Трофименко О.Г.
Прокоп Ю.В.
Задерейко О.В.

Алгоритмізація та програмування

Навчально-методичний посібник

О. Г. Трофименко, Ю. В. Прокоп, О. В. Задерейко

Алгоритмізація та програмування

Навчально-методичний посібник
для підготовки здобувачів вищої освіти
галузі знань 12 «Інформаційні технологій»

Одеса
Фенікс
2020

УДК 004.43:004:421(076)
Т 76

Рекомендовано Вченою Радою
Національного університету «Одеська юридична академія»,
Протокол № 3 від 23 грудня 2019 р.

Р е ц е н з е н т и:

Казаков А. І. – доктор технічних наук, професор, завідувач кафедри «Інформаційні технології проектування в електроніці та телекомунікаціях» Одеського національного політехнічного університету;

Щербакова Г. Ю. – доктор технічних наук, професор, професор кафедри «Інформаційні системи» Одеського національного політехнічного університету.

Т 76 Трофименко О. Г., Прокоп Ю. В., Задерейко О. В. Алгоритмізація та програмування : навчально-методичний посібник. Одеса : Фенікс, 2020. 310 с. URL : http://dspace.onua.edu.ua/handle/11300/12345.

ISBN 978-966-928-484-6

Посібник призначений для підготовки до лабораторних і самостійних робіт, які виконуються в межах навчальної дисципліни, присвяченої вивченню основ алгоритмізації та програмування. Розглянуто основні засоби програмування мовою С++, як базових алгоритмів, так і опрацювання структурованих типів, робота з вказівниками, засоби динамічного керування пам’яттю тощо. Запропоновані роботи мають по 30 варіантів індивідуальних завдань різного рівня складності та супроводжуються прикладами програм. Подано завдання до шістнадцяти лабораторних і шістнадцяти самостійних робіт, містить теоретичні відомості з детальним описом матеріалу й контрольні запитання, відповіді на які сприятимуть систематизації та закріпленню набутих знань з основ алгоритмізації та програмування у майбутніх ІТ-спеціалістів.

Для підготовки здобувачів вищої освіти галузі знань 12 «Інформаційні технології».

УДК 004.43:004:421(076)

ISBN 978-966-928-484-6

Передмова

У навчально-методичному посібнику подано короткий теоретичний матеріал та численні наочні приклади створення програмних проєктів засобами мови С++ для організації обчислень лінійної, розгалуженої та циклічної структур, опрацювання одно- та двовимірних масивів, програмного використання вказівників та динамічної пам’яті, рядків, текстових та бінарних файлів тощо. Запропоновано до виконання шістнадцять лабораторних та шістнадцять самостійних робіт, кожна з яких має по тридцять індивідуальних варіантів завдань різних рівнів складності. Студент сам або за вказівкою викладача вибирає завдання того чи іншого рівня складності відповідно до варіанта. У подальшому при оцінюванні знань викладач може враховувати рівень складності виконання роботи та оптимальність алгоритму програми виконаної роботи.

Перед виконанням лабораторного завдання студентові потрібно:

- уточнити у викладача індивідуальне завдання;
- вивчити відповідні розділи теоретичного курсу згідно з лекційними записами та навчальною літературою;
- розробити схеми алгоритму розв’язання задач;
- написати тексти програм мовою С++;
- підготувати протокол виконання лабораторної роботи і подати його викладачеві для перевірки.

До виконання лабораторної роботи допускається студент, який має попередньо підготовлений самостійно заповнений протокол лабораторної роботи.

Зміст протоколу лабораторної роботи:

- назва теми і мета лабораторної роботи;
- відповіді на контрольні запитання;
- схеми алгоритмів для розв’язання індивідуального завдання;
- тексти програм мовою С++;
- результати обчислень на комп’ютері.

Правильність роботи програми та здобутих результатів перевіряються і оцінюються викладачем.

Структура дисципліни

Навчальні дисципліни, присвячені вивченню основ алгоритмізації та програмування, вивчаються здобувачами вищої освіти, що навчаються за освітньо-кваліфікаційним рівнем «бакалавр» галузі знань 12 «Інформаційні технології» на першому курсі навчання. Вони разом з іншими складають теоретично-практичну основу сукупності компетентностей, що формують профіль фахівця в галузі інформаційних технологій.

Актуальність таких дисципліни обумовлена об'єктивною необхідністю успішного засвоєння майбутніми ІТ-фахівцями принципів опрацювання цифрової інформації, формування навиків алгоритмізації та програмування алгоритмічною мовою програмування високого рівня, вміння використовувати основні знання основ програмування при розробці інформаційних технологій, що зараз охоплюють майже всі сфери життєдіяльності.

Метою вивчення навчальних дисциплін з основ програмування є формування у здобувачів вищої освіти теоретичних знань з алгоритмізації і програмування алгоритмічною мовою високого рівня С++, навиків створення програмних проєктів, застосування здобутих навиків у процесі навчання і майбутній професійній діяльності при вирішенні різноманітних завдань у практичній діяльності за фахом.

Предметом вивчення є методи розроблення алгоритмів і конструювання програмного забезпечення.

Основними завданнями дисципліни є формування у майбутніх фахівців навиків написання та налагодження програм мовою високого рівня С++, знань основ конструювання програмного забезпечення, оволодіння методологією компонентної розроблення програмного забезпечення.

Підсумкові результати навчання навчальної дисципліни деталізують такі програмні **результати навчання**:

– аналізувати, цілеспрямовано шукати і вибирати необхідні для вирішення професійних завдань інформаційно-довідникові ресурси і знання з урахуванням сучасних досягнень науки і техніки;

– вміти розробляти людино-машинний інтерфейс;

– знати і застосовувати методи розроблення алгоритмів, конструювання програмного забезпечення;

– вміти застосовувати компонентну розробку програмного забезпечення.

Перелік тем лабораторних робіт

Лабораторна робота № 1. Знайомство з С++. Способи введення-виведення даних.

Лабораторна робота № 2. Програмування лінійних алгоритмів.

Лабораторна робота № 3. Умовний оператор if.

Лабораторна робота № 4. Оператор вибору варіантів switch.

Лабораторна робота № 5. Програмування циклів. Оператор циклу з параметром for.
Лабораторна робота № 6. Вкладені цикли.
Лабораторна робота № 7. Оператори циклу while та do-while.
Лабораторна робота № 8. Організація функцій у C++.
Лабораторна робота № 9. Одновимірні масиви.
Лабораторна робота № 10. Двовимірні масиви.
Лабораторна робота № 11. Вказівники і динамічна пам'ять при опрацюванні одновимірних масивів.
Лабораторна робота № 12. Вказівники і динамічна пам'ять при опрацюванні двовимірних масивів.
Лабораторна робота № 13. Робота з символьними даними.
Лабораторна робота № 14. Робота з текстовими файлами.
Лабораторна робота № 15. Програмування з використанням структур (struct).
Лабораторна робота № 16. Робота з бінарними файлами.

Кожна із запропонованих до виконання робіт має теоретичні відомості з докладним описом порядку виконання та контрольні запитання для закріплення відповідного тематичного матеріалу.

Перелік тем самостійної роботи

Самостійна робота № 1. Запис математичних виразів мовою C++.
Самостійна робота № 2. Програмування логічних операцій.
Самостійна робота № 3. Побітові логічні операції.
Самостійна робота № 4. Програмування розгалужень.
Самостійна робота № 5. Циклічне опрацювання послідовностей чисел.
Самостійна робота № 6. Циклічне опрацювання чисел.
Самостійна робота № 7. Застосування циклів.
Самостійна робота № 8. Функції.
Самостійна робота № 9. Опрацювання одновимірних масивів у функціях.
Самостійна робота № 10. Опрацювання двовимірних масивів у функціях.
Самостійна робота № 11. Створення бібліотеки функцій.
Самостійна робота № 12. Рядки char*.
Самостійна робота № 13. Опрацювання текстових файлів з числовими даними.
Самостійна робота № 14. Опрацювання даних типу "дата-час" у текстових файлах.
Самостійна робота № 15. Робота з бінарними файлами.
Самостійна робота № 16. Відбір даних бінарного файлу за умовою з формуванням текстового документа.

Правильність виконаних лабораторних та самостійних робіт перевіряє викладач.

ТЕМА 1
ЕЛЕМЕНТИ МОВИ ПРОГРАМУВАННЯ С++

Лабораторна робота № 1
Знайомство з С++.
Способи введення-виведення даних

Мета роботи: набути навиків створювання та налагоджування програм мовою С++.

Теоретичні відомості

1. Введення-виведення даних у стилі С++

Для введення-виведення даних у С++ найчастіше використовуються потокові команди **cin**>> (вхідний потік) та **cout**<< (вихідний потік) бібліотеки iostream.h, наприклад:

```
cout<< "Введіть число: ";   cin>> x;
cout<< "Квадрат цього числа: " << x*x << endl;
```

Перша з команда cout<< виведе рядок символів і залишить курсор на тому ж самому рядку. При цьому символи рядка виводитимуться по одному, доки не зустрінеться завершальний символ '\0'. Друга команда дозволить ввести значення змінної x. Третя команда сформує рядок, поряд виведе числове значення та перемістить курсор на новий рядок (маніпулятор endl).

До речі, крім маніпулятора endl, новий рядок можна сформувати виведенням символу '\n'. Ще одним часто використовуваним спеціальним символом для кращого вигляду виведених даних є символ табуляції '\t', який формує відступ.

Зауважимо, що при виведенні тексту для коректного відображення літер кирилиці слід застосувати команду setlocale(0,".1251") (або setlocale(LC_ALL, "Russian")). Виведення замість літер кирилиці усілякої абракадабри спричинено тим, що, наприклад, Visual Studio в консольних застосунках використовує для набраного тексту кодування Windows 1251, а для введеного тексту – кодування DOS. Коректно відобразити введений за допомогою cin>> текст дозволить команда setlocale(LC_ALL, ".OCP"), повернувши початкові налаштування кодування.

Крім того, встановити потрібну кодову таблицю на потоки введення та виведення можна і такими способами:

```
system("chcp 1251");
```

або

```
system("chcp 1251 > null");
```

або
```
#include <Windows.h>
. . . . . .
SetConsoleCP (1251);
SetConsoleOutputCP(1251);
```
Після цього у властивостях консольного вікна на вкладці *Шрифт* слід вибрати Lucida Console.

Доволі зручною є можливість виведення за допомогою cout<< чисел не лише в десятковому форматі, а і в шістнадцятковому чи вісімковому, використовуючи модифікатори dec, hex і oct усередині вихідного потоку:
```
setlocale(0,".1251");
cout<< "Вісімковий:\t\t "      << oct << 10 << " " << 255 << endl;
cout<< "Шістнадцятковий:\t"<< uppercase<<hex<<10<<"  "<<255<<endl;
cout<< "Десятковий:\t\t "      << dec << 10 << " " << 255 << endl;
```
Результатом виконання цих команд будуть рядки:
```
Вісімковий:         12 377
Шістнадцятковий:    A  FF
Десятковий:         10 255
```
Слід зазначити, що використання одного з цих модифікаторів залишиться в силі, допоки чи то програма не завершиться, чи то не буде використано інший модифікатор.

Для форматування даних при виведенні командою cout<< можна використовувати модифікатор setw, який дозволяє задавати ширину (мінімальну кількість символьних позицій) кожного виведеного числа. При цьому зазначена модифікатором ширина є дійсною лише для одного числа. Для можливості використання модифікаторів слід долучити заголовний файл iomanip. Якщо виведене число має меншу, аніж зазначену у модифікаторі setw ширину, перед ним будуть виведені пробіли. Так, команди
```
#include <iomanip>
. . . . .
cout<< "x=" << setw(1) << 155 << endl;
cout<< "x=" << setw(3) << 155 << endl;
cout<< "x=" << setw(5) << 155 << endl;
```
сформують рядки у такому вигляді:
```
x=155
x=155
x=  155
```
Тобто в останньому рядку перед значенням 155 виведуться два пробіли, а все число займе п’ять позицій.

При виведенні дійсних чисел, особливо коли йдеться про виведення матриць дійсних чисел, досить доречним є обмеження кількості знаків після десяткової крапки за допомогою маніпулятора **setprecision(int count)**:
```
cout << fixed << setprecision(3) << (13.5 / 2) << endl; // 6.750
cout << fixed << setprecision(2) << 24.16425 << endl;   // 24.16
```
Використаний тут маніпулятор fixed задає виведення дійсних чисел з рухомою крапкою у форматі з фіксованою крапкою.

Команда cout<< дозволяє перенаправляти виведення на пристрій чи до файлу за допомогою операторів перепризначення виведення операційної системи. Однак, повідомлення про помилки звичайно недоречно спрямовувати до

файлу і при цьому не виводити на екран. Для уникнення таких ситуацій існує спеціальний вихідний потік `cerr`, пов'язаний зі стандартним пристроєм помилок. Наприклад, команда

```
cerr<< "Повідомлення про помилку" << endl;
```

сформує відповідне повідомлення на екрані, а операційна система не дозволить перенаправити виведення на інший пристрій чи до файлу.

2. Введення-виведення у стилі C

1) Для форматного введення-виведення в С існують функції **`scanf`** (форматоване введення даних) та **`printf`** (форматоване виведення даних), які містяться в бібліотеці `stdio.h`. Обидві функції мають схожий формат:

`scanf`(<*формат*>, < *список_змінних*>);
`printf`(<*формат*>, <*список_змінних*>);

де: *формат* − рядок специфікаторів формату у подвійних лапках. Найбільш поширеними специфікаторами є: `%i` − для цілих чисел, `%f` − для дійсних чисел типу `float` чи `double`, `%s` − для рядка символів;
список_змінних − послідовність розділених комами змінних, значення яких вводиться чи виводяться.

Наприклад, функція `printf("x= %7.3f\n", x)` виведе дійсне число `x` у заданому форматі з трьома знаками дійсної частини числа після десяткової крапки, а функція `scanf("%i %f", &kol, &vart)` введе значення цілої змінної `kol` і дійсної змінної `vart`.

2) Існує ще одна пара функцій С для введення-виведення – **`gets()`** і **`puts()`**, які використовуються лише для рядків. Ці функції мають лише один параметр: текст повідомлення для функції `puts` і змінну-рядок для функції `puts`, наприклад:

```
puts("Hello, Dolly");
char s[13];
gets(s);
```

Функція `puts(s)` виводить рядок `s` на екран, замінюючи нуль-символ на *Enter*. Функція `gets(s)` зчитує символи з клавіатури до появи символу переведення рядка *Enter* і записує їх у рядок `s` (власне символ *Enter* до рядка не долучається, а замість нього записується нуль-символ).

3. Послідовність створення програмних проєктів в онлайн компіляторах

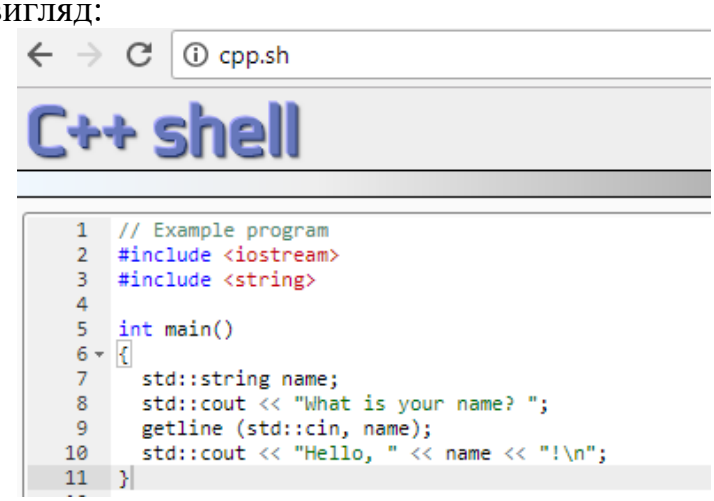
Останнім часом онлайн середовища програмування стають все більш популярними інструментами під час навчання програмуванню, завдяки перевагам:

- онлайн компілятори не потребують встановлення на комп'ютери, через що зникають проблеми несумісності версій програмного забезпечення;
- немає прив’язки до операційної системи, вимог до потужності ПК навчатися і створювати програми можна як на стаціонарному комп’ютері у комп’ютерному класі чи вдома, так і на мобільному телефоні;
- більшість онлайн компіляторів після реєстрації надають можливості автоматичного зберігання програм, завдяки чому не потрібно копіювати їх на флешку або запам'ятовувати файл з проєктом десь у хмарі;

– деякі онлайн середовища програмування пропонують можливості спільної роботи над програмним проєктом, тобто стає можливою взаємодія як декількох студентів при розробленні спільних проєктів, так і більш гнучка роль викладача при перевірці роботи та під час її виконання.

