коригування за допомогою кривих**. Інструмент виклику методу тонових кривих **Криві** — найбільш складний інструмент, призначений для здійснення зміни кольору, яскравості, контрасту або прозорості в активному шарі або виділенні.

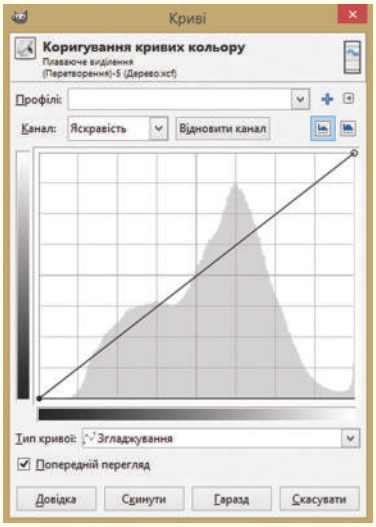


Рис. 3. Вікно коригування кривих кольору

У вікні, що розкриється (рис. 3), горизонтальна вісь на графіку визначає тональні рівні на вході, вертикальна вісь — на виході. Вказівник миші перетворюється на піпетку. Графік є лінією — кривою, проведеною з лівого нижнього кута в правий верхній.

Спочатку графік є прямим відрізком, кожен рівень на вході відповідає тому самому значенню рівню на виході. GIMP автоматично проставляє вузли на кінцях кривої: для чорного (0) і білого (255) тонів. Показчик значень координат постійно міститься в лівій верхній частині сітки, за його значеннями віддразу видно, як змінюється тон вихідного зображення відносно початкового.

Регулювання яскравості або кольорового тону (вибирають із списку) вихідного зображення здійснюється новими вузлами на графіку. Слід клацнути ЛКМ криву, потім переміщення вузла догори вниз змінює вигляд кривої і додає / зменшує тон на зображенні. Можна поставити два вузли — так визначиться сегмент кривої, яку теж переміщують.

Ознайомимося з правилами тонового коригування методом кривих.

- Підняти вузол — додати в даній точці світла.
- Зробити криву більш горизонтально — звузити вихідний колірний діапазон зображення.
- Переміщення нижнього вузла вправо та верхнього вліво аналогічне переміщенню чорного та білого маркерів на гістограмі **Рівні**.

Переміщення білого маркера дає такий результат: всі точки з тоном, більшим за встановлений білим маркером, стають білими. Аналогічно працює переміщення чорного маркера. Крива має інший кут нахилу, а зображення стає більш контрастним.

- Збільшити контрастність — вигнути криву: зліва від середини опустити, а справа підняти.
- Інвертувати кольори — перемістити правий вузол у верхню точку, а лівий у нижню.

Виконаємо вправу. Ви можете вибрати будь-яке зображення й проаналізувати, якзмінюватиметься вигляд зображення залежно від ваших дій.

Якщо «взяти» вказівником колір у певній точці зображення, на графіку з'явиться вертикальна пряма, яке перетинає вісь входу саме в точці значення каналу.

Для перегляду результатів опрацювання зображення має бути прапорець біля команди **Попередній перегляд**. Щоб завершити роботу з коригуванням, треба клацнути кнопку **Гаразд**.

Запитання для перевірки знань

1. Поясніть поняття каналів у GIMP.
2. Які є загальні дефекти фотографій?
3. Поясніть різницю між тоновим та колірним коригуванням.
4. Як коригують яскравість у вікні **Рівні**?
5. Як коригують зображення командами вікна **Криві**?
6. До яких шарів застосовується тонове коригування?

Завдання для самостійного виконання

1. Відкрийте зображення у GIMP.
2. Запустіть команду КОЛІР — **Криві.**
3. Аналізуємо графік: на гістограмі (див. рис. 3) видно, що лівий чорний маркер розташований окремо від самої гістограми. Це означає відсутність певної гами кольорів в зображенні внаслідок чого зображення є розпливчатим. Чорний маркер необхідно перемістити ближче до основних присутніх тонів. Зображення стає більш контрастним і темнішим.
4. Додайте приблизно посередині прямої вузол і трохи підніміть його і, як наслідок, пряму. Відслідковуйте зміни, щоб не з'явилося занадто багато білих плями на листі під сонцем.
5. Виберіть кольорові канали й поекспериментуйте з кривими, вузлами, відслідковуючи зміни в зображенні.
6. Підтвердьте зміни кнопкою Гаразд і збережіть зображення.

5.17. Коригування зображення. Інструменти ретушування в GIMP

Першою в історії людства фотографією був кадр «Вигляд з вікна» французького винахідника-фотографа Ж. М. Ньепса (1826 р.).

Льюїс Керрол (Доджсон), автор усім відомих пригод Аліси, досяг великих результатів в справі фотографування та опрацювання фото. Разом із Генрі Пічем Робитсоном та Оскаром Гюставом Рейландером він вважається одним із найкращих фотографів XIX ст.

У 1850-х років почав розвиватися художній спосіб ретуші. Колись фотомайстер на негативі вичищав задній план до білого фону, на відбитку фотографії малював мальовничий пейзаж, інтер'єр тощо. Так змінювали тло фотографії.

Завдяки параметрам налагодження інструменту **Штамп** можна виконувати різноманітні операції, а також автоматизувати багаторазове штампування.

Рис. 1. Вікно коригування кривих кольору

Часто на фотографіях з'являються плями, наприклад, на об'єктив може потрапити пил тощо. Це локальні дефекти. Щоб їх уникнути, використовують можливості редактора з опрацюванням локального місця зображення.

Ретушування — процес усунення дефектів фотографій: плям, подряпин, дефектів зйомки, ефекту «червоних очей» тощо.

Із розвитком фотографування стали звертатися до корекції фото. Метод монтажу фотографій був винайдений французьким фотографом Іполитом Байярдом. Колажі робилися з негативів або готових фотографій, які потім перезнімалися або передруковувалися. З'єднання негативів часто виконувалося для досягнення особливого художнього ефекту.

Розглянемо **інструменти, принцип дії яких полягає у клонуванні** нормально зображених ділянок фото — зразкових, та накладання їх на ділянки з дефектом. Щоб уникнути невідповідності кольорів, слід вибрати зразок поблизу ділянки з дефектом.