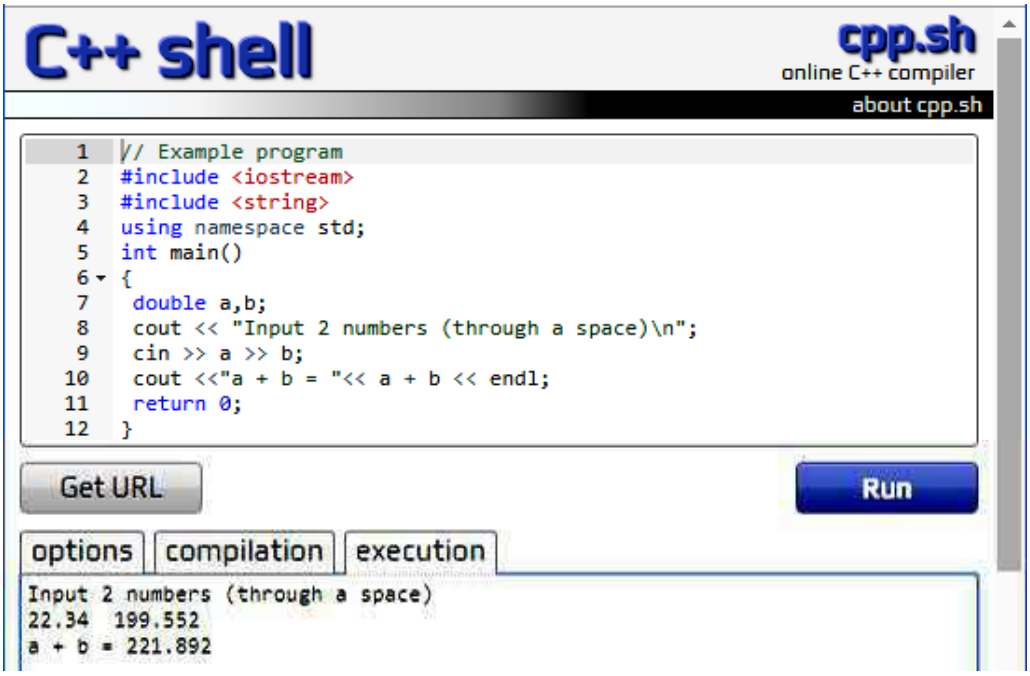
Нині існують численні онлайн компілятори мовою C++, наприклад: http://cpp.sh, https://repl.it, https://ideone.com, http://codepad.org та ін.

Так, при початковому завантаженні **C++ shell** (http://cpp.sh) у браузері його вікно матиме вигляд:



Онлайн компілятор C++ shell має суттєві переваги: простий короткий URL, використання стандартних для C++ команд інтерактивного введення-виведення, проста можливість поділитися кодом, скориставшись кнопкою *Get URL*, вибір стандартів C++ (C++11, C++14), підтримка попереджень тощо. До недоліків можна віднести: роботу тільки з однією мовою програмування C++, можливість транслювати лише однофайлові проекти, відсутність підтримки програмування файлів, періодичні відключення та зависання, відсутність підтримки кирилиці.

Наявний при завантаженні у вікні програмний код слід замінити на власний. Наприклад, програмний код для обчислення суми двох введених чисел разом із результатами (внизу вікна) можуть мати вигляд:



Більш потужним, порівняно з попереднім онлайн компілятором, є **repl.it** (https://repl.it), позаяк він підтримує 40 мов програмування: C, C++, C#, Python, Java, Javascript, HTML/CSS/JS, Ruby, PHP, Nodejs, Go, F#, Rust, Swift, R тощо. Всі програми виконуються безпосередньо у браузері на стороні клієнта. Компілятор підтримує опрацювання літер кирилиці. Крім того, після реєстрації repl.it автоматично зберігає всі програми розробника у його профілі і надає змогу поділитися кодом будь-якої своєї програми за посиланням. Відкривши таке посилання, будь-який інший користувач може створити на його основі свій проект, натиснувши на кнопку Fork.

При початковому завантаженні repl.it і виборі мови C++ його вікно у браузері матиме вигляд:

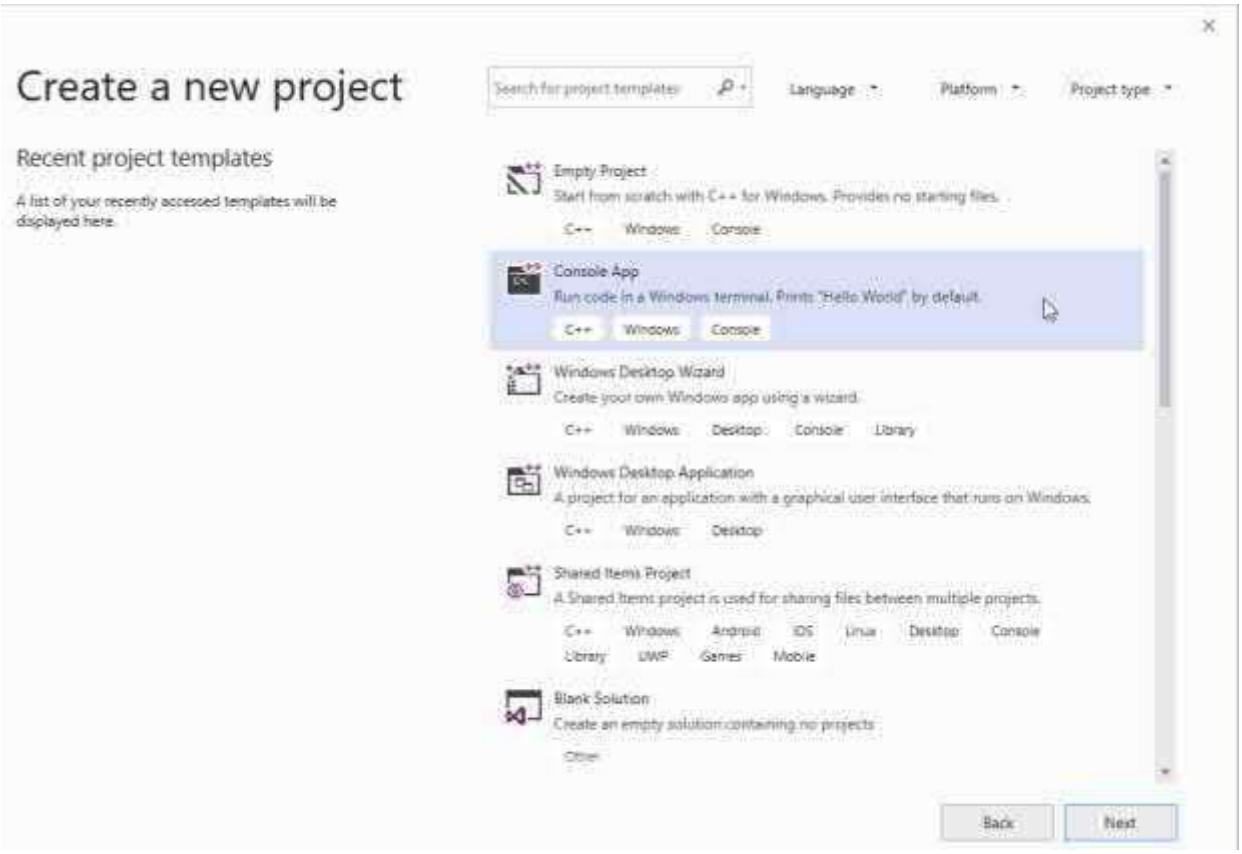
Якщо замінити цей код, наприклад, на програмний код для обчислення суми двох введених чисел і натиснути кнопку Run, результати можна буде побачити праворуч у вікні:

4. Послідовність створення програмних проєктів у Visual C++

Visual C++ є частиною комплекту Visual Studio, який дозволяє розробляти консольні програми мовою C++.

Створення консольного застосунка

Після запуску Visual Studio відкриється початкова сторінка, на якій слід виконати команду *Створити проєкт* (*Create a new project*), що призведе до відкриття діалогового вікна *Створення проєкту*. Після вибору мови C++ слід обрати вид створюваного програмного проєкту *Консольний застосунок* (*Console App*).



Не закриваючи вікно *Створення проєкту*, слід ввести ім'я програмного проєкту, при цьому для вибору папки для зберігання можна скористатися кнопкою *Огляд*. Після цього натиснути *OK* (*Create*).

Після цього відкриється вікно cpp-файлу програмного проєкту з автоматично згенерованим порожнім шаблоном головної функції:

```
#include "pch.h"
int main()
{  return 0;
}
```

Тепер можна приступити до написання програмного коду.

Як приклад наведемо програмний код обчислення суми двох чисел:

```
#include "pch.h"
#include <iostream>
using namespace std;

int main()
{
 setlocale(0,".1251");
 double a,b;
 cout << "Введіть два числа (через пробіл)\n";
 cin >> a >> b;
 cout <<"Сума цих чисел - "<< a + b << endl;
 system ("pause>>void");
 return 0;
}
```

Узагальнене тлумачення програмного коду розглянемо дещо нижче, а зараз наведемо порядок запуску програм на виконання.

Запустити програму можна одним зі способів:

– командою меню *Налагодження / Запуск без налагодження*;
– натисненням клавіш *Ctrl + F5*.

Результатом роботи програми буде поява вікна консолі з таким текстом (при натисненні будь-якої клавіші вікно закриється):

```
Введіть два числа (через пробіл)
129.5  0.75
Сума цих чисел - 130.25
```

Ще одним способом запуску програми є запуск наладчика:

– командою меню *Налагодження /Почати налагодження*;
– кнопкою  на панелі інструментів;
– натисненням клавіші *F5*.

Відмінність цього способу буде видно лише за відсутності у програмному коді останньої команди, яка організує паузу з очікуванням натиснення будь-якої клавіші. До речі, останню команду `system("pause>>void")` можна змінити парою команд:

```
cin.get();  cin.get();
```

Ще однією суттєвою відмінністю запуску програми у наладчику є можливість встановлення точок зупину, скориставшись клавішею *F9* чи то командою *Налагодження / Точка зупину*. При встановленні такої точки, після запуску програми на виконання, вона зупиниться на відповідній команді, і можна буде відслідкувати значення тієї чи іншої змінної на певному етапі виконання програми. Найпростішим способом здійснення цього є наведення покажчика миші на змінну, при цьому біля курсора сформується віконце з поточним значенням цієї змінної.

Тепер, як було обіцяно, наведемо узагальнене тлумачення вищенаведеного програмного коду, а більш глибоке розуміння Ви набудете при подальшому вивченні відповідних програмних засобів.

На початку програми для можливості коректного використання операторів введення-виведення даних `cin` та `cout` у програмному коді було долучено бібліотеку `iostream`. Крім того, оскільки ці команди належать простору імен `std`, було прописано використання цього простору – `using namespace std.`

В основній програмі `main()` на початку для можливості коректного відображення символів кирилиці було долучено використання кодування 1251. Далі оголошено та введено значення двох дійсних (тип `double`) змінних `a` та `b`. Після цього виводиться відповідний коментар та викликається функція `nod()`, числовий результат якої виводиться. Наприкінці організовується пауза (`system("pause>>void")`) як очікування натиснення будь-якої клавіші.

Зберегти всі файли програмного проєкту можна командою *Файл* / *Зберегти все*, чи то натисненням клавіш *Ctrl+Shift+S*, чи то кнопкою  на панелі інструментів.

Як змінювати колір фону

Змінити фон і колір виведеного тексту в консолі можна використовуючи функцію `system`, в яку передати рядок: "`color <A><B>`", де `<A>` і `<B>` – шістнадцяткові цифри – перша задає колір фону, а друга – колір переднього плану (колір шрифту).

Значення цифр:

0 – чорний	5 – ліловий	A – світло-зелений
1 – синій	6 – жовтий	B – світло-блакитний
2 – зелений	7 – білий	C – світло-червоний
3 – блакитний	8 – сірий	E – світло-жовтий
4 – червоний	9 – світло-синій	F – яскраво-білий

Приклад:

```
system("color F0"); // Встановити білий фон і чорний текст
```

Контрольні запитання для самоконтролю

1) Що таке консольна програма?
2) Назвати послідовність створення консольного проєкту мовою C++.
3) Які функції введення-виведення даних у консольному режимі Вам відомі?
4) За допомогою якої директиви до програми долучають бібліотечні модулі (заголовні файли)?
5) Які заголовні файли слід долучити для використання функцій введення-виведення?
6) В які способи можна запустити проєкт на виконання?
7) В який спосіб можна зберегти програмний проєкт?

Лабораторне завдання

1) У протоколі лабораторної роботи дати відповіді на контрольні запитання.

2) У протоколі лабораторної роботи записати програму мовою C++ для розв’язання завдання.

3) Засобами C++ створити на комп’ютері програму для визначення суми двох чисел у вигляді консольного застосунка.

4) Дописати програмний код ще й для визначення різниці двох чисел у цьому консольному застосунку.

Самостійна робота № 1

Запис математичних виразів мовою С++

Мета роботи: набути практичних навиків записування арифметичних виразів мовою С++.

Теоретичні відомості

1. Алфавіт мови С++

Алфавіт мови – набір символів, які можна використовувати для записування програмного коду: це великі та малі літери латиниці, цифри, знаки операцій та спеціальні символи. Кожен зокрема чи в комбінаціях *знаки операцій* і *спеціальні символи* дозволяють задавати лексеми, вирази. Список усіх операцій у порядку спадання їхніх пріоритетів наведено у додатку В.

Змінна – іменована ділянка оперативної пам'яті, застосовувана для зберігання даних під час роботи програми. При оголошенні змінної для неї резервується певна ділянка пам'яті, розмір якої залежить від конкретного типу змінної. *Значення змінної* – фактичне значення, яке міститься у цій ділянці пам'яті.

Ім'я (ідентифікатор) змінної може складатися з літер латиниці, цифр та символу «_» (підкреслювання), але неодмінно має розпочинатися з літери чи символу підкреслювання. Великі й малі літери розрізнюються, тобто мова С++ є чутливою до регістру, наприклад – *a* та *A* – це два різних об'єкти. Ідентифікатор створюється на етапі оголошення змінної, функції, структури тощо. Після цього його можна використовувати в командах розроблюваної програми.

Коментарі застосовуються лише для пояснень і жодних дій у програмі не спричинюють, тому що ігноруються компілятором. Текст коментарів розміщують поміж знаків `/*` та `*/`. Наприклад:

```
/*  Коментар до програми
може займати кілька рядків    */
```

Окрім символів `/*` та `*/` для невеликих коментарів в один рядок у С++ використовують знак `//`.

```
//  Текст після цього символу і до кінця рядка є коментарем.
```

Також, окрім пояснень, доволі часто коментарі використовують при налагодженні для тимчасового «вимкнення» певного фрагмента програми.

2. Типи даних С++

У програмі мовою С++ усі змінні мають бути оголошеними, тобто для кожної змінної має бути зазначено її тип. На відміну від інших мов, *у С++ задавати тип змінної можна в будь-якому місці програми до її використання*. При оголошенні змінної для неї резервується ділянка пам'яті, розмір якої залежить від типу змінної.

Тип змінної – вид і розмір даних, які змінна може зберігати. Кожен тип даних зберігається й опрацьовується за певними правилами. Слід зауважити, що розмір одного й того самого типу даних може відрізнятися на комп'ютерах різних платформ, а також залежить від налагоджень компілятора.

Усі типи мови С++ поділяють на дві групи: основні типи та структуровані. Список основних типів даних С++ із зазначенням діапазону та прикладами можливих значень змінних наведено в табл. 1.1.

До *основних (базових)* типів можна віднести `char`, `int`, `float` та `double`, а також їхні варіанти зі специфікаторами `short` (короткий), `long` (довгий), `signed` (зі знаком) та `unsigned` (без знака).

Структуровані (похідні) типи базуються на основних, до них належать масиви будь-яких типів, вказівники, функції, класи, файли, структури, об'єднання, перерахування тощо.

Таблиця 1.1

Основні типи даних С++

Тип	Назва	Розмір, байт	Діапазон	Приклади можливих значень	Типи чисел
`char`	символьний (знаковий)	1	–128…127	'a', '\n', '9'	цілі
`unsigned char`	беззнаковий символьний	1	0…255	1, 233	
`short`	короткий цілий	2	–32 768…32 767	1, 153, –349	
`unsigned short`	беззнаковий короткий	2	0…65 535	0, 4, 65 000	
`int`	цілий (знаковий)	4*	–2 147 483 648… …2 147 483 647	–30 000, 0, 690	
`unsigned int`	беззнаковий цілий	4	0…4 294 967 295	2 348, 60 864	
`long`	цілий (знаковий)	4	–2 147 483 648… …2 147 483 647	–30 000, 0, 690	
`float`	дійсний одинарної точності	4	$3.4\cdot10^{-38}$…$3.4\cdot10^{38}$	3.23, –0.2 100.23, 12, –0.947, 0.0001, 6.34e–3, 4e5	дійсні
`double`	дійсний подвійної точності	8	$1.7\cdot10^{-308}$ … …$1.7\cdot10^{308}$		
`long double`	довгий дійсний	10	$3.4\cdot10^{-4932}$… …$1.1\cdot10^{4932}$		
`bool`	логічний	1	false чи true	false(0), true(>=1)	
`enum`	перераховний	2 або 4			
`void`	порожній, без значення				

* – залежно від налагоджень компілятора й апаратних характеристик тип `int` може мати 4 або 2 байти.

Для подання *цілих чисел* використовують типи char, short, int, long. Специфікатор unsigned застосовують при роботі з додатними числами (без знака), а специфікатор signed – для яких завгодно чисел як додатних, так і від'ємних. За замовчуванням призначається знаковий тип, а тому специфікатор signed зазначати необов'язково.

Типи float і double визначають *дійсні змінні* розміром у 32 і 64 біти відповідно, а long double – 80 бітів. В C++ для відокремлення цілої частини числа від дійсної застосовується десяткова крапка. Окрім звичної форми, дійсні константи можна записувати у формі з рухомою крапкою. Наприклад: 2.38e3 (яке дорівнює $2.38 \cdot 10^3 = 2380$), 3.61e–4 (яке дорівнює $3.61 \cdot 10^{-4} = 0.000361$). Число перед символом "e" називається *мантисою*, а після символу "e" – *порядком*, тобто замість основи 10 використовується літера "e", після якої ставиться показник степеня, наприклад: 1e3 ($1 \cdot 10^3 = 1000$), –2.7e–4 ($-2.7 \cdot 10^{-4} = 0.00027$).

Отже, на базових типах (char, int, float, double) будується решта типів даних за допомогою специфікаторів:

signed (знаковий – за замовчуванням) та unsigned (беззнаковий) – до цілих типів char та int;

long (довгий), short (короткий) – до типів double та int.

Наприклад:

```
unsigned int n;
int c;          // Інтерпретується як  signed int c
short  a;       // Інтерпретується як  signed short int a
unsigned d;     // Інтерпретується як  unsigned int d
signed f;       // Інтерпретується як  signed int f
```

При оголошуванні змінних їм можна присвоювати початкові значення, які надалі може бути змінено. У прикладі

```
int i, j = 1, q = 0xFFF;
```

ціла змінна i не є ініціалізована; j – ініціалізована значенням 1; q – ініціалізована шістнацятковим значенням 0xFFF (десяткове 4095).

При ініціалізації змінних їм можна присвоювати арифметичні вирази:

```
long MB=1024*1024;
```

Наведемо ще кілька прикладів оголошення змінних:

```
char tol='a';     // Символьна змінна tol ініціалізується символом 'a'
char x, c='\n';   // x не є ініціалізований, c – ініціалізований символом <Enter>
char *s="Одеса"; // Рядок символів
```

Якщо при оголошенні числові змінні не ініціалізовано, то їхні значення є невизначені (випадкові).

Тип void (порожній, без значення) використовують для зазначення типу функцій, які не повертають жодного результату, а всі дії та обчислення виконуються всередині цих функцій. Також цей тип застосовують для зазначення порожнього списку аргументів функції.

Для *визначення розміру пам'яті*, займаної змінною, існує операція sizeof(), яка повертає значення довжини зазначеного типу, наприклад:

```
a = sizeof(int);              // a = 4
b = sizeof(long double);      // b = 10
```

3. Константи

Константа – величина, яка не змінюється упродовж виконування програми. Для оголошення константи у програмі використовується специфікатор `const`. При цьому зазначається тип константи й обов'язково надається значення, наприклад:

```
const double  Pi = 3.14159;
```

У якості значення константи можна подавати константний вираз, який містить раніш оголошені константи та змінні. Наприклад:

```
const double  Pi2 = 2 * Pi,  k = Pi / 180;
```

4. Математичні функції

Математичні функції широко використовуються для записування різних математичних виразів. Список математичних функцій С++ наведено у табл. 1.2.

Таблиця 1.2

Основні математичні функції С++

Функція С++	Опис
int **abs**(int i)	модуль (абсолютне значення) цілого числа x – $\|x\|$
double **fabs**(double x)	модуль дійсного числа x – $\|x\|$
double **sqrt**(double x)	корінь квадратний – $\sqrt{x}$
double **pow**(double x, double y)	піднесення x до степеня y – x^y
double **exp**(double x)	експонента e^x
double **log**(double x)	натуральний логарифм – $\ln(x)$
double **log10**(double x)	десятковий логарифм – $\lg(x)$
double **cos**(double x)	косинус – $\cos(x)$
double **sin**(double x)	синус – $\sin(x)$
double **tan**(double x)	тангенс – $\mathrm{tg}(x)$
double **acos**(double x)	арккосинус – $\arccos(x)$
double **asin**(double x)	арксинус – $\arcsin(x)$
double **atan**(double x)	арктангенс – $\mathrm{arctg}\ (x)$
double **cosh**(double x)	гіперболічний косинус – $(e^x + e^{-x})/2$
double **sinh**(double x)	гіперболічний синус – $(e^x - e^{-x})/2$
double **ceil**(double x)	округлення доверху: найменше ціле, не менше за x
double **floor**(double x)	округлення донизу: найбільше ціле, не більше за x

Стандартні математичні функції С++ розміщені у заголовному файлі (бібліотеці) `math.h`, який долучають директивою `#include <math.h>`.

До речі, константи `math` (наприклад, `M_PI` ($\pi \approx 3.14159$), `M_E` (число Ейлера $e \approx 2.71828$, яке є основою натуральних логарифмів) та багато інших) не вказані у стандартному С/С++. Щоб використовувати їх, слід спочатку визначити `_USE_MATH_DEFINES`, а тоді долучити `cmath` або `math.h`.

```
#define _USE_MATH_DEFINES
#include <math.h>    // для С або #include <cmath>  для С++
```

До речі, для обчислення значення логарифма за будь-якою основою в мові С доведеться використовувати вираз `log(x)/log(N))`.

5. Правила записування арифметичних виразів

Вираз, який завершується крапкою з комою, є **оператором**. Оператор задає закінчений опис певної дії.

Арифметичний вираз складається із операндів, арифметичних операцій (+ , –, *, /, %, ++, --) і оператора присвоювання (=).

Таблиця 1.3

Арифметичні операції

Позначення	Операція	Типи операндів і результату	Приклади
+	додавання	арифметичний, вказівник	x + y
–	віднімання та унарний мінус	арифметичний, вказівник	x – y
*	добуток	арифметичний	x * y
/	ділення	арифметичний	x / y
%	остача від ділення цілих чисел	цілий	i % 6
++	збільшення на одиницю (інкремент)	арифметичний, вказівник	i++; ++i
--	зменшення на одиницю (декремент)	арифметичний, вказівник	i--; --i

Результатом операції обчислення остачі від цілочислового ділення (%) є залишок від ділення першого операнда на другий. Операндами цієї операції мають бути цілі числа.

```
int n = 49, m = 10, x, y;
x = n / m;    // x = 4
y = n % m;    /* y = 9    */
```

Текст, який розміщений після двох скісних рисок (//) (або у середині /* */), є **коментарем** і не береться до уваги при компілюванні файлу.

Операції ++ (**інкремент**) та -- (**декремент**) є унарними, тобто мають лише один операнд. Операція ++ додає одиницю до операнда, операція –– віднімає одиницю від операнда. Ці операції можуть бути записані як праворуч, так і ліворуч операнда. Залежно від місця розміщення операції відносно операнда розрізняють дві форми цих операцій: префіксну та постфіксну. У *префіксній* формі, в якій операцію розміщують перед операндом, наприклад: ++i, --j, спочатку збільшується або зменшується на одиницю значення змінної, а вже потім ця змінна з її новим значенням бере участь в арифметичному виразі. У *постфіксній* формі цих операцій, навпаки, операцію розміщують після операнда, наприклад: i++, j--, й у виразі спочатку використовується поточне значення цієї змінної, а потім збільшується або зменшується її значення.

Чотири нижченаведені оператори дають одинакові результати, але мають різницю при використанні у виразах:

```
int  j, i = 1;
i = i + 1;  i += 1;  ++i;  i++;
```

Надамо ще три приклади використання операцій інкремента. Всі вони будуть виконуватись за початкового значення `i = 1`:

1) `j = ++i * ++i;` *// 1) i=i+1=2, 2) i=i+1=3, 3) j=i*i=3*3=9. Результат i=3, j=9.*
2) `j = i++ * i++;` *// 1) j=i*i=1*1=1, 2) i=i+1=2, 3) i=i+1=3. Результат i=3, j=1.*
3) `j = i++ * ++i;` *// 1) i=i+1=2, 2) j=i*i=2*2=4, 3) i=i+1=3. Результат i=3, j=4.*

Обчислення в арифметичних виразах виконуються зліва направо згідно з таким ***пріоритетом*** операцій:

1) стандартні функції, `++`, `--`;
2) множення (*), ділення (`/`), остача від ділення (`%`);
3) додавання (`+`) та віднімання (`–`).

Вирази у круглих дужках виконуються першочергово.

Для здобуття правильного результату слід дотримуватися таких правил записування арифметичних виразів в операторах С++:

– кожна команда (інструкція) має завершуватись крапкою з комою (;);
– мова С++ є чутлива до регістру, тобто `x` та `X` – це дві різні змінні;
– аргумент функції завжди записують у круглих дужках;
– знаки множення не можна пропускати (`3ab -> 3*a*b`);
– якщо знаменник або чисельник має операції (`+`, `–`, *, `/`), то його слід записувати у круглих дужках;
– для записування раціональних дробів, у чисельнику або знаменнику яких є числові константи, хоча б одну з цих констант слід записати як дійсне число із зазначенням десяткової крапки, наприклад, $\frac{2}{k}$ записують як `2.0/k`;
– радикали (тобто корінь кубічний і вище) замінюють на дробові степені, наприклад, $\sqrt[3]{x+1}$ записують як `pow(x+1, 1/3.0)`;
– слід враховувати правила зведення типів, оскільки в арифметичних виразах можуть брати участь різнотипні дані та відбувається зведення типів.

6. Оператори присвоювання

Оператор присвоювання – основний оператор програмування – має формат:

<*ім'я_змінної*> = <*вираз*>;

Цей оператор обчислює значення *виразу* і надає здобуте значення *змінній*; при цьому слід враховувати відповідність типів виразу і змінної.

У мові С++, крім простого присвоювання, є і складені операції присвоювання: `+=` (присвоювання з додаванням), `– =` (присвоювання з відніманням), `*=` (присвоювання з множенням), `/=` (присвоювання з діленням) тощо. Наприклад, вираз `x += y` є еквівалентний до виразу `x = x + y`, але записується компактніше і виконується швидше.

Особливістю оператора присвоювання в С++ є те, що він може використовуватись у виразах і допускає багатократне застосування, наприклад:

```
a = b = c = x * y;
```

Виконується ця команда справа наліво, тобто спочатку обчислюється значення виразу `x * y`, після чого це значення присвоюється `c`, потім `b` і лише потім `a`.

7. Зведення типів

Перетворення типів виконуються, якщо операнди, які входять до виразу, мають різні типи. Зведення типів здійснюється автоматично за правилом: ***менш точний тип зводиться до більш точного***, тобто до типу того операнда, котрий має більший розмір. Наприклад, якщо в арифметичному виразі беруть участь коротке ціле `short` та ціле `int`, результат зводиться до `int`, якщо ціле та дійсне – до дійсного.

Розглянемо ще кілька правил зведення типів.

1) Результат операції ділення буде цілим числом, якщо ділене і дільник є цілими, і дійсним числом, якщо один з операндів є дійсного типу. Наприклад, результатом `2/3` буде `0`, а результатом `2.0/3` (або `2./3`) – `0.666(6)`.

2) При присвоюванні результат зводиться до типу змінної, яка стоїть ліворуч "="; при цьому тип може як підвищуватися, так і знижуватись, наприклад:

```
float a = 5.1, b = 1.5;
int c = a * b; // c=7, а дробова частина від результату 7.65 буде втрачена.
```

3) Зведення типів може бути явним із зазначенням конкретного типу:

(<*тип*>) <*арифметичний_вираз*>

Наприклад:

```
int m = 2, n = 3;
double a = (double) m/n;      // в результаті  a=0.66(6).
```

Іншим рішенням може бути домноження чисельника чи знаменника на 1.0:

```
double a =  (m*1.0)/n;        // в результаті  a=0.66(6).
```

Приклади записування арифметичних виразів мовою С++

Приклад 1. Сформувати вираз мовою С++ $y=\dfrac{0.2x^2-x}{(\sqrt{3}+x)(1+2x)}+\dfrac{2(x-1)^3}{\sin^2 x+1}$.

```
y=(0.2*x*x-x)/((sqrt(3.)+x)*(1+2*x)) + 2*pow(x-1,3)/(pow(sin(x),2)+1);
```

Приклад 2. Записати за правилами мови С++ такі математичні вирази: $y=\sqrt[3]{\dfrac{\sin^2(mp-\pi)}{bx+p}}$; $x=\operatorname{arctg}\left(b^2+\sqrt{(b+m)}\right)$ і $p=\cos\left(\mathrm{e}^{|b-x|}-\lg|x-b|\right)^2$.

```
y=pow(pow(sin(m*p-pi),2)/(b*x+p), 1./3);
x=atan(b*b+sqrt(b+m));
p=cos(pow(exp(fabs(b-x))-log10(fabs(x-b)),2));
```

Приклад 3. $V=\lg^5\left(x^3\ln^6\left(x^2+1.7\right)+\mathrm{e}^{\sqrt{x}}\right)-\operatorname{arctg}\left|\sqrt[3]{x}+\operatorname{tg}^3 x^8\right|$

```
V=pow(log10(x*x*x*pow(log(x*x+1.7),6)+exp(sqrt(x))),5)-
  atan(fabs(pow(x,1.0/3)+pow(tan(pow(x,8)),3)));
```

Контрольні запитання та завдання для самоконтролю

1) Який процес називають лінійним?
2) Записати константу `0.2731e3` у фіксованому форматі.
3) Записати число `0.0001` в експоненційній формі з рухомою крапкою.
4) Записати вираз $y = x^3 + \sin 2x$ засобами С++.

Завдання до самостійної роботи

1) Дати відповіді на контрольні запитання.

2) Записати мовою С++ арифметичні вирази, наведені в табл. 1.4 … 1.6 відповідно до індивідуального варіанта.