- Інструмент Штамп використовують для копіювання однієї частини зображення в іншу. Під час усунення локальних дефектів зображення з використанням **Штампа** важливо вибрати ділянку для вставлення її копії на потрібне місце. **Штамп** має такі самі параметри, які і **Пензель.**

Щоб **скопіювати зразок**, слід навести вказівник в необхідне місце, утримуючи клавішу **Ctrl**, і клацнути ЛКМ — зразок скопійовано. Після виділення зразка слід відпустити клавішу Ctrl, перемістити вказівник миші на місце з дефектом і клацнути. Якщо яскравість і колір у місцях зразка і клонування однакові, дефект усувається непомітно. Таку операцію здійснюють стільки разів, скільки потрібно для усунення дефекту.

- Параметр Джерело. Зразкову ділянку як з існуючого зображення, так і з інших — текстур — вибирають **Штампом**. Вибір здійснюється налагодженням параметру **Джерело**. Зразок береться з будь-якого шару існуючого зображення. Цей шар має бути активним. Утримуючи клавішу Ctrl, беруть зразок, активним роблять шар для вставлення і вставляють зразок.

Якщо вибрати як джерело текстуру, й активізувати мініатюру текстури, відкриється вікно із зразками текстур. Зразок вибирають клацанням ЛКМ. На рис. 1 показано вставлення у правий прямокутник зразка з лівого прямокутника поточного шару, з іншого шару (зелений колір) та з текстури.

- Параметр Вирівнювання має кілька значень. Не можна просто взяти зразок і вставити, його відповідає значенню немає вирівнювання.

Приклад

На рис. 2 у випадку А показано зразкову ділянку й ділянку відпрацювання штампу, в випадкуБ що штамп перемістили на блакитному прямокутнику, й одразу змістилася зразкова ділянка на синьо-червоному прямокутнику; аналогічно відбулось і в зображенні в випадку В. Отже, встановленням вирівнювання взяли один раз зразок і використали 4 рази його штампування (рис. 5.43, г). Команда **Вирівнювання З реєстрацією** фіксує шар для зразка.

а б в г

Рис. 2. Результат використання штампу з вирівнюванням

Фіксоване вирівнювання дозволяє багаторазове вставлення з одного зразкового місця: один раз взяли зразок і вставили кілька раз його.

Для багаторазового використання штампу краще працювати з вирівнюванням. У редакторі запам'ятовується відносне положення зразка і першого місця вставлення. Надалі не беруть зразок, а просто переміщують штамп і клацають для вставлення — буде вставлено зразок з таким же відносним розташуванням: автоматично зміститься так же зразок і вставлена ділянка буде відповідати новому зразку. (Працює цей режим вирівнювання як багаторазове вставлення клітинки з відносним посиланням в Excel.)

- Лікувальний пензель. За своєю дією інструмент схожий на **Штамп**, але додатково коригує результат відповідн до тону одного кольору пікселів у місці вставлення. Редактор аналізує колір пікселів навколо області дії інструменту та підлагоджує колір зразка до кольору ділянки для вставлення. В момент виділення зразка вигляд інструмента набуває вигляду піпетки.

Як вже зазначалось, інструменти **Штамп** і **Лікувальний пензель** використовують поточні налагодження **Пензля**. У вікні налагодження параметрів можна змінити і вигляд пензля, і його розміри. Щоб уникнути різких переходів кольорів, також можна проставити прапорець біля команди **Поступовий перехід**, вибравши режим **Розчинення**.

На панелі інструментів є ще один штамп — **Штамп з перспективою**. Виконайте практичну роботу для знайомства з вказаним інструментом, а також з інструментом перетворення **Перспектива**.

Запитання для перевірки знань

1. Поясніть алгоритм роботи з інструментом **Штамп**.
2. Назвіть типи зразків для штампування.
3. Які можливості має редактор для багаторазового штампування зразків?
4. Поясніть різницю між інструментами **Штамп** і **Лікувальний пензель**?
5. Які зміни відбуваються із зображенням після використання інструмента **Перспектива**?
6. Який алгоритм використання інструменту **Штамп з перспективою**?

5.18. Фільтри. Інструмент Текст

Деякі фільтри не мають параметрів і застосовуються для зображення одразу після виклику.

Для вибору фільтрів користуються пунктом меню **ФІЛЬТР**. Після вибору необхідного фільтру може відкритися вікно налаштування параметрів цього фільтру. У вікні буде показано зміни зображення після застосування фільтра (якщо встановлено прапорець біля команди Перегляд), але дії фільтра будуть застосовані до зображення тільки після запуску команди **Гаразд**.

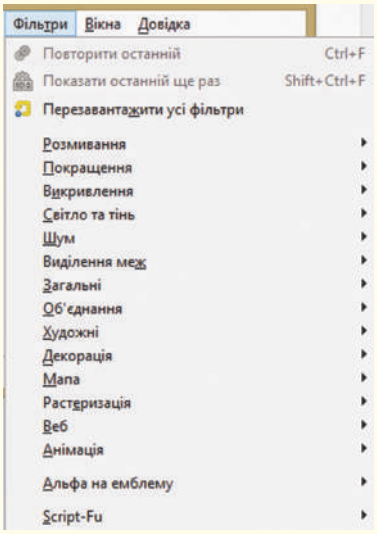


Рис. 1. Фільтри GIMP

Gimp має набір фільтрів для застосування спеціальних ефектів перетворення зображень активного шару. Використання кількох фільтрів дозволяють зі звичайного зображення отримати твір мистецтва.

Фільтр — це засіб перетворення зображення активного шару із застосуванням спеціального математичного алгоритму. Результат виконання фільтру — новий вигляд зображення.

Редактор Gimp містить такі **групи фільтрів** (рис. 1).

- Розмивання — викликає зменшення різкості переходів між кольорами.
- Покращення — усуває дефекти зображення (підвищити різкість, видалити ефект червоних очей тощо).
- Викривлення — перетворює вигляд зображення різними способами.
- Світло і тінь — містить три групи фільтрів: фільтри світлових ефектів для ефектів освітлення зображення, фільтри для створення тіней і фільтри ефекту скла. Останні перетворюють зображення так, ніби його переглядають крізь лінзу або скляні блоки.
- Шум — додає шумові ефекти в зображення: додаються пікселі з випадковими колірними значеннями, що ніби пом'якшує вигляд зображення.
- Виділення меж — використовуються для встановлення меж виділення та для інших художніх цілей. Межею є межа між різними кольорами зображення.
- Загальні — містить фільтри, які не можна віднести до інших груп фільтрів.
- Об'єднання — дозволяють об'єднати кілька зображення певним способом.
- Фільтри Художні, Декорація, Мапа, Растерізація перетворюють зображення із застосуванням художніх ефектів (комікс, кубізм, малювання маслом тощо) та з додаванням додаткових елементів, тривимірних ефектів.
- Веб — дозволяє підготувати зображення для інтернет сторінки.
- Анімація — дозволяє переглянути й оптимізувати анімаційне зображення.

Процес опрацювання зображень не завжди завершується бажаним результатом. Особливо важливо скасувати дії із зображенням у процесі налаштування його якості та використання фільтрів, оскільки не завжди досягається бажаний результат.

Щоб **скасувати застосування до зображення** останніх дій, використовується команда **ПРАВКА — Вернути**, біля якої

вказана остання дія. Передбачено й повернення відміненої дії: **ПРАВКА — Повторити** і команда останньої скасованої дії.

Діалог історії дій входить до стандартного набору відображення вікон, також викликається з меню **ПРАВКА** або з **ВІКНА — Діалоги** з підтримкою прикріплення. У відповідь відкривається вікно **Відмінити** із назвами дій із зображенням. Знизу вікна містяться кнопки, що дублюють команди **Вернути** (виділену дію), **Повторити** та **Очистити історію дій**. У верхній частині вікна біля назви **Історія скасування дій** стрілкою відкривають налаштування команд історії.

Основне призначення діалогу — вибір точки в історії дій, у яку треба повернутися. Це дає можливість перейти до будь-якого моменту роботи із зображенням. Обмежень на кількість переходів немає.

Ви вже виконували завдання з покращення вигляду фотографій. Деякі фільтри вирішують такі самі проблеми. Це фільтри **Групи Покращення: Покращення різкості, Прибирання плям, Прибирання штрихів.** Під час зйомок у приміщенні на фото очі можуть набувати червоного кольору. Їх необхідно виділити та застосувати фільтр **Покращення — Прибрати ефект червоних очей.**

Ознайомимося з інструментом Текст.

Після активізації інструменту **Текст** вказівник миші має вигляду текстового. Клацання мишею в області зображення створює новий текстовий шар. Після цього можна вводити текст, автоматично відкривається вікно текстового редактора. Налагодження кольору текста, шрифта, розміру, типу начертання встановлюється параметрами інструмента.

Виконайте практичну роботу, в якій створіть напис з літерами у рамці та застосуйте ефекти фільтрування до напису.

Запитання для перевірки знань

1. Що таке фільтр у GIMP?
2. Назвіть типи фільтрів.
3. Які можливості має редактор для повернення однієї або кількох дій?
4. Як застосуватия фільтри до зображення?
5. До якого місця зображення застосовуються фільтри?
6. Які можливості має GIMP щодо застосування тексту в зображенні?

Завдання для самостійного виконання

1. У редакторі створіть вигляд жовто-блакитного прямокутного прапору.
2. Виділіть зображення, запустіть команду **ФІЛЬТРИ — Викривлення — Згин за кривою**. Працюємо у вікні фільтру.
3. Виберіть команду опрацювання верхнього краю виділеного зображення.
4. Вікно містить графік, подібний до графіка **Кривих**. Проставляйте на графіку вузли та їх переміщенням змінюйте його вигляд. Змініть коригуючу криву і для нижнього краю. (Активна крива зображена чорним, неактивна — білим). Підтвердьте — **Гаразд**.
5. У вікні шарів видно створення шару з плаваючим виділенням, а початкове зображення залишилось незмінним. Збережіть шар із виділенням у зручний для вас спосіб. Збережіть роботу у файлі **Прапор**.

5.19. Комп'ютерна анімація

Які види графіки ви знаєте? Поясніть поняття інтерполяції. Яка буває інтерполяція?

Пригадаємо, що комп'ютерна анімація є видом мультиплікації, в якій для створення на екрані дисплею рухомого об'єкта використовують апаратне та програмне забезпечення комп'ютера.

Анімація (латин. *Animare* — оживити) — вид мистецтва, твори якого створюються шляхом покадрової зйомки окремих малюнків або сцен. Анімацію часто називають «мультиплікацією» (латин. Multiplicatio — множення, розмноження).

Кадри — намальовані або сфотографовані зображення послідовних фаз зміни об'єктів або їх частин.

Під час перегляду послідовності кадрів виникає ілюзія руху зображених на кожному з них статичних об'єктів. Залежно від промальованих у кадрах об'єктів для їх динамічних змін виділяють покрокову та покадрову (або автоматичну) комп'ютерну анімацію.

Промальовані кадри називають ключовими. В покроковій анімації кожний кадр є ключовим. У автоматичній анімації ключовими є пари кадрів для певного руху: створюється інтерполяція вигляду зображення між двома ключовими кадрами (таких пар ключових кадрів може бути кілька).

У покроковій анімації кожний кадр містить окрему фазу руху, в автоматичній — промальовуються початкове й кінцеве положення певного руху, а проміжні кадри заповнюються зображенням автоматично. Для автоматичної анімації необхідно спеціальне програмне забезпечення.

В основі будь-якої анімації лежить фіксація фаз зміни об'єктів — визначення в кожен момент часу їх положення, форми, розмірів та інших властивостей, наприклад, кольору.

Частота зміни кадрів — кількість кадрів на секунду для сприйняття людиною зміни розміщення і форми об'єкту зображення або його частин.

Частота змінення кадрів для створення ефекту плавної зміни має бути не менше ніж 12–16 кадрів на секунду, це пояснюється особливостями сприйняття руху людиною. У кіно використовується частота 24, в телебаченні 25 або 30 кадрів в секунду.

Зазвичай програми, призначені для створення анімаційних зображень, наприклад Flash, містять команди виклику часової шкали. Часова шкала впорядковує вміст анімаційного зображення за часом, керує шарами і кадрами. На ній відображається інформація про шари, ключові кадри й ті, що генерує програма, точка відтворення. За допомогою часової шкали переміщають ключові кадри і цілі частини анімації.

У редакторі GIMP можна створити покрокову анімацію: кожний кадр є окремим шаром із зображенням кожної фази руху. Для перегляду кожного шару певний час застосовують фільтр Анімація і команду Відтворити (фільтр містить ще команди Оптимізувати та Розоптимізувати, про це далі). Відкриється вікно показу фільму.