Таблиця 1.4

Варіанти арифметичних виразів

№ вар.	Арифметичний вираз	№ вар.	Арифметичний вираз	№ вар.	Арифметичний вираз
1	$Z = \frac{2t + y\cos t}{\sqrt{y + 4.831}}$	2	$D = y^2 + \frac{0.5n + 4.8}{\sin y}$	3	$Q = \frac{\sqrt{k + 2.6p\sin k}}{x - d^3}$
4	$F = \ln(d) + \frac{3.5d^2 + 1}{\cos(2y + 2.3)}$	5	$R = \frac{\sin(2t+1)^2 + 0.3}{\ln(t + y)}$	6	$L = \cos^2 c + \frac{3t^2 + 4}{\sqrt{c + t}}$
7	$U = \frac{\ln(k - y) + y^4}{e^y + 2.355k^2}$	8	$A = \frac{\sin(2y + h) + h^2}{e^h + y}$	9	$R = \frac{\sin^2 y + 0.3d}{e^y + \ln(d)}$
10	$G = \frac{9.33w^3 + \sqrt{w}}{\ln(y + 3.5) + \sqrt{y}}$	11	$P = \frac{e^{y+2,5} + 7.1h^3}{\ln\sqrt{y + 0.04h}}$	12	$U = \frac{\ln(2k + 4.3)}{e^{k+y} + \sqrt{y}}$
13	$D = \frac{7.8a^2 + 3.52t}{\ln(a + 2y) + e^y}$	14	$F = \frac{2\sin(0.354y + 1)}{\ln(y + 2j)}$	15	$T = \frac{\sin(2 + u)}{\ln(2y + u)}$
16	$L = \frac{0.81\cos i}{\ln(y) + 2i^3}$	17	$W = \frac{4t^3 + \ln(r)}{e^{y+r} + 7.2\sin r}$	18	$G = \frac{e^{2y} + \sin(f + 3)}{\ln(3.8y + f)}$
19	$N = \frac{m^2 + 2.8m + 0.355}{\cos 2y + 3.6}$	20	$H = \frac{y^2 - 0.8y + \sqrt{y}}{23.1n^2 + \cos n}$	21	$Z = \frac{\sin(p + 0.4)^2}{y^2 + 7.325p}$
22	$T = \frac{2.37\sin(t + 1)}{\sqrt{4y^2 - 0.1y + 5}}$	23	$R = \frac{\sqrt{\sin^2 y + 6.835}}{\ln(y + k) + 3y^2}$	24	$W = \frac{0.004v + e^{2y}}{e^{y/2}}$
25	$V = \frac{(y + 2w)^3}{\ln(y + 0.75)}$	26	$E = \frac{\ln(0.7y + 2q)}{\sqrt{3y^2 + 0.5y + 4}}$	27	$T = \frac{0.355h^2 - 4.355}{e^{y+h} + \sqrt{2.7y}}$
28	$S = \frac{4.351y^3 + 2t\ln(t + x)}{\sqrt{\cos 2y + 4.351}}$	29	$K = \frac{2t^2 + 3l + 7.2}{\ln(y) + e^{2l}}$	30	$N = \frac{3y^2 + \sqrt{y + 1}}{\ln(p + y) + e^p}$

Таблиця 1.5

Варіанти арифметичних виразів

№ вар.	Арифметичний вираз	№ вар.	Арифметичний вираз
1	$F = \cos\left(x^2 + 2\right) + \dfrac{3.5x^2 + 1}{\cos^2 y}$	2	$T = \dfrac{\sqrt{x + b - a} + \ln(x)}{\operatorname{ctg}(b + a)}$
3	$R = \dfrac{\sin\left(x^2 + a\right)^3 + 4.3^a}{\cos^3 x^4}$	4	$D = \dfrac{K^{-arx} - a\sqrt{6} - \cos(3ab)}{\sin^2(a \arcsin x + \ln y)}$
5	$L = \operatorname{ctg}^2 c + \dfrac{2x^2 + 5}{\sqrt{c + t}}$	6	$U = \dfrac{\ln\left(x^3 + y\right) - y^4}{\mathrm{e}^y + 5.4k^3}$
7	$P = \dfrac{a^5 + \arccos(a + x^3) - \sin^4(y - c)}{\sin^3(x + y) + \lvert x - y\rvert}$	8	$A = \dfrac{\operatorname{tg}\left(y^3 - h^4\right) + h^2}{\sin^3 h + y}$
9	$F = \dfrac{\sqrt{(2 + y)^2 + \sqrt[7]{\sin(y + 5)}}}{\ln(x + 1) - y^3}$	10	$R = \dfrac{\cos^2 y + 2.4d}{\mathrm{e}^y + \ln\left(\sin^2 x + 6\right)}$
11	$G = \dfrac{\operatorname{tg}(x^4 - 6) - \cos^3(z + xy)}{\cos^4 x^3 c^2}$	12	$F = \dfrac{\sqrt{\lvert x\rvert + \cos^3 x + z^4}}{\ln x - \arcsin(bx - a)}$
13	$U = \dfrac{\operatorname{tg}^3 y + \sin^5 x\sqrt{b - c}}{\sqrt{a - b + c}}$	14	$D = \dfrac{\cos(x^3 + 6) - \sin(y - a)}{\ln x^4 - 2\sin^5 x}$
15	$P = \dfrac{a^5 + \sin^4(y - c)}{\sin^3(x + y) + \lvert x - y\rvert}$	16	$G = \dfrac{\operatorname{tg}(x^4 - 6) - \cos^{3x}(z + x^3 y)}{\cos^2 x^3 + c^2}$
17	$R = \dfrac{\cos^3 y + 2^x d}{\mathrm{e}^y + \ln\left(\sin^2 x + 7.4\right)}$	18	$S = \dfrac{4.351y^3 + 2t\ln(t)}{\sqrt{\cos 2y + 1}}$
19	$U = \dfrac{\mathrm{e}^{x^3} + (\cos^2 x - 4)}{\operatorname{arctg} x + 5.2y}$	20	$N = \dfrac{\sqrt[5]{z + \sqrt{zx}}}{\mathrm{e}^x + a^5 \operatorname{arctg} x}$
21	$I = \dfrac{2.33\ln\sqrt{1 + \cos^2 y}}{\mathrm{e}^y + \sin^2 x}$	22	$K = \dfrac{\sqrt{(3a + x)^6 - \ln x}}{\mathrm{e}^{a+x} + \arcsin 6x^2}$
23	$K = \dfrac{\cos^3\lvert y + x\rvert - (x + y)}{\operatorname{arctg}^4(x + a)x^5}$	24	$R = \dfrac{\sqrt{\sin^2 y + 6.835}}{\ln(y + k) + 3y^2}$
25	$R = \dfrac{a}{x - a} + \dfrac{b^x + \cos^3 x}{\lg^3 a + 4.5}$	26	$G = \dfrac{9.33w^3 + \sqrt{w}}{\ln(y + 3.5) + \sqrt{y}}$
27	$L = \dfrac{\sqrt{\mathrm{e}^x - \cos^4(x^2 a^5)} + \operatorname{arctg}^4(a - x^5)}{\sqrt{\lvert a + xc\rvert}}$	28	$f = \dfrac{\cos^7 bx^5 - (\sin a^2 + \cos(x^3 + z^5))}{(\arcsin a^2 + \arccos(x^7 - a^2))}$
29	$N = \dfrac{m^2 + 2.8m + 0.355}{\cos 2y + 3.6}$	30	$H = \dfrac{y^2 - 0.8y + \sqrt{y}}{23.7n^2 + \cos n}$

Таблиця 1.6

Варіанти арифметичних виразів

№ вар.	Арифметичний вираз
1	$V1 = \sqrt{x^2 + \left(\sqrt{\text{arctg}\lvert x\rvert - e^2} \,/ \sin^2\left(x^3 + 1.8\right)\right)^4} + 2.8^{\sqrt{x}}$
2	$V2 = \left(\sin x - 5.4\right)^{3x} + \sqrt[3]{\left\lvert \lg\left(x - 1.5\right)^2 \right\rvert + x^{3.5}}$
3	$V3 = x^{2.8} \Big/ \left(\cos^2\left(x^3 - 1.5\right)^4 + \sqrt{\lvert x\rvert}\right) - \text{arctg}\left(x / \ln\lvert x\rvert\right)^5$
4	$V4 = \sin^5\left(x^4 - \sqrt[3]{\lg^4\left(x^2 - \ln^2\left(x - 1.8\right)\right)}\right) + \text{arctg}^2\, x$
5	$V5 = \left(15.4^x - x^{3.9}\right) \Big/ \sqrt{x^2 + \lg^2 \ln^3\left\lvert x^3 - 1.8\right\rvert} + 9^{5.3}$
6	$V6 = e^{\sqrt{\lg^3\left(x^2 - 1.8\right)^3}} + x^{4.5} / \text{arctg}(x^2 + a^2)^4 - \sqrt{x^{3.2}}$
7	$V7 = \left(\cos^3 x^{1.5} + \sin^2 x^3\right)^4 \Big/ \left(\lg^2\left(\lvert x\rvert + e^{\sqrt[3]{x+1.8}}\right) + \sqrt{x}\right)$
8	$V8 = \lg^4\left(\ln^3\left(x^2 + \sqrt{\lvert x\rvert}\right) \Big/ \left(x^3 + e^x\right)^3\right)^5 - x^{3.5} / \sin^2\left(x^3 + 1.8\right)$
9	$V9 = \cos^5\left(x^2 + \text{arctg}\sqrt{\lvert x - 1.8\rvert}\right) \Big/ \sin^2\left(x^2 + 1.5\right)^5 + \sqrt[3]{x^{3.5}}$
10	$V10 = \text{arctg}^5\left(\sin^3\left(x^2 + 1.8\right)^5 - \sqrt{x}\right)^4 - e^{3.8} \Big/ \left(x^{4.5} + \sqrt{\lvert x\rvert}\right)$
11	$V11 = \cos^3\left(x^4 + \lg^2 \ln^3\left(\sqrt{x} - \sqrt[3]{\lvert x\rvert}\right)^2\right) + \left(4.8^{\sqrt[3]{x}} - \sqrt{\lvert x\rvert}\right)^5$
12	$V12 = \sin^8\left(\sqrt{x} + \sqrt[3]{\lvert x^2 + 1.8\rvert} \Big/ \cos^2\left(x^3 - 1.5\right)^6\right) - x^{3.7}$
13	$V13 = e^x\left(\cos^2\left(\lvert x\rvert - \sqrt[3]{x^4 + 5.3}\right)^4 + \text{arctg}^3\, x^2\right)^5 + x^2 / \left(1 + x^{6.6}\right)$
14	$V14 = \sqrt[3]{\lvert x\rvert + \sqrt{x^3 + 1.3}} \Big/ \cos^2\left(x^3 (1 + x)^4\right) \Big/ e^{\sqrt{x}} - x^{7.5}$
15	$V15 = \sin^5\left(\sqrt{\lvert x\rvert} + \cos^3\left(x^2 + 5.4\right)\right) - \text{arctg}\left(e^{\sqrt{x}} + 5.8^{3.7}\right)$
16	$V16 = \left(x^2 + \sqrt{\sin^2 x - \ln\left(x^2 - 3\right)}\right) \Big/ \lg^2\left(x + \sqrt[3]{5.5x + \ln x}\right) - e^{2.5 + x^2}$
17	$V17 = \cos^2\left(x^2 + \sqrt{x + 2}\right) \Big/ \sin\left(x^2 + \sqrt{x}\right) + \ln^2 x / \left(\lg x + e^{\sqrt{x}}\right)$
18	$V18 = \sqrt{x^2 + \sin\left(\sqrt{x} + 2x\right)} - e^{2x + \sqrt{x}} / \left(\cos^2 x + \lg^2\left(\ln x\right)\right)$
19	$V19 = \sqrt[3]{x + \cos^2 x + \sin x^2 + \lg x} \Big/ \left(x^2 + \ln^2 x^3 - e^{\sqrt{x}}\right)$
20	$V20 = \sqrt{x + \sqrt{x + \sqrt{x + \sin^2 x}}} \Big/ \cos\left(x^2 + \ln^2\left(1 + e^{\sqrt{x}}\right)\right)$
21	$V21 = \left(x^2 + \cos\left(x + \ln\sqrt{x^3 + 1.8}\right)^2 + \sqrt{\lvert x\rvert}\right) \Big/ \lg\left\lvert x + e^{\sqrt{x}}\right\rvert$
22	$V22 = \sin^3\left(x^2 \sqrt[3]{x + \sqrt{x^2 + 1.5}}\right) \Big/ \text{arctg}^2\left(x^2 + 2.5e^x\right)$
23	$V23 = \cos^2\left(x + \sin\left(\sqrt{x^3 + \sqrt{x + 1.5}} - \ln^2 x\right) + e^x\right)$

Закінчення табл. 1.6

№ вар.	Арифметичний вираз
24	$V24 = \left(x^{4.5}\, e^{\sqrt{x}}\right) / \ln^2\left(x^2 + \cos\left(x + \sqrt[3]{x}\right)\right) - \sin^3\left\|x - e^3\right\|$
25	$V25 = \operatorname{arctg}^2\left(x^2 + \cos\left(\sqrt{x + \left\|x^2 - e^x\right\|}\right)\right) / \lg^2\left(x^3 - \sin x\right)$
26	$V26 = \sin^2\left(x^2 + \sqrt[3]{x + \cos^2\left\|x\right\|}\right) / \ln^2\left(\sqrt{x} - \operatorname{arctg}^2\left\|x\right\|\right)^2$
27	$V27 = \ln^2\left(\sqrt{\left\|x\right\|} + x^2 + \sin\left\|x\right\|\right) / \lg^2\left(e^{\sqrt{x}} + x^4\right)$
28	$V28 = \lg\left(\ln^2(x)\left\|x^2 + \sqrt{x^2 + 1.5}\right\|\right) / e^{\sqrt{x}} + \cos^3 x^2$
29	$V29 = \cos^3\left(\operatorname{arctg}^2\left(\left\|x + \sqrt[3]{x}\right\|\right) + \sin^2 x\right) / \sin\left\|\lg^2(x + 2.8)\right\|$
30	$V30 = \lg^5\left(x^3 \ln^6\left(x^2 + 1.7\right) + e^{\sqrt{x}}\right) - \operatorname{arctg}\left\|\sqrt[3]{x} + \operatorname{tg}^3 x^8\right\|$

ТЕМА 2
ПРОГРАМУВАННЯ БАЗОВИХ АЛГОРИТМІВ

Лабораторна робота № 2
Програмування лінійних алгоритмів

Мета роботи: набути практичних навиків створення програмних проєктів лінійної структури мовою С++.

Теоретичні відомості

Різновиди алгоритмів. Лінійні алгоритми (послідовності)

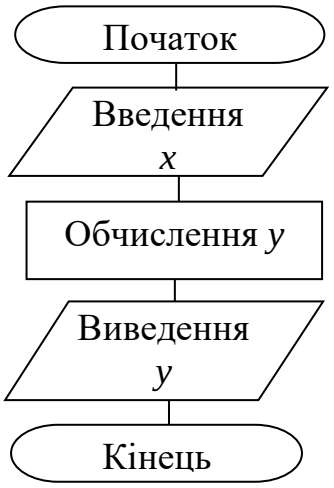
Базові алгоритми організації обчислень поділяють на три основні види:
- лінійні (послідовності);
- розгалужені;
- циклічні.

Переважно вони є окремими частинами обчислювального процесу, тоді як загальний обчислювальний процес має складнішу (комбіновану) структуру. Здебільшого при написанні програми ці базові алгоритми поєднуються в такій структурі.

Лінійним називається алгоритм, в якому всі дії, від першої до останньої, виконуються послідовно у порядку їхнього запису. Іноді таку структуру називають просто **послідовністю**. Типовим прикладом такого алгоритму є стандартна обчислювальна схема, яка складається з трьох етапів:
- введення початкових даних;
- обчислення за формулами;
- виведення результату.

Приклади програм з лінійною структурою в С++

Приклад 1. Розробити схему алгоритму і створити програму для обчислення $y = \frac{0.2x^2 - x}{(\sqrt{3}+x)(1+2x)} + \frac{2(x-1)^3}{\sin^2 x + 1}$, де x – довільна змінна, яку слід ввести.