Робоче поле вікна містить зображення шару — тла, а перегляд анімації здійснюється такими кнопками (рис. 1):

- Відтворити — автоматично відбувається почерговий перегляд кожного шару.
- Крок і Назад — дозволяють перегляд кожного кадру (наступного чи попереднього) після використання кнопки.

- Від'єднати — відтворюють зображення шару без шару — тла.

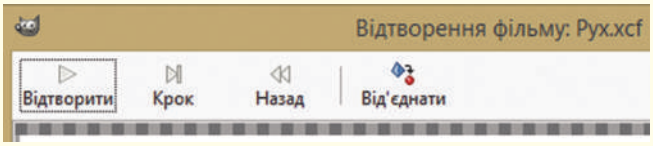


Рис. 1. Кнопкове меню фільтра відтворення анімації

У GIMP є можливість одразу зберегти файл — експортувати його у формат gif. Для цього слід скористатися командою ФАЙЛ — Зберегти як…; у вікні збереження відкрити список Вибрати тип файлу (За розширенням). У списку прописано формати файлів, у які експортується це зображення. Далі вибрати формат gif і підтвердити — відкриється вікно налаштування параметрів файлу перед експортом (рис. 2).

Оскільки метою є збереження як анімація, то й вибирають відповідну команду. Після підтвердження кнопкою Експорт відкриється вікно вибору параметрів анімаційного gif-файла (рис. 3).

У рядку стану фіксується масштаб і номер кадру перегляду. Якщо закрити вікно, то в діалозі шарів у назві кожного шару в дужках додався параметр — час відображення шару в показі анімації.

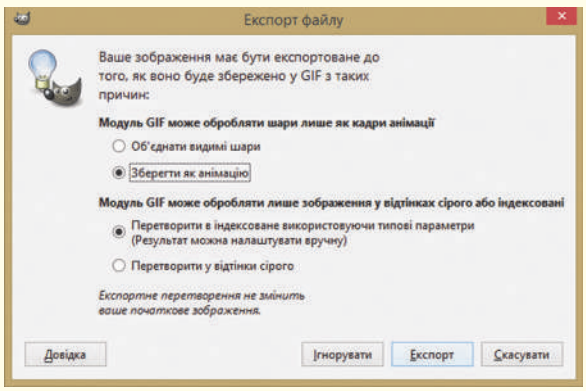


Рис. 5.46. Вікно експорту файла

Команда **Розміщення кадру** має три режими: **Неважливо**, **Накладення шарів** (об'єднання), **Один кадр на шар** (заміна).

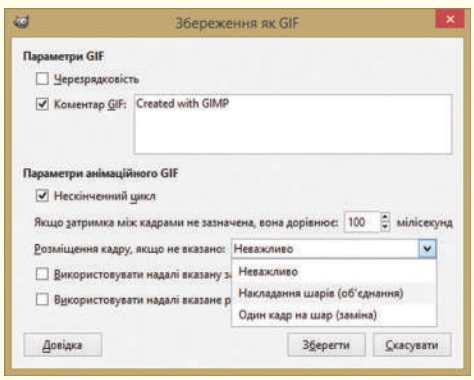


Рис. 5.47. Вікно налаштування параметрів gif-файла

У вікні в області Параметрів анімації gif є можливість автоматичного повтору анімації після її завершення (команда Нескінченний цикл); зміни часу показа кадру (шару) у мікросекундах.

Команда Розміщення кадру має три режими: Неважливо — редактор за замовчуванням запускає такий режим; **Накладення шарів** (об'єднання) — послідовно накладає один шар на інший — в анімації з прозорими шарами буде видно зображення нижніх шарів; Один кадр на шар (заміна) — заміщує попередній шар на новий. Після налаштування параметрів файл зберігають — кнопка Зберегти.

В анімаційному gif-файлі зберігаються зображення кожного шару. Для зменшення розміру файла можна зменшити кожний шар до розміру зображення і прибрати деякі частини зображення. Автоматично оптимізувати зображення можна командою Оптимізація фільтра Анімація (рис. 4).

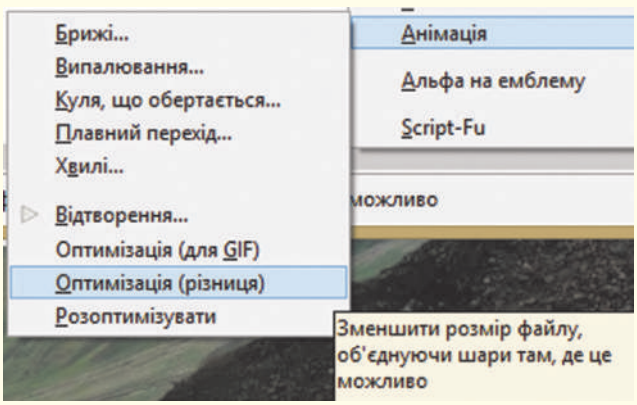


Рис. 4. Команди фільтру Анімація

Команда працює за таким алгоритмом: здійснюється перегляд кожного шару, якщо шар містить точки, які відрізняються від відповідних точок попереднього шару, то вони залишаються на шарі, а розмір шару змінюється на мінімально можливий. Решта точок перетворюються на прозорі. У вікні шарів до назви шару додається назва режиму — combine (новий кадр додається до попередніх) або replace (новий кадр замінює всі попередні).

Команди Оптимізація (для GIF) і Оптимізація (різниця) порізному змінюють розміри шару: перша урізає шар до зображення на ньому, а друга ще й об'єднує шари там, де це доцільно. Команду Розоптимізувати використовують для зняття оптимізації з метою подальшого коригування зображення.

Існує вид анімації, який називається **інтерактивним** (оснований на взаємодії). Інтерактивною є презентація, в якій на слайді відбуваються зміни як реакція застосування тригера. З інтерактивними зображеннями кнопок ми стикаємося на веб-сторінках сайта. Грецький художник і програміст Петрос Врелліс створив інтерактивну анімацію картини Вінсента Ван Гога «Зоряна ніч». Використовуючи сенсорний інтерфейс, глядач може деформувати зображення, змінюючи напрямок руху частинок. Якщо не торкатися до картини, то відтворюється початкове зображення картини.

Для *створення простої анімації* необхідно дотримуватися таких дій.

Крок 1	Розробіть ідею. Визначте об'єкт і властивості, які будуть змінюватись стосовно тематики анімації.
Крок 2	Розробіть схему анімаційного сюжету, яка визначатиме послідовність змін.
Крок 3	Створіть сценарій. Що потрібно намалювати? Що потрібно повторити з різними властивостями об'єкту? Що залишиться статичним?
Крок 4	Створіть кадри анімації — комп'ютерні малюнки в будь-якому графічному редакторі з урахуванням техніки малювання і сценарію.

Крок 5	Налаштуйте параметри анімації.
Крок 6	Виконайте експорт зображення у формат GIF.

Розглянемо приклад застосування редактора GIMP для створення анімаційного зображення.

Приклад. Створити анімаційне зображення.

1. Ідея: рух об'єкта — літери та її «падіння» з площини.
2. Схема руху: на площині з'являється об'єкт; рух об'єкта по горизонталі; на краю зображення площини об'єкт обертається; вертикальний рух об'єкта з обертанням.
3. Сценарій: зображення площини є статичним; літера Р (від слова рух) промальовується на різних шарах, остання — на краю площини, до неї застосовується перетворення — обертання на 45°; нижче по вертикалі повторюється зображення літери і знову застосовується обертання на 45°; наступне повторення останньої літери знизу — «літера впала».
4. У наведеному сценарії літера не буде рухатись, кожна наступна з'являтиметься в новому місці, а попередня незникатиме. Щоб створювалося враження руху однієї літери, на кожному шарі повторюють зображення літери попереднього шару із заповненням кольором фону.
5. У подальшому для перегляду створеного користуються фільтром **Відтворення анімації**. Доцільно оптимізувати анімаційне зображення відповідним фільтром.
6. Анімацію експортують в gif-файл, за потреби налагоджують час затримки шарів.

Запитання для перевірки знань

1. Що таке кадр?
2. Назвіть типові значення частоти зміни кадрів.
3. Що таке ключові кадри?
4. Який вид анімації створюють у редакторі GIMP?
5. Як зберегти анімаційне зображення у вигляді файла — фільму?
6. Які формати збереження зображень містить GIMP?

5.20. Макетування та верстка графічного документа. Макетування для Веб

Що таке верстка, макет? Які програми для верстки документа вам знайомі?

Пригадаємо, що дизайн будь-якого друкованого видання починається зі створення **макету** — графічного планування верстки. У процесі створення макету в ескізно-узагальненому вигляді планується розташування текстового та ілюстративного матеріалу, всі деталі їх оформлення.

Важливими етапами створення макету є підготовка тексту та ілюстрацій. Розглянемо приклад.

Версткою є процес розміщення текстових блоків та підготовленого набору всіх видів ілюстрацій з урахуванням дизайну макету.

Приклад

• Текстові та графічні блоки планують відповідно до тематики публікації, сайта; в публікаціях ураховують тип видання (шпальта газети та сторінка книги матимуть різні макети).

• Геометричний розмір графічного зображення потрібно вибирати відповідно до розміру тематичного блоку.

• Для веб-сторінок обов'язково присутні вимоги до розміру зображення, що впливає на формат збереження зображення.

• Для різних видань і сайтів можуть бути різні вимоги до колірної гами зображень.

Є вимоги також до текстових даних, у тому числі до тексту як компонента зображень.

Більшість програм верстки підтримують векторний формат EPS (Encapsulated PostScript). Для векторної графіки фірма Microsoft створила формат метафайла Windows (WMF). У ньому зручно переносити прості малюнки (наприклад, діаграми), які створено в середовищі електронних таблиць. У поліграфії найбільш популярним растровим форматом є TIFF.

Для завантаження на сайт фахівці рекомендують такі формати графічних зображень, як JPEG, GIF, PNG. Для збереження якості фотографій та малюнків із градієнтним заповненням краще використовувати формат JPEG. Формати GIF і PNG підтримують прозорість: тло малюнка є прозорим.

У процесі створення нового графічного зображення в растровому редакторі зазначають кількість пікселів на дюйм, зазвичай 72 або 96. Зображення квадратної форми 1200 x 1200 пікселей найкраще підходять для новин у Facebook і LinkedIn, альбомні 1200 x 628 пікселів упишуться в пости Facebook або Twitter, а портретні зображення 736 x 1128 пікселів використовують у Google+.

Для ілюстрування публікацій, сторінок сайтів використовують різні можливості отримання графічних зображень: готові зображення отримують із файлів, скануванням, із цифрових камер, для створення потрібних зображень використовують векторні та растрові редактори.

Вимоги до графічних зображень для макетів друкованих матеріалів та макетів для сайтів можуть бути різними. У загальному вигляді вимоги є такими.

Для підготовки розміщення графічного зображення на сторінці сайта велику роль відіграє розмір файла із зображенням. Великий за розміром файл завантажується довго.

На розмір файла впливає розмір зображення, формат його збереження і ступінь стиснення, обсяг кольорових відтінків.

У векторному редакторі Inkscape для збереження зображення у форматі PNG використовують команду ФАЙЛ — Експортувати та у вікні, що відкриється, налагоджують параматри збереження.

У GIMP для експорту зображення у файл з іншим розширенням, не GIMP, потрібно вибрати команду Вибрати тип файлу (За розширенням) у вікні збереження файла, у списку форматів файлів — формат JPEG. Після підтвердження відкриється вікно налаштування параметрів файла перед експортом. Оскільки цей формат підтримує стиснення, то у вікні зазначають параметр якості зображення (без стиснення 100). Виконайте вправу для знайомства з вказаною командою.

Отже, розмір файла з графічним зображенням залежить від розміру малюнку (растрового) та ступеня стиснення. Щоб підібрати правильно розміри малюнків для сайта, необхідно знати розміри та роздільну здатність екрану. Але зазвичай розміри сторінок сайта є змінними, вони програмуються з додатковим налаштуванням до параметрів пристрою, на якому переглядають сайт.

Малюнки, що мають розмір, менший за рекомендований, збільшуються, а отже, розтягуються і втрачають якість. Малюнки більшого розміру можуть стискатися програмним способом, і їх вигляд теж може погіршитись порівняно з оригіналом. Картинки, в яких порушена запропонована

пропорційність у розмірах, обрізаються, що призводе до неправильного їх вигляду.