Текст програмного коду та схема алгоритму:

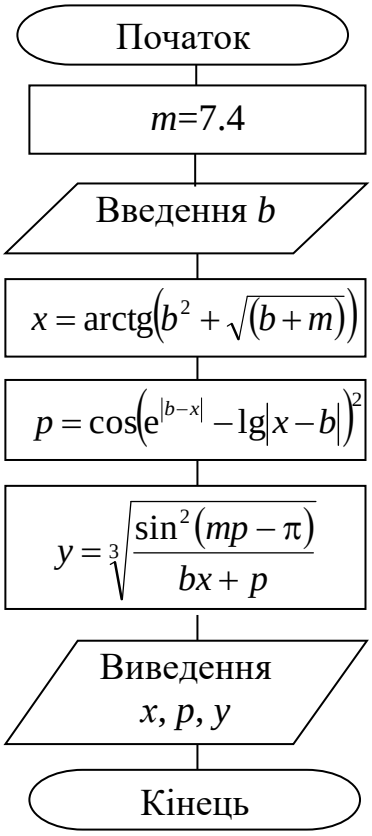
```
#include <iostream>
#include <math.h>
using namespace std;
```

```
int main ()
{
 setlocale(0,".1251");
 double y, x;  // Оголошення змінних
 cout<<"Введіть значення x= ";
 cin>>x;
 y=(0.2*x*x-x)/((sqrt(3.)+x)*(1+2*x))+2*pow(x-1,3)/(pow(sin(x),2)+1);
 cout<<"Результат y= "<< y <<endl;
 system ("pause>>void");
 return 0;
}
```

Результати роботи:

```
Введіть значення x= 2.15
Результат y= 1.72928
```

Приклад 2. Розробити схему алгоритму і програмний проєкт для обчислення $y=\sqrt[3]{\dfrac{\sin^2(mp-\pi)}{bx+p}}$; $x=\operatorname{arctg}\left(b^2+\sqrt{(b+m)}\right)$ і $p=\cos\left(e^{|b-x|}-\lg|x-b|\right)^2$. Значення m = 7.4 задати як константу, а довільне значення змінної b слід ввести.



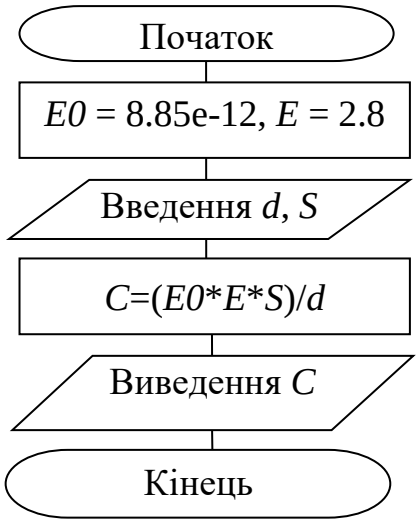
Текст програми та схема алгоритму:

```
#include <iostream>
#include <math.h>
using namespace std;
int main ()
{
  setlocale(0,".1251");
  const double m=7.4, pi=3.1415;
  double b, x, p, y;
  cout<<"Введіть значення  b= ";
  cin>>b;
  x = atan(b*b+sqrt(b+m));
  p = cos(pow(exp(fabs(b-x))-log10(fabs(x-b)),2));
  y = pow(pow(sin(m*p-pi),2)/(b*x+p),1./3);
  cout<<"Результати:\n x= "<< x <<endl;
  cout<<" p= "<< p << endl <<" y= "<< y <<endl;
  system ("pause>>void");
  return 0;
}
```

Результати роботи:

```
Введіть значення  b= 8.4
Результати:
 x= 1.55738
 p= -0.128333
 y= 0.3709956
```

Приклад 3. Обчислити ємність конденсатора за формулою $C = \frac{\varepsilon_0 \cdot \varepsilon \cdot S}{d}$, де $\varepsilon_0 = 8.85 \cdot 10^{-12}$ Ф/м, $\varepsilon = 2.8$ Ф/м, S – площа кожної пластини конденсатора, d – відстань між пластинами, значення яких ввести з клавіатури.



Текст програми та схема алгоритму:

```
#include <iostream>
#include <math.h>
using namespace std;
int main ()
{
  setlocale(0,".1251");
  double d, S, C;
  const double E0 = 8.85e-12, E = 2.8;
  cout<< "Введіть значення відстані між пластинами d= ";   cin>>d;
  cout<< "Введіть значення площі пластини конденсатора S= "; cin>>S;
  C=(E0*E*S)/d;
  cout<<"Ємність конденсатора C= "<< C <<endl;
  system ("pause>>void");
}
```

Результати роботи:

```
Введіть значення відстані між пластинами d= 0.004
Введіть значення площі пластини конденсатора S= 50000
Ємність конденсатора C=0.0030975
```

Контрольні запитання та завдання для самоконтролю

1) Який процес називають лінійним?
2) Яких значень набудуть змінні `j` та `i` після обчислення для `int j, i=2;`
   а) `j = i++ + i++;` в) `j = i++ + ++i;`
   б) `j = ++i + ++i;` г) `j = ++i + i--;`
3) Якого значення набуде змінна y після обчислення:
   а) $y = (1 + 2) / (3 + 2)$; в) $y = 1 + 2.0 / 3 + 2$;
   б) $y = 1 + 2 / 3 + 2$; г) $y = (1 + 2) / (3.0 + 2)$.

Лабораторне завдання

1) У протоколі лабораторної роботи дати відповіді на контрольні запитання.

2) У протоколі лабораторної роботи скласти схеми алгоритмів і написати програми мовою С++ для розв'язання завдань, наданих у табл. 2.1 … 2.3 відповідно до індивідуального варіанта.

3) Створити на комп'ютері програмні проєкти для реалізації написаних програм. Занести результати обчислень до протоколу.

Таблиця 2.1

Варіанти завдань для програм з лінійною структурою

(довільне значення *x* слід ввести)

№ вар.	Функція	№ вар.	Функція
1	$y=\frac{2x^2-\sin^2 x}{\cos(2x)+x^2}-\frac{x+1}{\ln x}$	2	$y=\frac{\ln x^2+\cos^2 x}{\cos(2x)+x^2}+\frac{\sqrt[3]{x}}{x}$
3	$y=\frac{\ln x^2+2\cos^2 x}{\cos(2x)^2}+\frac{\sqrt[3]{x}}{x}$	4	$y=\frac{2\cos^2 x}{1+x\cos(2x)}+\frac{0.3^x}{x\ln x-2\sin^2 x}$
5	$y=\frac{x\ +2x+\sin x}{\cos^2 x+x^2}+\frac{0{,}3^x}{\ln x}$	6	$y=\frac{2x+\sin x}{\cos^2 x+x^2}+\frac{0.5^x}{\sqrt{x}}$
7	$y=\frac{\sin x-x^2}{2x+1}+\frac{(1+x)^x}{1+3x}$	8	$y=\frac{x-\ln x}{2x-1}+\frac{2x-1}{x^2+3x}$
9	$y=\frac{\ln x+2x}{x^2+1}+\frac{x+1}{2x^2+1}$	10	$y=\frac{3x^2+2x}{\sin x+x^2}-\frac{2x}{(1+x^2)(1+2x)}$
11	$y=\frac{4x^2+3x}{(1+x)(1+2x)}+\frac{2x+1}{\sin x+1}$	12	$y=\frac{(2x^2-1)}{x^2+\sin}-\frac{2x+1}{(x+2)(x+3)}$
13	$y=\frac{(4x^2-2)(x+2)}{2x+3}+\frac{x^2\sin x}{2x+1}$	14	$y=\frac{x^2+2\sin x}{2x+1}+\frac{\sqrt{x}-\cos x}{(2x+1)(\ln x^2+1)}$
15	$y=\frac{x^2+2(x-1)}{(x+1)(x+\sqrt{3})}+\frac{2\sin^2 x}{2x+3^x}$	16	$y=\frac{4x^2-3^x}{2x^2+1}+\frac{\ln x}{2x+3}$
17	$y=\frac{3x-2}{(2x+3)(x+1)}+\frac{\sin 2x}{(x^2+1)(x+2)}$	18	$y=\frac{x^2-2x}{(2x+3)(x+4)}+\frac{\sqrt[3]{x}}{2x+3}$
19	$y=\frac{x^2+1}{x^3+3}+\frac{\sin x}{2x+3}$	20	$y=\frac{4x^2+3x}{3x+4}+\frac{\sin x}{2\cos x+1}$
21	$y=\frac{3x+2}{2x+3}+\frac{x^2}{(2x+1)(\sin x+2)}$	22	$y=\frac{4x-\sin x}{x^2+3x+1}+\frac{3x^2+2^x}{(x+1)(x^2+1)}$
23	$y=\frac{2x+\sin x}{(x+2)(x+\sqrt{x})}+\frac{4x}{(2x+\sqrt[3]{3})(x^2+1)}$	24	$y=\frac{3x+4}{(x+3)(x+1)^2}+\frac{2x-1}{(x+\sin x)(\ln x+1)}$
25	$y=\frac{4x}{(x+\sin)^2}+\frac{2x+\sqrt[3]{x}}{(x^2+1)(x+1)}$	26	$y=\frac{2x+3^x}{(x+1)(x+3)}+\frac{x^2+\sqrt{x}}{(2x+1)(x+\sin x)}$
27	$y=\frac{2x+1}{x+1}+\frac{x^2+\sqrt{x}}{(2x+1)(x+\sin x)}$	28	$y=\frac{3x^2-\sqrt[3]{x}}{(2x+1)(1+x)}+\frac{2x+1}{(x+3)}$
29	$y=\frac{2x^2+1}{x+\sin(x+1)}+\frac{x-3^x}{(2x+1)(x+2)}$	30	$y=\frac{x^2+\sin 2x}{2\sqrt{x}+3x}+\frac{x^2+1}{(x+2)(x+3)}$

Таблиця 2.2

Варіанти завдань з лінійною структурою

(перше зі значень параметрів задати як константу, друге – ввести з екрана)

№ вар.	Функція $y = f(x)$	Значення параметрів
1	$y = a\sin^2 b + b\cos^2 a;\ a = \sqrt[3]{\lvert b + c\rvert};\ b = \sqrt{x}$	x = 1.52; c = 5
2	$y = a^2 + b^2;\ a = \ln\lvert x\rvert;\ b = \mathrm{e}^k + a$	x = 5.3; k = 3
3	$y = \mathrm{e}^x + 5.8^c;\ c = a^2 + \sqrt{b};\ a = b^3 + \ln\lvert b\rvert$	x = 2.5; b = 0.7
4	$y = \sqrt[3]{\lvert a - b\rvert};\ a = \lg x;\ b = \sqrt{x^2 + t^2}$	x = 1.7; t = 3
5	$y = a^3 / b^2;\ a = \mathrm{e}^{\sqrt{\lvert x\rvert}};\ b = \left(\sin p^2 + x^3\right)$	x = 2.1; p = 2
6	$y = p^2 + t^4;\ p = x^2 - \sqrt{\lvert x\rvert};\ t = \sqrt[3]{x + a^2}$	x = 4; a = 3.7
7	$y = c^3 / \cos c;\ c = a^2 + b^2;\ a = \sqrt{\lvert x\rvert} + \mathrm{e}^{\sqrt{b}}$	x = –11; b = 12.5
8	$y = \sin^3(a + b);\ \ a = t^3 + \sqrt{b};\ b = \lg^2\lvert x\rvert$	x = 10.9; t = 2
9	$y = \mathrm{arctg}^3 x^2;\ \ x = p + k;\ \ k = \sqrt{p + t^2}$	t = 4.1; p = 3
10	$y = \cos^2(a + \sin b);\ \ a = \sqrt{\lvert x\rvert};\ \ b = x^4 + m^2$	m = 2; x = 1.1
11	$y = \sin^3 a + \cos^2 x;\ \ a = c + k^2;\ \ c = \mathrm{arctg}\lvert x\rvert$	k = 7.2; x = 5
12	$y = \mathrm{e}^{\sqrt{\lvert x\rvert}} + \cos x;\ \ x = a + c^3;\ \ a = \sin^5 b$	b = 3; c = 1.7
13	$y = a\cos x - b\sin x;\ \ x = \sqrt[3]{a - b};\ \ a = t^2 b$	t = 2.2; b = 3
14	$y = \sqrt{x}\sin a + \sqrt{b}\cos x;\ \ a = \lg\lvert x\rvert;\ \ b = x + p^3$	x = 11; p = 2.6
15	$y = \lg a / \lg b;\ \ a = \sqrt{x^2 + b^2};\ \ x = \mathrm{e}^b + n$	n = 9.1; b = 3
16	$y = \ln\lvert x + t\rvert;\ \ x = t^2 + p;\ \ t = \sqrt{m}$	m = 3.8; p = 2
17	$y = \mathrm{e}^{a+b};\ \ a = \lg\lvert t + b^2\rvert;\ \ \ t = b^2 + \sqrt{bx}$	b = 3; x = 5.2
18	$y = \sqrt[3]{x^2 + c^2};\ \ x = \mathrm{e}^{mk};\ \ c = \cos^2 m + k^2$	k = 2; m = 1.8
19	$y = \mathrm{e}^x + 5.8^c;\ \ c = a^2 + \sqrt{b};\ \ a = b^3 + \ln\lvert b\rvert$	x = 2.8; b = 3
20	$y = x^3 / t^2;\ \ x = \mathrm{e}^{\sqrt{p+a}};\ \ t = p^3 + a^3$	a = 2; p = 2.6
21	$y = c^2 + \sqrt{\lvert a\rvert};\ \ c = \lg\lvert b\rvert;\ \ a = (b + x)^3$	b = 7; x = 2
22	$y = \mathrm{arctg}^2\lvert x\rvert;\ \ x = t^3 + b^2;\ \ t = b^3 + \mathrm{e}^{\sqrt{q}}$	q = 2; b = 1.8
23	$y = v^3 + \cos^2 w;\ \ v = \cos^2 a;\ \ w = \sqrt{a + \lvert x\rvert}$	x = 2.9; a = –0.9
24	$y = x^2 + \sqrt[3]{\lvert x\rvert};\ \ x = \cos^2 b + \sin^2 a;\ \ a = \sqrt{b + t^2}$	b = 7.1; t = 2
25	$y = \sin^3 x + \cos x^2;\ \ x = \lg\lvert ct\rvert;\ \ c = t^2 + \sqrt{a}$	t = –3; a = 8.8

Закінчення табл. 2.2

№ вар.	Функція $y = f(x)$	Значення параметрів
26	$y = \lg^2\lvert x+a\rvert;\ x = \sqrt{a+b};\ a = \mathrm{e}^{t+b}$	$t = 2;\ b = 1.8$
27	$y = \mathrm{arctg}^3\lvert p\rvert;\ p = \sqrt{x^2+a^2};\ x = \sqrt{a}+\sqrt{b}$	$a = 7;\ b = 2.3$
28	$y = \sin^4\left(a^2+b^2\right);\ a = \sqrt{b+t};\ t = b^2+k^3$	$b = 5;\ k = 2.8$
29	$y = \cos^3 x+\lvert a\rvert;\ x = \mathrm{e}^b;\ b = \lvert a\rvert+\sqrt{a+p^2}$	$a = -4;\ p = 3$
30	$y = \sin^4\left(a^2+b^2\right);\ a = \sqrt{b+t};\ t = b^2+k^3$	$b = 2;\ k = 1.8$

Таблиця 2.3

Варіанти задач з лінійною структурою

№ вар.	Завдання
1	Трикутник задано координатами своїх вершин. Обчислити його площу, використовуючи формулу Герона: $S = p(p-a)(p-b)(p-c)$, де $p = (a+b+c)/2$; a, b і c – довжини сторін трикутника. Координати вершин ввести з клавіатури. Для обчислення довжини відрізка між точками (x_1, y_1), (x_2, y_2) використовувати формулу $\sqrt{(x_1-x_2)^2+(y_1-y_2)^2}$
2	Знайти період T і частоту ν коливань у контурі, ємність конденсатора в якому – C, індуктивність – L. Значення C і L ввести з клавіатури. $T = 2\pi\sqrt{LC}$, $\nu = 1/T$
3	Обчислити довжину і площу кола для заданого радіуса. Значення радіуса ввести з екрана
4	Обчислити площу трикутника за трьома сторонами – a, b, c, – використовуючи формулу Герона (див. варіант 1). Довжини сторін ввести з клавіатури
5	Знайти косинус кута між векторами $\vec{a} = (a_1, a_2)$ та $\vec{b} = (b_1, b_2)$ за формулою $\cos\alpha = \left(\vec{a}\cdot\vec{b}\right)\Big/\left(\lvert\vec{a}\rvert\cdot\lvert\vec{b}\rvert\right)$. Модуль вектора $\lvert\vec{a}\rvert = \sqrt{a_1^2+a_2^2}$. Скалярний добуток векторів обчислити за формулою $\vec{a}\cdot\vec{b} = a_1b_1 + a_2b_2$
6	Обчислити відстань від точки M до площин $22x-4y-20z-45=0$ та $3x-y+5z+1=0$, використовуючи формулу відстані від точки до площини $\rho = \dfrac{\lvert ax_0+y_0+cz_0+d\rvert}{\sqrt{a^2+b^2+c^2}}$. Координати точки M ввести з екрана
7	Радіолокаційна станція випромінює $n = 1000$ імпульсів за 1 с з довжиною хвилі $\lambda = 0{,}03$ м. Потужність одного імпульсу $P_1 = 7\cdot10^{-4}$ Вт, а тривалість $\tau = 3\cdot10^{-7}$ Вт. Обчислити енергію одного імпульсу $W_1 = P_1\tau$; середню корисну потужність станції $P = W_1k$; кількість довжин хвиль в одному імпульсі $k = c\tau/\lambda$, якщо $c = 30$ ввести з екрана