Багато відтінків графічного зображення, яке чудово виглядає на екрані дисплею та створене з використанням RGB моделі, не передається на папір під час друку.

Кольороподілом називають розкладання кольорового зображення з режиму RGB на чотири складові фарби CMYK, які потім з'єднуються при друку, утворюючи багатобарвне зображення. Кольороподілом також називають конвертацію RGB — CMYK.

У графічних редакторах завжди можна вибрати колірну модель для відтворення відтінків кольору зображення, але у будь-якому випадку, колір на екрані не передається у повному обсязі на друк.

Вимоги до зображень для розміщення на сайті є важливими в процесі перетворення макету сайта в формат HTML (мова гіпертекстової розмітки документа).

Версткою веб-сторінки (англ. *page-proof*) є процес створення веб-сторінки із попередньо створеного макету дизайну сайта, заздалегідь створеного за допомогою графічних редакторів.

Макет веб-сторінки є результатом творчості веб-дизайнера. Наразі розглянемо макетування сторінки, що має найбільш поширений вигляд, та розглянемо призначення кожної її частини (рис. 1).

Шапка (Header) **сайта** — блок у верхній частині сторінки сайта зазвичай містить логотип, меню, контакти, перемикач мов або кошик для сайтів інтернет-магазину.

Блок у нижній частині сторінки називається **підвалом** (Footer). Підвал містить корисну, але не першорядну інформацію, наприклад, назву організації, яка розробляла сайт, або розробника сайта, контакти, також можуть дублюватись пункти меню.

Зазвичай веб-програмісти так створюють сайт, що Header і Footer видно на всіх сторінках сайта.

Блок з основним вмістом сторінки може бути поділений на кілька колонок: у лівій колонці зазвичай розміщено меню сайта, перелік корисних посилань тощо; наступна колонка містить інформацію вибраного пункту меню: може бути й третя (права) колонка з об'явами, посиланням на важливі події у графічному вигляді.

Розподіл на колонки має свої особливості: є отступи зліва і справа сторінки (Margin), між колонками (Gutter). Значення Margin може дорівнювати 0 для перегляду сайтів за комп'ютером. Та сьогодення характеризується поширенням мобільних пристроїв, ширина сайта на яких дорівнюватиме ширині екрану і вигляд прилипання тексту до країв буде непривабливим. Виконайте вправу, яка допоможе вам створити макет веб-сторінки (цю вправу можна запропонувати як домашнє завдання).

Скільки команд містить графічний редактор GIMP для створення нового файла?

Зазвичай для обміну зображеннями в соціальних мережах фахівці пропонуютьтакі такі розміри графічних зображень, як Facebook — 1200 x 628; Twitter — 1024 x 512; LinkedIn — 800 x 800; Google+ — 800 x 1200; Instagram — 600 x 600.

Під час друку відтворюються не всі відтінки. Для більш точної передачі якого-небудь відтінку застосовуються так звані «прості» (Spot) кольори, які отримують попередньо змішуючи фарби у змішувачі. Існує кілька систем простих кольорів. Найбільш поширеною з них є система Pantone, в якій кожна фарба має свій цифровий код. Випускаються каталоги простих кольорів, які допомагають користувачеві дібрати потрібний відтінок, а потім, скориставшись кодом, замовити потрібну фарбу. Так, зокрема, друкується золотий чи срібний колір.

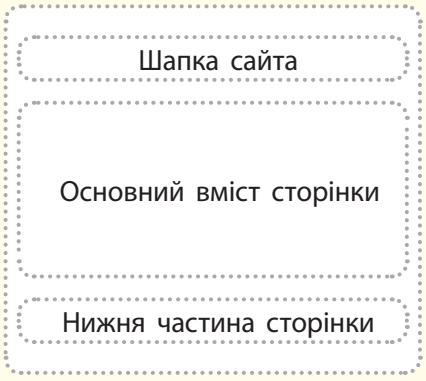


Рис. 1. Загальний макет веб-сторінки

Растровий редактор GIMP містить підкоманду Створити (меню ФАЙЛ) із набором можливостей створення графічних елементів для розміщення на сайті. Завдяки їй на шарах зображення одразу створюють кнопки різного вигляду (прямокутні із заокругленими краями, об'ємні); емблеми — текст, який вводить користувач із застосуванням різного типу фільтрів до нього. У команді Створити також вибирають Текстуру, а команда Теми веб-сторінок дозволяє оформити елементи веб-сторінки в одному стилі.

Запитання для перевірки знань

1. Назвіть вимоги до графічного зображення видання або сайта.
2. У який формат краще експортувати графічне зображення, підготовлене для веб-сайта?
3. Що називається кольороподілом або конвертацією кольорів?
4. Яка кольорова модель використовується у оформленні сайта, а яка у видавництві? Поясніть чому.
5. Що означає верстка веб-сторінки?
6. На що звертають увагу, вибираючи коефіцієнт стиснення при збереженні растрового малюнку в JPEG-файлі?

Завдання для самостійного виконання 1

1. Відкрийте GIMP, завантажте готове растрове графічне зображення.
2. Надалі збережемо його з різною якістю JPEG-файла: запустіть команду **ФАЙЛ — Зберегти як**; виберіть папку для збереження, введіть назву файла; відкрийте список форматів: **Вибрати тип файла** (**За розширенням**); виберіть тип JPEG.
3. У вікні експорту зазначте якість 100, збережіть ще раз з іншим іменем та якістю 40, як у пунктах 3, 4.
4. Відкрийте папку, встановіть детальний перегляд її вмісту та порівняйте розміри файлів, збережених вами. Передивіться зображення і зробіть висновок щодо якості і розміру файлів.

Завдання для самостійного виконання 2

1. Перейдіть до сайта МАН України (Малої академії наук) (http://man.gov.ua/ua).
2. Проаналізуйте вміст сайта, визначте наявність складових сайта, їх призначення.
3. Дайте відповідь на такі запитання (запишіть у зошит).
   - Із чого складається Header і Footer сайта? Чи змінюють вони свій вигляд на різних сторінках сайта?
   - Чи містить сайт логотип?
   - Що означає наявність прапорців у верхньому лівому куту хедера?
   - Із чого складається основна частина сайта?
4. Перейдіть на сайт Міністерства освіти і науки України (https://mon.gov.ua/ua).
5. Порівняйте вигляд сайтів і створіть детальний макет головної сторінки вашого сайта, наприклад, про спортивні досягнення класу.
6. Доберіть і збережіть зображення, які б ви хотіли розмістити на сайті (розміри підберіть кадруванням, налаштуйте якість та стиснення зображень, збережіть їх у відповідному для сайта форматі).