Продовження табл. 2.3

№ вар.	Завдання
8	Обчислити корені рівняння $ax^2 + bx + c = 0,$ заданого коефіцієнтами *a*, *b*, *c* (припускаючи, що $a \neq 0$ і що корені є дійсні). Значення *a* = 2, *b* = −8, *c* = −10 ввести з екрана
9	Обчислити корінь рівняння $2x/a + b - 12 = 0$ за різних значень параметрів *a* та *b*. Значення *a*, *b* ввести з екрана
10	Обчислити гіпотенузу і площу прямокутного трикутника за двома заданими катетами. Довжини катетів ввести з екрана.
11	Тіло рухається за законом $S = t^3 - \sqrt{t}$. Обчислити швидкість тіла і відстань у момент часу *t*. Значення *t* ввести з екрана. (Функція швидкості є похідною від функції відстані)
12	Обчислити катет та площу прямокутного трикутника за заданими катетом і гіпотенузою. Довжини катета й гіпотенузи ввести з екрана
13	Обчислити $Z = (v1 + v2 + v3)/3$, де *v1*, *v2*, *v3* – об'єми куль з радіусами R_1, R_2, R_3 відповідно. Об'єм кулі обчислити за формулою $V = 4/3\pi R^3$. Значення радіусів ввести з екрана
14	У коливальному контурі ємність конденсатора $C = 10^{-6}$Ф, індуктивність котушки L = 0,04 Гн, амплітуда напруги на конденсаторі U = 100 В. Обчислити амплітуду сили струму $I = U\sqrt{C/L}$, повну енергію $W = LI^2/2$
15	Чотирикутник задано координатами власних вершин. Обчислити його периметр. Вершини ввести з екрана
16	Обчислити значення функції $W = \mathrm{sh}(x) \cdot \mathrm{tg}(x+1) - \mathrm{tg}^2(2 + \mathrm{sh}(x-1))$, де $\mathrm{sh}(x) = (e^x - e^{-x})/2$. Значення *x* ввести з екрана
17	При зміненні сили струму в котушці, індуктивність якої L = 0,5 Гн, в n = 2 рази енергія магнітного поля змінилась на ΔW = 3 Дж. Знайти початкові значення енергії $W_1 = \Delta W/(n^2 - 1)$ та сили струму $I_1 = \sqrt{2W_1/L}$
18	Обчислити периметр трикутника, заданого координатами його вершин. Координати вершин ввести з екрана
19	Задано трикутник *ABC* довжинами власних сторін *a*, *b*, *c*, які слід ввести з екрана. Обчислити його бісектриси (бісектриса, проведена до сторони *a*, дорівнює $\sqrt{bc(a+b+c)(b+c-a)}/(b+c)$)
20	Задано трикутник *ABC* довжинами власних сторін *a*, *b*, *c*. Обчислити його медіани (медіана, проведена до сторони *a*, дорівнює $0{,}5\sqrt{2b^2 + 2c^2 - a^2}$. Значення *a*, *b*, *c* ввести з екрана
21	Обчислити $Z = (R_1 + R_2 + R_3)/3$, де R_1, R_2, R_3 – радіуси куль з об'ємами V_1, V_2, V_3 відповідно. Радіус кулі обчислити за формулою $R = \sqrt[3]{3V/4\pi}$. Значення об'ємів ввести з екрана

Закінчення табл. 2.3

№ вар.	Завдання
22	Задані довжини *a, b* і *c* сторін певного трикутника. Обчислити медіани трикутника, сторонами якого є медіани вихідного трикутника. Довжина медіани, проведеної до сторони *a*, дорівнює $0{,}5\sqrt{2b^2 + 2c^2 - a^2}$
23	За якого значення напруги на конденсаторі коливального контура (в долях амплітудного значення $u/U_{макс}$) і через який час (в долях періоду t/T) енергія електричного поля буде в *n* разів відрізнюватися від енергії магнітного поля? Значення *n* ввести з екрана. $u/U_{макс} = \sqrt{n/(n+1)}$; $t/T = \arccos\sqrt{n/(n+1)}/(2\pi)$
24	Обчислити об'єм зрізаної піраміди, основами якої є квадрати зі сторонами *a* та *b*. $V = h(S_1 + \sqrt{S_1 S_2} + S_2)/3$; S_1, S_2 – площі основ, *h* – висота піраміди. Значення *a, b, h* ввести з екрана
25	Обчислити рентабельність роботи підприємства за місяць за формулою *рентабельність = прибуток / собівартість · 100 %*, якщо собівартість продукції в поточному місяці зменшилась порівняно з минулим на 2 %. Значення прибутку і собівартості за минулий місяць ввести з екрана
26	Обчислити хвильовий опір напівхвильового вібратора $p = 120(\ln(2\lambda/(\pi d)) - 0{,}577)$, $\lambda = (3 + 0{,}1\,n)$. Значення *n* та *d* ввести з екрана
27	Найти радіуси описаного *R* і вписаного *r* кіл для правильного многокутника з числом сторін *n* і довжиною сторони *a*. $R = a/(2\sin(\pi/n))$, $r = a/\mathrm{tg}(\pi/n)$. Значення *n* і *a* ввести з екрана
28	Обчислити об'єм зрізаного конуса, основи якого мають радіуси *R* та *r*. $V = h\left(S_1 + \sqrt{S_1 S_2} + S_2\right)/3$; S_1, S_2 – площі основ, *h* – висота конуса. Значення *R, r, h* ввести з екрана
29	Ввести координати точки площини (*x*, *y*). Здійснити перехід до полярних координат (ρ, ϕ), де $\rho = \sqrt{x^2 + y^2}$, $\mathrm{tg}\ \phi = y/x$
30	Тіло рухається за законом $S = t^3 - 3t^2 + 2$. Обчислити швидкість тіла в момент часу *t*. Значення *t* ввести з екрана. (Функція швидкості є похідною від функції відстані)

Самостійна робота № 2

Програмування логічних операцій

Мета роботи: вивчити основи алгебри логіки.

Теоретичні відомості

1 Основні поняття алгебри логіки

Логічною основою комп'ютера є алгебра логіки, яка розглядає логічні операції над висловлюваннями.

Алгебра логіки – це розділ математики, що вивчає висловлювання, що розглядаються з боку їх логічних значень (істинності або хибності) та логічних операцій над ними.

Логічне висловлювання має два значення: або воно істинне (`true`), або хибне (`false`). Наприклад, висловлювання (логічний вираз) «3 – просте число» є істинним завжди, а висловлювання «x+2>5» є істинним при x>3, інакше – ні.

Щоб звертатися до логічних висловлювань, їм призначають імена. Наприклад, позначимо через *A* просте висловлювання «число 6 ділиться на 2», а через *B* просте висловлювання «число 6 ділиться на 3». Тоді складене висловлювання «число 6 ділиться на 2, і число 6 ділиться на 3» можна записати як «A і B». Тут «і» – логічне зв'язування або логічна операція, *A* та *B* – логічні змінні, які можуть набувати тільки два значення «істина» або «хибність», що означають, відповідно, «1» і «0».

2. Операції відношення та логічні операції

Логічні змінні можуть набувати значень: `true` (істина) або `false` (хибність). У C++ здебільшого ці значення позначають цифрами 1 та 0 (1 – істина, 0 – хибність). Логічний тип ще називають *булевим*, від прізвища англійського математика Джорджа Буля, засновника математичної логіки. При оголошенні змінних булевого типу їх позначають як `bool`, наприклад:

```
bool m;
```

Операції відношень (<, >, <=, >=, = = (дорівнює), != (не дорівнює)) порівнюють два вирази і видають значення 1 (**`true`** – істина – “так”) або 0 (**`false`** – хибність – “ні”). Типом результату є `int` (або `bool`).

Логічна операція – дія, яка виконується над логічними змінними, її результат є 1 (`true`) або 0 (`false`). Логічні операції обчислюють кожен операнд з огляду на його еквівалентність нулю.

Базові логічні операції:

`||` – операція “АБО” (*логічне додавання*, *диз’юнкція*), результатом виконання якого є значення 1 (`true`), якщо хоча б один з операндів має ненульове

значення. Якщо перший операнд має ненульове значення, то другий операнд вже не обчислюється;

&& – операція “І (*кон’юнкція*”, *логічне множення*), результатом виконання якого є 1, якщо обидва операнди мають ненульові значення, у решті випадків результат – 0. Якщо значення першого операнда дорівнює 0, то другий операнд не обчислюється;

! – операція “НЕ” (*логічне заперечення*, *інверсія*) виконується над однією логічною змінною, результатом чого є значення `true` (1), якщо початковим значенням було `false` (0), і `false` (0) – якщо початковим значенням було `true` (1), наприклад:

```
bool w, q = true;
w = !q;          // Змінна w набуде значення false
int t, z = 0;
t = !z;          // Змінна t набуде значення 1
```

Логічні операції мають свої назви і позначення (табл. 2.4).

Таблиця 2.4

Основні логічні операції

Позначення операції	**Читається**	**Назва операції**	**Альтернативні позначення**	**Позначення логічної операції в С++**	**Позначення побітової логічної операції в С++**
¬	НЕ	Заперечення (інверсія)	Риска згори	**!**	~
∧	І	Кон'юнкція (логічне множення)	&	**&&**	&
∨	АБО	Диз'юнкція (логічне додавання)	+	**||**	|
XOR	АБО-АБО	Виключне АБО (додавання по модулю 2)	⊕	відсутній	^

3 Пріоритет логічних операцій при обчисленнях у виразах

Обчислення в логічних виразах виконуються зліва направо у відповідності з таким *пріоритетом* операцій: спочатку виконується операція “НЕ”, потім “І”, а вже потім “АБО”.

За допомогою логічних змінних і операцій будь-яке висловлювання можна формалізувати, тобто замінити логічною формулою (логічним виразом). Наприклад, $F(A,B) = A \,\&\, B \vee A$ – логічна функція двох змінних *A* і *B*.

Значення логічної функції для різних поєднань значень вхідних змінних – або, як це інакше називають, наборів вхідних змінних – зазвичай задаються спеціальною таблицею. Така таблиця називається **таблицею істинності**.

Наведемо таблицю істинності основних логічних операцій (табл. 2.5).

Таблиця 2.5

Таблиця істинності основних логічних операцій

A	*B*	$\bar{A}$ (¬A)	*A* ^ *B* (*A* & *B*)	*A* ∨ *B*	*A* ⊕ *B*
1	1	0	1	1	0
1	0	0	0	1	1
0	1	1	0	1	1
0	0	1	0	0	0

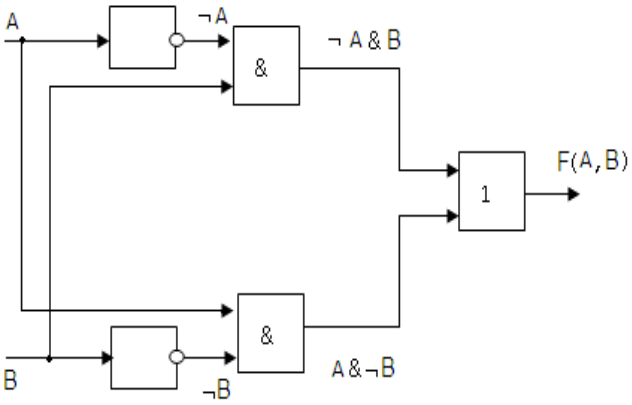
На базі цієї таблиці можна складати таблиці істинності для складніших формул.

4 Програмне опрацювання логічних виразів

Приклад. Скласти таблицю істинності для виразу $C = \neg A \wedge B \vee A \wedge \neg B$, записати його мовою C++ та скласти відповідну програму.

Таблиця істинності для виразу

A	*B*	¬*A* ^ *B*	*A* ^ ¬*B*	*C*
0	0	1 && 0 = 0	0 && 1 = 0	0 || 0 = 0
0	1	1 && 1 = 1	0 && 0 = 0	1 || 0 = 1
1	0	0 && 0 = 0	1 && 1 = 1	0 || 1 = 1
1	1	0 && 1 = 0	1 && 0 = 0	0 || 0 = 0



Вираз $C = \neg A \wedge B \vee A \wedge \neg B$ мовою C++ матиме вигляд:

```
C = (!A && B) || (A && !B);
```

Програма для реалізації цього логічного виразу може бути такою:

```
#include <iostream>
using namespace std;
int main()
{
    bool A, B, C;
    cout << "Input two boolean numbers (0 or 1):";
    cin >> A >> B;
    C = (!A && B) || (A && !B);
    cout << "C= " << C;
    system("pause");
    return 0;
}
```

Результати роботи:

```
Input two boolean numbers (0 or 1): 1  0
C= 1
```

Контрольні запитання та завдання для самоконтролю

1. Назвати усі логічні операції?
2. Як логічні операції позначаються у мові С++?
3. Вказати пріоритет виконання логічних операцій.
4. Обчислити логічний вираз `(-3 >= 5) || (7 < 9) && (0 < 3)`.
5. Вказати значення `w` після виконання оператора `bool w=2*5<=17%3;`.

Завдання до самостійної роботи

1. Скласти таблицю істинності логічного виразу, вибраного відповідно до індивідуального варіанта з табл. 2.6.
2. Записати логічний вираз індивідуального варіанта його мовою С++ та скласти відповідну програму.

Таблиця 2.6

Індивідуальні варіанти завдань

№ варіанта	Логічний вираз	№ варіанта	Логічний вираз
1	¬ (A ∧ ¬B) ∨ B	16	¬ (A ∧ B) ∨ A ∧ ¬B
2	A ∧ ¬B ∧ ¬(A ∨ B)	17	A ∧ (¬B ∨ B)
3	¬ B ∨ (A ∧ ¬B)	18	A ∨ ¬B ∧ ¬A
4	¬ A ∧ B ∨ ¬A	19	¬ (A ∨ ¬ B) ∧ B
5	¬ (A ∧ B) ∨ (B ∨ ¬A)	20	A ∧ B ∨ ¬A
6	¬B ∧ (¬A ∨ B)	21	A ∧ (¬B ∨ A)
7	A ∨ ¬(B ∨ A) ∧ A	22	¬ (A ∨ B) ∧ ¬B
8	¬B ∧ (A ∨ B)	23	(¬A ∨ ¬B) ∧ ¬A
9	¬ (A ∨ ¬B) ∨ ¬B ∧ A	24	¬ (¬A ∨ B) ∨ A
10	(A ∧ B) ∨ (¬A ∧ ¬B)	25	¬ A ∨ B ∨ (¬A∧ B)
11	A ∨ (B ∧ ¬A)	26	¬ (A ∨ B) ∧ (A ∧ ¬B)
12	B ∧ ¬A ∨ B ∧ A	27	B ∧ (A ∨ ¬B) ∨ A ∧ B
13	A ∧ (¬A ∨ B) ∧ A ∨ B	28	A ∧ ¬B ∨ (A ∨¬ B)
14	¬ A ∨ ¬B ∨ (A ∨ B)	29	¬A ∨ ¬B ∧ A
15	¬ A ∧ ¬B ∨ A	30	(¬A ∨ B) ∧ (A ∨ ¬B)

Лабораторна робота № 3
Умовний оператор if

Мета роботи: набути практичних навиків використання умовного оператора **`if`** при створюванні програмних проєктів розгалуженої структури в С++.

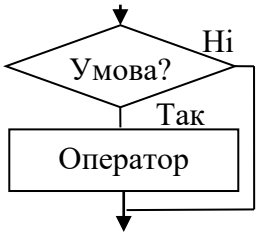
Теоретичні відомості

1. Розгалужені алгоритми

Розгалуженням називається вибір програмою тієї чи іншої низки команд залежно від того, чи виконується певна умова. При цьому спрацьовує лише одна з гілок алгоритму.