Виконайте тестове завдання до розділу 5 з автоматичною перевіркою результату на сайті interactive.ranok.com.ua

Комп'ютерний словник

База даних — сховище організованої сукупності даних різного типу, які відображають стан об'єктів певної предметної галузі та зв'язки між ними.

Вибірка (вибіркова сукупність) — частина генеральної (загальної) сукупності об'єктів, яка охоплюється експериментом.

Виняток — подія, яка може виникнути під час виконання програми і яка може змінити подальший хід її виконання.

Віджети — об'єки, із яких створюється графічний інтерфейс.

Глобальна змінна — змінна, оголошена в основній програмі.

Діапазон — незмінна послідовність цілих чисел, що визначається початковим, кінцевим значеннями і кроком їх заміни.

Екземпляр класу — окремий об'єкт класу.

Електронне урядування — форма організації державного управління, яка сприяє підвищенню ефективності, відкритості та прозорості діяльності органів державної влади та органів місцевого самоврядування з використанням інформаційно-телекомунікаційних технологій для формування нового типу держави, орієнтованої на задоволення потреб громадян.

Змінна екземпляру класу — змінна, що визначена у середині методу класу і яка належить тільки цьому класу.

Змінна класу — змінна, що визначається у середині класу і яка є доступною для всіх екземплярів даного класу.

Інтерактивний режим — режим роботи з програмою, за яким результат виконання інструкції виводиться одразу після її введення.

Інфографіка (від латин. Informatio — інформування, роз'яснення, і грец. γραφικός — письмовий) — це візуальне відображення інформації, статистичних даних для простої і наочної демонстрації тенденцій, співвідношень, а також зацікавлення в предметі дослідження.

Інформаційна безпека — стан захищеності систем опрацювання й зберігання даних, за якого забезпечено конфіденційність, цілісність і доступність даних.

Інформаційна модель — сукупність інформації, яка описує суттєві для даного розгляду властивості об'єкта, зв'язок між ними та яка досліджує можливий стан об'єкта при зміні його властивостей.

Інформаційна революція (англ. Information Revolution) — поняття, що відображає революційний вплив інформаційних технологій на всі сфери життя суспільства в останній чверті XX ст.

Інформаційна система — сукупність взаємозв'язаних елементів, призначених для виконання інформаційних процесів та орієнтованих на рішення задач певної галузі людської діяльності.

Інформаційна технологія — сукупність процесів, що використовує засоби та методи пошуку, збирання, накопичення, зберігання, опрацювання і передавання первинної інформації для отримання інформації нової якості про стан об'єкту, процесу або явища за допомогою засобів обчислювальної та комунікаційної техніки.

Клас — об'єкт, що складається із сукупності методів і змінних, які описують даний об'єкт.

Колонтитул (від франц. colonne — стовпець і латин. titulus — напис, заголовок) — написи, які розміщують над текстом (верхній колонтитул) або під текстом (нижній колонтитул) кожної сторінки документу (книги, газети, журналу тощо).

Команда — вказівка комп'ютеру, що визначає: яку і як слід виконати дію (операцію).

Комп'ютерна програма — сукупність команд (інструкцій), що забезпечує реалізацію на комп'ютері конкретного алгоритму.

Комп'ютерне моделювання — процес створення інформаційних моделей комп'ютерними засобами.

Кореляційний аналіз, або кореляція (від латин. Correlatio — співвідношення, взаємозв'язок) — статистичний взаємозв'язок двох або більше показників; використовується для визначення, чи є між показниками в одній або двох вибірках зв'язок, і для аналізу, чи можна за величиною одного показника передбачити можливе значення іншого.

Кортеж — сукупність об'єктів у круглих дужках.

Лексема — мінімальна одиниця мови програмування, що має самостійний смисл і яку розуміє комп'ютер.

Масив — структурований тип даних, усі елементи якого мають один тип.

Математична статистика — наука про математичні методи систематизації, опрацювання та використання статистичних даних для наукових і практичних висновків.

Метод — програма, що реалізує відповідну функцію.

Множина — неупорядкована колекція унікальних (тих, що не повторюються) об'єктів.

Мова програмування — штучна мова, що являє собоюсистему позначень іправил для запису алгоритму у формі, придатній для подальшого виконання на комп'ютері.

Модель (від латин. Modulus — зразок) — аналог (образ) будь-якого об'єкта, процесу або явища, який використовується як замінник оригіналу.

Наслідування — здатність об'єктів класу застосовувати атрибути цього самого класу, а також здатність одними класами застосовувати атрибути інших класів.

Поліморфізм — явище, за яким синтаксична сутність операції залежить від типу об'єктів, що опрацьовуються.

Розділ — частина документа з з одним і тим самим значенням параметрів форматування.

Рядок — структурований тип даних, елементами якого є букви і синтаксичні символи.

Середовище (система) програмування — комплекс програм, призначений для автоматизації процесу підготовки і виконання на комп'ютері програми користувача.

Словник — набір об'єктів будь-якого типу, доступ до яких здійснюється за допомогою ключа.

Список — сукупність об'єктів будь-якого типу у квадратних дужках, які відокремлюються один від одного комою.

Структура документа — схема розміщення складових документа. При перегляді документу в режимі структури його вміст показано у вигляді маркірованого списку.

Циклічний алгоритм — алгоритм, у якому певна сукупність інструкцій може виконуватися багаторазово.

Штучний інтелект (англ. Artificialintelligence) — наука (розділ математичної лінгвістики та комп'ютерних наук) та набір технологій, які дозволяють комп'ютеру виконувати різні функції, притаманні людині.