Для програмної реалізації таких обчислень слід використовувати оператори передавання керування, котрі дозволяють змінювати порядок виконування операторів програми. У мові С++ для цього передбачено інструкції: безумовного переходу – **`goto`**, умовного переходу – **`if`** та вибору варіанта – **`switch`**. Для записування умови переходу слід використовувати логічні (булеві) вирази.

2. Умовний оператор if



Оператор `if` має дві форми: скорочену та повну.

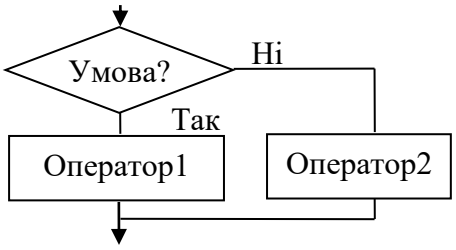
Скорочена форма має вигляд:

`if` (<*умова*>) <*оператор*>;

Якщо значення логічного виразу в *умові* є ненульове, тобто *умова* є істинна, то виконується *оператор* чи група операторів в операторних дужках {}, інакше відбудеться перехід на наступний оператор.

Повна форма цього оператора:

`if` (<*умова*>) <*оператор1*>;
`else` <*оператор2*>;



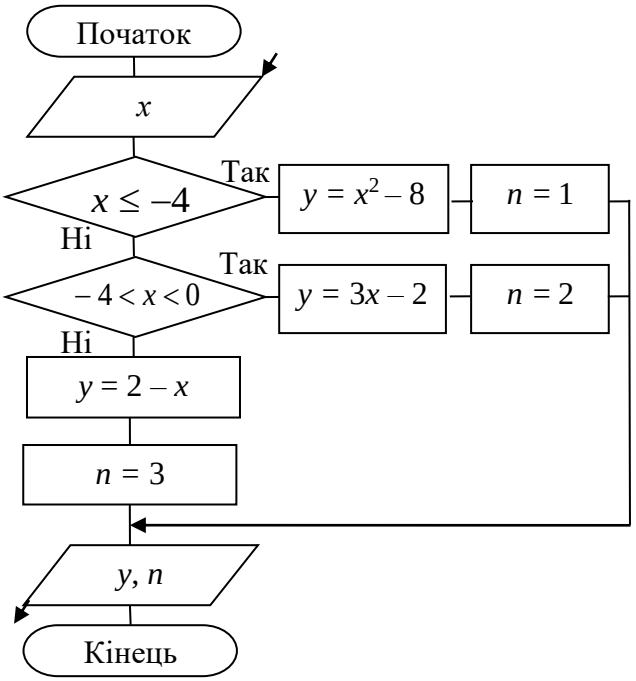
Якщо значення логічного виразу в *умові* є ненульове, тобто *умова* є істинна, то виконуватиметься *оператор1* чи група операторів в операторних дужках {}, інакше – виконуватиметься *оператор2*, після чого відбудеться перехід на наступний оператор. Зауважимо, що обчислюється лише один з операторів, а не обидва.

Наприклад, обчислення значення виразу $y = \begin{cases} 1+b^x & \text{за } x = b; \\ \dfrac{x+b}{b-x} & \text{за } x \neq b \end{cases}$ можна реалізувати чи то двома операторами `if` скороченої форми, чи одним оператором `if` повної форми:

```
1) if(x == b) y = 1 + pow(b, x);
   if(x != b) y = (x + b)/(b – x);
2) if(x == 0) y = 1 + pow(b, x); else y = (x + b)/(b – x);
```

Приклади програм з розгалуженою структурою в С++

Приклад 1. Ввести x та обчислити $y = \begin{cases} x^2 - 8 & \text{за} \quad x \le -4; \\ 3x - 2 & \text{за} \quad -4 < x < 0; \\ 2 - x & \text{за} \quad x \ge 0. \end{cases}$



Текст програми та схема алгоритму:

```
#include <iostream>
using namespace std;

int main ()
{
 setlocale(0,".1251");
 double y, x; int n;
 cout<<"Введіть значення x=";
 cin>>x;
 if(x <= –4) { y=x*x–8; n=1;}
 else
   if(–4<x && x<0){ y=3*x–2; n=2;}
   else  { y=2–x;   n=3;}
 cout<<"Результат y="<< y <<endl;
 cout<<"Виконано умову "<<n<<endl;
 system ("pause>>void");
 return 0;
}
```

Результати роботи:

```
Введіть значення x=0.4
Результат y=1.6
Виконано умову 3
```

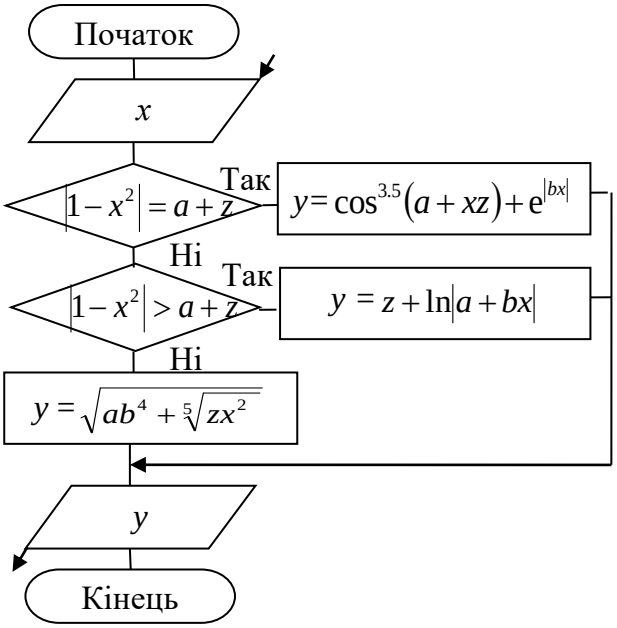
Приклад 2. Ввести x та обчислити $y = \begin{cases} \cos^{3.5}(a + xz) + e^{|bx|} & \text{за} \quad |1 - x^2| = a + z; \\ z + \ln|a + bx| & \text{за} \quad |1 - x^2| > a + z; \\ \sqrt{ab^4 + \sqrt[5]{zx^2}} & \text{за} \quad |1 - x^2| < a + z, \end{cases}$

де $a = 0.3$; $b = 1.25$; $z = (x + b)^2$.

Текст програми та схема алгоритму:

```
#include <iostream>
using namespace std;
```

```
int main ()
{ setlocale(0,".1251");
 double a=0.3, b=1.25, x, y, z;
 cout<<"Введіть значення x= ";  cin>>x;
 z = pow(x + b, 2);
 if (fabs(1-x*x) == a+z)
   y=pow(cos(a+x*z),3.5)+exp(fabs(b*x));
 else
   if (fabs(1-x*x) > a+z)
     y = z+log(fabs(a+b*x));
   else
     y = sqrt(a * pow(b,4) +
pow(z*x*x,1./5));
 cout<<"\nРезультат y = "<< y << endl;
 system ("pause>>void");
 return 0;
}
```



Результати роботи:

```
Введіть значення x = 2.45
Результат y = 1.77414
```

Приклад 3. Ввести координати точки B (змінні x та y) та визначити, чи лежить ця точка на кривій $f(x) = \begin{cases} -e^x + x^4 & \text{за } |x| > 1; \\ \arccos^2 x^4 & \text{за } |x| \le 1. \end{cases}$ Похибку задати eps = 10^{-3} (тобто $|f(x) - y| < \text{eps}$). Відповідь вивести у вигляді повідомлення.

Текст програми:

```
#include <iostream>
using namespace std;
int main ()
{ setlocale(0,".1251");
  double x, y, fx;
  const double  eps = 1e-3;
  cout<<"Введіть координати точки x та y:";
  cin >> x >> y;
  if (fabs(x) > 1)  fx = -exp(x) + pow(x, 4);
    else    fx = pow(acos(pow(x,4)),2);
  if (fabs(fx - y) < eps)  cout << "Точка лежить на f(x)" << endl;
    else  cout << "Точка не лежить на f(x)" << endl;
  system ("pause>>void");
  return 0;
}
```

Результати роботи:

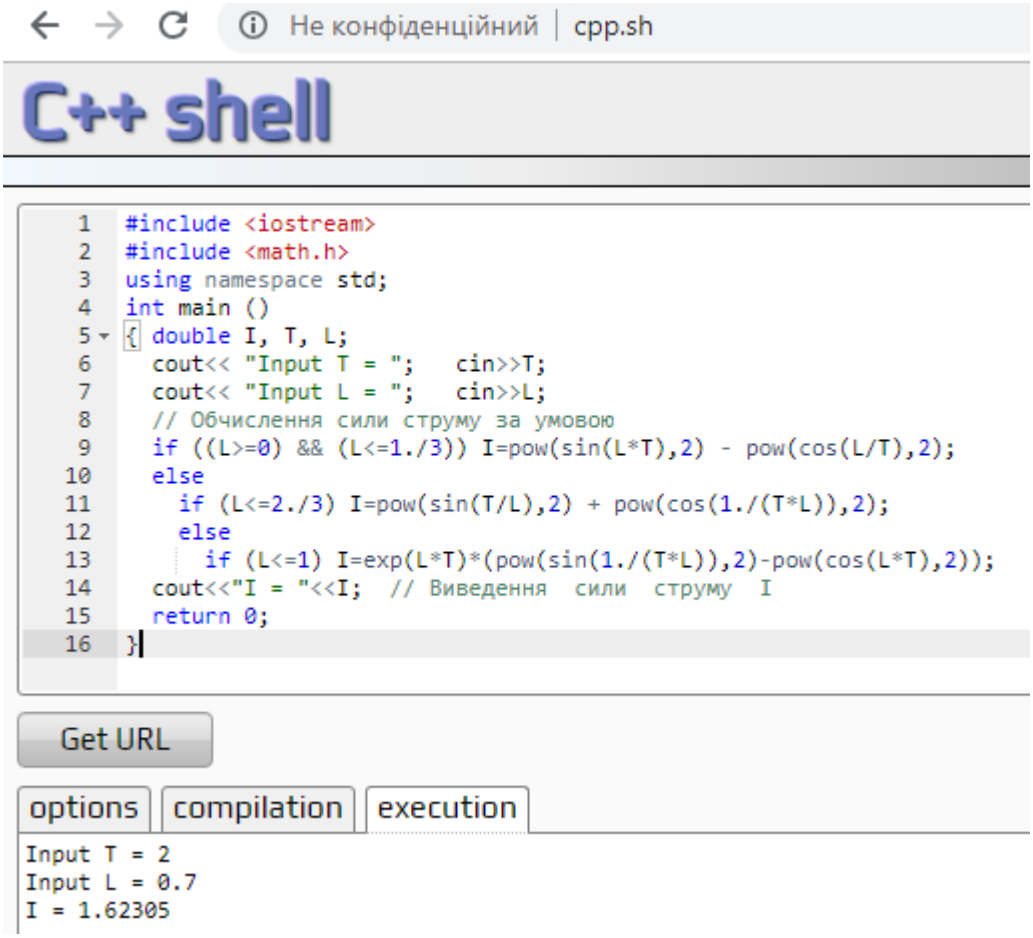
```
Введіть координати точки x та y:  -1   0
Точка лежить на f(x)
```

Приклад 4. Ввести значення періоду T і довжини L та обчислити силу струму I на ділянці електричного ланцюга довжиною L за формулою

$$I=\begin{cases}\sin^2(LT)-\cos^2\left(\dfrac{L}{T}\right) & \text{за } 0\le L\le\dfrac{1}{3};\\ \sin^2\left(\dfrac{T}{L}\right)+\cos^2\left(\dfrac{1}{TL}\right) & \text{за } \dfrac{1}{3}<L\le\dfrac{2}{3};\\ e^{LT}\left(\sin^2\left(\dfrac{1}{TL}\right)-\cos^2(LT)\right) & \text{за } \dfrac{2}{3}<L\le 1.\end{cases}$$

Текст програми та скріншот вікна онлайн компілятора http://cpp.sh:

```
#include <iostream>
#include <math.h>
using namespace std;
int main ()
{ double I, T, L;
  cout<< "Input T = ";   cin>>T;
  cout<< "Input L = ";   cin>>L;
  if ((L>=0) && (L<=1./3)) I=pow(sin(L*T),2) - pow(cos(L/T),2);
  else
    if (L<=2./3) I=pow(sin(T/L),2) + pow(cos(1./(T*L)),2);
    else
      if (L<=1) I=exp(L*T)*(pow(sin(1./(T*L)),2)-pow(cos(L*T),2));
  cout<<"I = "<<I;
  return 0;
}
```



Контрольні запитання для самоконтролю

1) Який процес називають розгалуженим?
2) Які оператори в С++ використовуються для організації розгалужень?
3) Записати умовні оператори `if` скороченої і повної форми для обчислення $y = \begin{cases} \sin x^2 & \text{за } x > 0.5; \\ \cos^2 x & \text{за } x \le 0.5. \end{cases}$
4) Назвати оператори без помилок:
   а) `if(x<=6)y=2*x; else y=cos(x);`
   б) `if y<=x then y:=exp(x*y);`
   в) `if(a<>0) if(b<>0) y=2*x;`
   г) `if(x>0)y=ln(x) else y=x;`
5) Вказати значення `x` після виконання фрагментів програми:
   а)
```
float x=1.5;
if(x<=0.5) x=7.7;
```
   б)
```
float x=1.5;
if(x<=0.5) x=7.7; else x=3;
```
6) Навести значення змінної `z` після виконання операторів:
```
float z, x=2.5;
if(x>=0.5) z=7.7; else z=5.5;
```

Лабораторне завдання

1) У протоколі лабораторної роботи дати відповіді на контрольні запитання.

2) У протоколі лабораторної роботи скласти схеми алгоритмів і написати програми мовою С++ із застосуванням умовного оператора `if` для розв'язання завдань, поданих у табл. 3.1 … 3.3 відповідно до індивідуального варіанта.

3) Створити на комп'ютері програмні проєкти мовою С++ для реалізації написаних програм. Занести результати обчислень до протоколу.