Алфавітний покажчик

А

Алгоритм 30
Атрибут 79

В

Виняток 105
Вказівник 61

Г

Графічний об'єкт 125
Графічний формат 207

Д

Діапазон 52

Е

Екземпляр класу 71
Електронне врядування 145

З

Злиття 193
Змінна класу 64

І

Інфографіка 171
Інформаційна безпека 150
Інформаційна модель 154
Інформаційна революція 137
Інформаційна система 136
Інформаційна технологія 135

К

Канал 235
Клас 77
Колірна модель 205
Команда 4
Комп'ютерна графіка 202
Комп'ютерна програма 4
Комп'ютерне моделювання 154
Кортеж 52

Л

Лексема 19

М

Масив 157, 161
Математична статистика 164
Метод 28, 46
Множина 52
Мова програмування 4
Модель 153
Модуль 74

Н

Наслідування 87

О

Оператор 24

П

Подія 121
Поліморфізм 64, 91
Публікація 197

Р

Рекурсія 71
Розгалуження 30
Розділ 186

С

Середовище програмування 8
Синтаксис 7
Словник 54
Список 44, 180
Стек 48
Структура документа 191

Ф

Фільтр 142
Функція 62

Ц

Цикл 37

Ш

Шар 232
Штучний інтелект 148

Зміст

Розділ 1. МОВА ПРОГРАМУВАННЯ ТА СТРУКТУРИ ДАНИХ

1. Структура і способи виконання проектів мовою Python
1.1. Класифікація і складові мов програмування ... 4
1.2. Призначення і склад середовища програмування ... 10
1.3. Основні можливості мови Python і структура проекту ... 11
1.4. Режими виконання програмного коду в середовищі IDLE ... 13

2. Оператори, вирази і засоби опрацювання чисел
2.1. Основні елементи мови Python ... 19
2.2. Поняття про перетворення типів даних ... 22
2.3. Оператори і вирази ... 24
2.4. Модулі, функції і методи для опрацювання числових даних ... 28

3. Реалізація базових алгоритмічних конструкцій
3.1. Реалізація алгоритмів з розгалуженням ... 30
3.2. Вкладені оператори умовного переходу ... 33
3.3. Реалізація циклічних алгоритмів ... 37

4. Вбудовані типи даних та їх опрацювання
4.1. Списки, стеки, черги ... 44
4.2. Кортежі, діапазони, множини ... 52
4.3. Словники. Функції, операції і методи опрацювання словників ... 54
4.4. Масиви ... 57
4.5. Вказівники ... 61

5. Функції користувача та модулі мови Python
5.1. Функції ... 62
5.2. Рекурсивні функції ... 70
5.3. Модулі ... 74

6. Класи, об'єкти, наслідування
6.1. Елементи теорії об'єктно-орієнтованого програмування (ООП) ... 77
6.2. Створення класів і об'єктів ... 79
6.3. Конструктор класу ... 83
6.4. Наслідування ... 87

7. Поліформізм, перевизначення методів, модулі користувача
7.1. Поліморфізм ... 91
7.2. Перевизначення та розширення можливостей методів ... 95
7.3. Композиційний підхід в ООП мовою Python ... 100
7.4. Створення та використання модулів користувача ... 102
7.5. Опрацювання виняткових ситуацій ... 105

8. Основи графічного інтерфейсу користувача
8.1. Загальний порядок створення графічного інтерфейсу ... 110
8.2. Графічні об'єкти і їх властивості ... 114
8.3. Опрацювання подій ... 121
8.4. Меню ... 124
8.5. Діалогові вікна ... 127
8.6. Графічні примітиви об'єкта Canvas ... 131

Розділ 2. СУЧАСНІ ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ

2.1. Сучасні інформаційні технології та системи. Людина в інформаційному суспільстві ... 135
2.2. Навчання в Інтернеті ... 138
2.3. Професії майбутнього — аналіз тенденцій на ринку праці ... 140
2.4. Системи електронного врядування ... 144
2.5. Поняття про штучний інтелект ... 148

Розділ 3. АНАЛІЗ І ВІЗУАЛІЗАЦІЯ ДАНИХ

3.1. Комп'ютерне моделювання об'єктів і процесів. Електронні таблиці ... 153
3.2. Розв'язування рівнянь, систем рівнянь, оптимізаційних задач із різних предметних галузей засобами інформаційних технологій ... 155
3.3. Матричні операції. Розв'язання систем лінійних рівнянь ... 160
3.4. Основи статистичного аналізу даних. Ряди даних. Кореляційний аналіз даних ... 164
3.5. Обчислення основних статистичних характеристик вибірки засобами електронного процесора ... 167
3.6. Візуалізація рядів і трендів даних. Інфографіка ... 172
3.7. Розв'язування задач із різних предметних галузей. Табличний процесор як засіб для фінансових розрахунків ... 177
3.8. Електронна таблиця як засіб подання відомостей про однотипні об'єкти. Операції з однотабличною базою даних ... 180

Розділ 4. ЕЛЕКТРОННІ ПУБЛІКАЦІЇ

4.1. Багатосторінкові текстові документи. Настроювання параметрів сторінок, розділи ... 184
4.2. Колонтитули ... 187
4.3. Схема документа ... 191
4.4. Використання полів злиття ... 194
4.5. Комп'ютерні публікації. Видавничі системи. Електронні книги ... 198

Розділ 5. ГРАФІКА. МУЛЬТИМЕДІА

5.1. Комп'ютерна графіка та сучасні напрями її використання. Види комп'ютерної графіки ... 203
5.2. Моделі відображення кольору ... 205
5.3. Формати графічних файлів ... 207
5.4. Створення векторних ілюстрацій в офісних програмних засобах ... 209
5.5. Векторний графічний редактор Inkscape. Інтерфейс редактора ... 212
5.6. Інструменти векторного редактора Inkscape та їх налаштування ... 214
5.7. Колір в Inkscape. Фарбування градієнтом ... 217
5.8. Складні векторні об'єкти в Inkscape ... 219
5.9. Опрацювання тексту в Inkscape ... 222
5.10. Художні ефекти в Inkscape ... 223
5.11. Растровий графічний редактор GIMP ... 226
5.12. Інструменти малювання в GIMP та їх налаштування ... 228
5.13. Інструменти виділення в GIMP та їх налаштування ... 230
5.14. Шари. Створення колажу ... 232
5.15. Редагування зображень у GIMP ... 234
5.16. Канали. Корекція кольору та тону ... 237
5.17. Коригування зображень. Інструменти ретушування в GIMP ... 240
5.18. Фільтри. Інструмент Текст ... 242
5.19. Комп'ютерна анімація ... 244
5.20. Макетування та верстка графічного зображення. Макетування веб-сторінки ... 247
Комп'ютерний словник ... 251
Алфавітний покажчик ... 253