Таблиця 3.1

Індивідуальні завдання базового рівня складності

№ вар.	Функція	№ вар.	Функція
1	$Y = \begin{cases} x^2+1 & \text{за } x < 0; \\ x^2-1 & \text{за } 0 \le x < 2; \\ x & \text{за } x \ge 2 \end{cases}$	2	$Y = \begin{cases} 2x+2 & \text{за } x < -3; \\ 2x-2 & \text{за } -3 \le x \le 0; \\ x^2 & \text{за } x > 0 \end{cases}$
3	$Y = \begin{cases} 6x+8 & \text{за } x \le -5; \\ x-2 & \text{за } -5 < x \le 3; \\ 2x^2 & \text{за } x > 3 \end{cases}$	4	$Y = \begin{cases} 2x-1 & \text{за } x \le -4; \\ x^2+2 & \text{за } -4 < x \le 5; \\ x+3 & \text{за } x > 5 \end{cases}$
5	$Y = \begin{cases} 6x^3-8 & \text{за } x \le -8; \\ x^3-8 & \text{за } -8 < x < 0; \\ 2x^2 & \text{за } x \ge 0 \end{cases}$	6	$Y = \begin{cases} 2x^3+3x & \text{за } x \le -1; \\ x^2-4 & \text{за } -1 < x < 0; \\ x^3 & \text{за } x \ge 0 \end{cases}$

Продовження табл. 3.1

№ вар.	Функція	№ вар.	Функція
7	$Y=\begin{cases}4x^2+2x & \text{за } x\le -12;\\ 2x^2+2x & \text{за } -12<x<3;\\ x+1 & \text{за } x\ge 3\end{cases}$	8	$Y=\begin{cases}x^3-1 & \text{за } x\le -4;\\ 2x-1 & \text{за } -4<x\le 3;\\ 3x^3 & \text{за } x>3\end{cases}$
9	$Y=\begin{cases}4x+3 & \text{за } x\le -2;\\ 2x^2-4 & \text{за } -2<x<4;\\ x^2-2 & \text{за } x\ge 4\end{cases}$	10	$Y=\begin{cases}2x+4 & \text{за } x\le -1;\\ x-4 & \text{за } -1<x<0;\\ x^3+4 & \text{за } x\ge 0\end{cases}$
11	$Y=\begin{cases}4x^2+2x & \text{за } x<-2;\\ 2x-1 & \text{за } -2\le x<3;\\ x^3+3 & \text{за } x\ge 3\end{cases}$	12	$Y=\begin{cases}3x^2+2x & \text{за } x<-3;\\ 2x+1 & \text{за } -3\le x<8;\\ 3x & \text{за } x\ge 8\end{cases}$
13	$Y=\begin{cases}4x+2x & \text{за } x\le -4;\\ x-2x & \text{за } -4<x<2;\\ x+2 & \text{за } x\ge 2\end{cases}$	14	$Y=\begin{cases}27x+3 & \text{за } x\le -6\\ x^3-1 & \text{за } -6<x<3\\ x^2+1 & \text{за } x\ge 3\end{cases}$
15	$Y=\begin{cases}x^3+2x^2 & \text{за } x\le -2;\\ x^2-1 & \text{за } -2<x<3;\\ 2x+2 & \text{за } x\ge 3\end{cases}$	16	$Y=\begin{cases}4x^3+2x & \text{за } x<-4;\\ 2x-5 & \text{за } -4\le x<4;\\ x-3 & \text{за } x\ge 4\end{cases}$
17	$Y=\begin{cases}6x^2+2x & \text{за } x\le -6;\\ 2x-6 & \text{за } -6<x<4;\\ 6x+1 & \text{за } x\ge 4\end{cases}$	18	$Y=\begin{cases}27x^2+1 & \text{за } x\le -3;\\ x-2 & \text{за } -3<x<5;\\ 3x+1 & \text{за } x\ge 5\end{cases}$
19	$Y=\begin{cases}8x^3+2 & \text{за } x\le -1;\\ x^2-1 & \text{за } -1<x<1;\\ x+1 & \text{за } x\ge 1\end{cases}$	20	$Y=\begin{cases}21-x & \text{за } x\le -7;\\ x^2+3 & \text{за } -7<x<4;\\ x^2-3 & \text{за } x\ge 4\end{cases}$
21	$Y=\begin{cases}2x^2+3 & \text{за } x<-2;\\ x^3-6 & \text{за } -2\le x<0;\\ 2(x+1) & \text{за } x\ge 0\end{cases}$	22	$Y=\begin{cases}4x^3+4 & \text{за } x\le -2;\\ 3x-3 & \text{за } -2<x\le 3;\\ 2x_2+2 & \text{за } x>3\end{cases}$
23	$Y=\begin{cases}x^3+2x & \text{за } x\le -3;\\ 2x-1 & \text{за } -3<x\le 8;\\ x^2+1 & \text{за } x>8\end{cases}$	24	$Y=\begin{cases}25x+1 & \text{за } x\le -2;\\ x^3-25 & \text{за } -2<x<4;\\ 24x+x^2 & \text{за } x\ge 4\end{cases}$
25	$Y=\begin{cases}26x+4 & \text{за } x\le -6;\\ 4x^2+2 & \text{за } -6<x<6;\\ 2x-3 & \text{за } x\ge 6\end{cases}$	26	$Y=\begin{cases}9x^3+1 & \text{за } x\le -9;\\ x^2-1 & \text{за } -9<x\le 1;\\ x+2 & \text{за } x>1\end{cases}$

Закінчення табл. 3.1

№ вар.	Функція	№ вар.	Функція
27	$Y=\begin{cases}4x^2+4 & \text{за } x\le -4;\\ x^3-1 & \text{за } -4<x<4;\\ x^2+1 & \text{за } x\ge 4\end{cases}$	28	$Y=\begin{cases}x^3-29 & \text{за } x\le -3;\\ 2x+3 & \text{за } -3<x\le 6;\\ x^2+1 & \text{за } x>6\end{cases}$
29	$Y=\begin{cases}3x+1 & \text{за } x\le -3;\\ x^2-1 & \text{за } -3<x<4;\\ x^3+1 & \text{за } x\ge 4\end{cases}$	30	$Y=\begin{cases}2x^3+4x & \text{за } x\le -1;\\ x+4 & \text{за } -1<x<3;\\ 2x+2 & \text{за } x\ge 3\end{cases}$

Таблиця 3.2

Індивідуальні завдання середнього рівня складності

№ вар.	Функція	№ вар.	Функція
1	$y=\begin{cases}a^3+\arcsin\left(\cos^3 bx\right) & \text{за } x\le a;\\ \sqrt{(a+bx)-2}+\sin x & \text{за } a<x<b;\\ \operatorname{tg}^2(a+bx+z) & \text{за } x\ge b,\end{cases}$ де $a=2.5;\ b=3.5;\ z=\sin(bx)$	2	$y=\begin{cases}a^{2b}x^2+\sqrt{b^4+2.7} & \text{за } \lvert x\rvert<0.7;\\ \operatorname{arctg}\left(3^x-px\right) & \text{за } \lvert x\rvert=0.7;\\ \sqrt[3]{\ln\lvert a-px\rvert+4.3} & \text{за } \lvert x\rvert>0.7,\end{cases}$ де $a=0.54;\ b=0.34;\ p=ax+b$
3	$y=\begin{cases}(a+z)\cos^2\left(bx+x^3\right) & \text{за } x<a;\\ a\ln(zx)+\sin^2\left(b^2\right) & \text{за } a\le x\le b;\\ \sqrt[3]{0.3b+\sqrt{\lvert(a-z^2)\rvert}} & \text{за } x>b,\end{cases}$ де $a=0.1;\ b=3.25;\ z=\cos^2(x)$	4	$y=\begin{cases}\operatorname{ctg}\left(x^2\mathrm{e}^{3k}\right)+\ln\lvert rx\rvert & \text{за } x=rs;\\ \sqrt[5]{x^2}+\sqrt{\lvert\sin k\rvert} & \text{за } x>rs;\\ \operatorname{tg}\left(kx+\operatorname{tg}(rs)\right) & \text{за } x<rs,\end{cases}$ де $r=2.4;\ s=5;\ k=0.5$
5	$y=\begin{cases}\dfrac{(2a+1)^2}{3.71-x^2} & \text{за } z>-0.5;\\ \sin^3\sqrt{bz}-ax & \text{за } -0.5\le z\le 10^{-3};\\ \dfrac{\operatorname{tg}(z+x)-\mathrm{e}^x}{3.5abx} & \text{за } z>10^{-3},\end{cases}$ де $a=0.3;\ b=0.7;\ z=\cos(x+2)$	6	$y=\begin{cases}\mathrm{e}^{ax}+f\cos^5 bx & \text{за } x\le a;\\ \cos^2\sqrt{bx}-\ln(fx) & \text{за } a<x\le b^2;\\ \cos^2(a+bfx) & \text{за } x>b^2,\end{cases}$ де $a=1.5;\ b=1.44;\ f=\sqrt{b}x$
7	$y=\begin{cases}a\cos^2 x+b\sin^2 zx & \text{за } x\le a;\\ a\cdot\operatorname{tg}\left(\sin^2 bx+z\right) & \text{за } a<x\le 4.5b;\\ \ln(ax-b)+z^2 & \text{за } x>4.5b,\end{cases}$ де $a=1.5;\ b=0.7;\ z=\operatorname{tg}\lvert\operatorname{tg}(bx)\rvert$	8	$y=\begin{cases}\ln\lvert bzx\rvert+za^{2.5} & \text{за } a^3<x\le b;\\ ax^2+bz^a+\sin^2 zx & \text{за } x\le a^3;\\ \cos(ax+b)+\ln\lvert zx\rvert & \text{за } x>b,\end{cases}$ де $a=0.5;\ b=0.7;\ z=0.2$

Продовження табл. 3.2

№ вар.	Функція	№ вар.	Функція
9	$y=\begin{cases}\sin\left(e^{a+b}\right)+x^2 & \text{за } a+b>x;\\ \operatorname{arctg}(abc)+\sqrt[3]{x} & \text{за } a+b=x;\\ \arcsin\left(\cos^2\left(\sqrt{\lvert x\rvert}\right)\right) & \text{за } a+b<x,\end{cases}$ де $a=0.5;\ b=1.5;\ c=3.2$	10	$y=\begin{cases}\ln\left(\lvert\lg\lvert kx+mn\rvert\rvert\right) & \text{за } x>\lvert m+n\rvert;\\ \sin(kmx)+\sqrt{\lvert nx\rvert} & \text{за } x=\lvert m+n\rvert;\\ e^{\cos x}+e^{m+n} & \text{за } x<\lvert m+n\rvert,\end{cases}$ де $m=2.1;\ n=1.9;\ k=8.5$
11	$y=\begin{cases}a\sin^2 x+\cos(zx) & \text{за } x<\ln(a);\\ \cos^3(a+zx) & \text{за } \ln(a)\le x\le b;\\ \sqrt{2.5a^3+\left(b-zx^2\right)^6} & \text{за } x>b,\end{cases}$ де $a=0.1;\ b=3.25;\ z=\cos^2(x)$	12	$y=\begin{cases}\sin(bm+\cos(nx)) & \text{за } \lvert bm\rvert>x^2;\\ \cos(bm-\sin x) & \text{за } \lvert bm\rvert<x^2;\\ \sqrt{e^{\lvert\cos x\rvert}+\sqrt{\lvert bmx\rvert}} & \text{за } \lvert bm\rvert=x^2,\end{cases}$ де $m=0.5;\ b=-2;\ n=0.2$
13	$y=\begin{cases}xe^x+\left(z+7.7abx\right) & \text{за } x<a;\\ \operatorname{tg}(ax+z)+\cos^2 bx & \text{за } a\le x\le b^2;\\ \ln\left(\sin^2\left(a+bx+zx^2\right)\right) & \text{за } x>b^2.\end{cases}$ де $a=1.5;\ b=-1.7;\ z=1.2$	14	$y=\begin{cases}c\sin^2\left(b^2x\right)+\ln(cx+a) & \text{за } x<a;\\ a+\ln(bx)-\sin(cx) & \text{за } a\le x<b;\\ \sqrt{\lvert\cos(a+bx)+cx^2\rvert} & \text{за } x\ge b,\end{cases}$ де $a=0.5;\ b=0.7;\ c=3.4$
15	$y=\begin{cases}a\sin^2 x+b\cos(zx) & \text{за } x<a^3;\\ (a+bx)^2-\sin(zx) & \text{за } a^3\le x\le b;\\ \sqrt{\lvert x-\sin(bx+z)\rvert} & \text{за } x>b,\end{cases}$ де $a=0.9;\ b=1.25;\ z=x^2$	16	$y=\begin{cases}2.8\sin^2 ax-bx^3z & \text{за } x<a;\\ z\cos(ax+b)^2 & \text{за } a\le x\le b^2;\\ e^{\lvert 2,5ax\rvert}+zabx & \text{за } x>b^2,\end{cases}$ де $a=0.7;\ b=-1.25;\ z=3.5$
17	$y=\begin{cases}a+\sin bx+\cos x^2 & \text{за } x\le a;\\ \sqrt{a+bx}+\sin zx & \text{за } a<x<\ln b;\\ \ln(a+bx+z) & \text{за } x\ge\ln b,\end{cases}$ де $a=0.2;\ b=12.5;\ z=(2.5+b)^2$	18	$y=\begin{cases}\sin\left(e^{a+b}\right)+x^2 & \text{за } e^{a+b}>e^x;\\ \operatorname{arctg}(abz)+\sqrt[3]{x} & \text{за } e^{a+b}=e^x;\\ \cos\left(\sqrt{\lvert x+abz\rvert}\right) & \text{за } e^{a+b}<e^x,\end{cases}$ де $a=0.7;\ b=2.3;\ z=(x+b)^2$
19	$y=\begin{cases}\ln^2(0.5a-u) & \text{за } u<-0.5;\\ \dfrac{2x-a}{7\pi+x+u} & \text{за } -0.5\le u\le 10^{-3};\\ \lg(u+x)-e^x & \text{за } u>10^{-3},\end{cases}$ де $a=2.3;\ u=\sin(x+a)$	20	$y=\begin{cases}xe^a+e^{\lvert bc\rvert} & \text{за } \lvert 1-x^2\rvert=a+c;\\ \sin^2 ax+\cos bc & \text{за } \lvert 1-x^2\rvert>a+c;\\ \sqrt{ab^4+\sqrt[5]{cx^2}} & \text{за } \lvert 1-x^2\rvert<a+c,\end{cases}$ де $a=0.3;\ b=0.7;\ c=2.7-x$
21	$y=\begin{cases}x^2e^{2k}+\ln\lvert rx\rvert & \text{за } \cos x=\cos(rs);\\ \sqrt[3]{x^2}+\sqrt{\lvert k+rsx\rvert} & \text{за } \cos x>\cos(rs);\\ \operatorname{arctg}(kx+rs) & \text{за } \cos x<\cos(rs),\end{cases}$ де $s=0.15;\ r=10;\ k=0.7$	22	$y=\begin{cases}\sqrt[3]{b+\sqrt{\lvert x+c\rvert}} & \text{за } \lg a<x;\\ \cos(x-b-c) & \text{за } \lg a=x;\\ \sin(x+a-b) & \text{за } \lg a>x,\end{cases}$ де $a=10;\ b=0.7;\ c=3.7$
23	$y=\begin{cases}a+bx+\sin^2 zx^{3.5} & \text{за } x<a;\\ a+\ln\lvert ab-zx\rvert & \text{за } a\le x\le b;\\ \sqrt{\lvert a+\operatorname{ctg}(zx)\rvert}+bx & \text{за } x>b,\end{cases}$ де $a=0.2;\ b=0.9;\ z=1.7$	24	$y=\begin{cases}z^2-\cos^2(x) & \text{за } x<3.5a;\\ (a-x)^2-bx & \text{за } 3.5a\le x\le b;\\ \sqrt{\sqrt{bx}+zx^2} & \text{за } x>b,\end{cases}$ де $a=0.1;\ b=2.1;\ z=abx$

Закінчення табл. 3.2

№ вар.	Функція	№ вар.	Функція
25	$y=\begin{cases}3.5\sin^2(bx+z)^3 & \text{за } x\le a;\\ \ln(a+b^3x)+a & \text{за } a<x\le b^{2.5};\\ \cos^2(a^b+xz) & \text{за } x>b^{2.5},\end{cases}$ де $a=0.3;\ b=1.7;\ z=2$	26	$y=\begin{cases}(3.5a-7.3bx)^3 & \text{за } x<-\ln\lvert a\rvert;\\ a^b-\cos^3(zx) & \text{за } -\ln\lvert a\rvert\le x<b;\\ \sqrt{\lvert\operatorname{tg}a-x\rvert}-x & \text{за } x\ge b,\end{cases}$ де $a=0.3;\ b=3.7;\ z=12.7/x$
27	$y=\begin{cases}\ln\lvert mx+n\rvert & \text{за } x^2>m+n;\\ e^{\cos\lvert mx-n\rvert} & \text{за } x^2=m+n;\\ \sqrt[3]{k^2+\cos^2 x} & \text{за } x^2<m+n,\end{cases}$ де $m=2.1;\ n=1.9;\ k=3.5$	28	$y=\begin{cases}2.5b^2+ax-\cos xz & \text{за } x\le 5a;\\ (a^2-x)^3+\ln(xz) & \text{за } x>b;\\ \sqrt{b^2+(a-x^3z)^2} & \text{за } 5a<x\le b,\end{cases}$ де $a=0.3;\ b=1.7;\ z=0.7$
29	$y=\begin{cases}\sqrt{\lvert a-\cos^2 b^3x+c^2\rvert} & \text{за } \lvert 1-x^2\rvert=a+c;\\ e^{0.04x}+\ln\lvert b^5\cos x\rvert & \text{за } \lvert 1-x^2\rvert>a+c;\\ \cos^2(b^3x)+\ln\lvert bx-a\rvert & \text{за } \lvert 1-x^2\rvert<a+c,\end{cases}$ де $a=0.7;\ b=1.25;\ c=1.3$	30	$y=\begin{cases}e^{ax}-3.5\cos^2(z+bx) & \text{за } x\le a;\\ a+\ln\lvert a+bx\rvert-2x & \text{за } a<x\le b;\\ a+\cos^{3.5}(a+bxz) & \text{за } x>b,\end{cases}$ де $a=0.7;\ b=1.25;\ z=(x+b)^2$

Таблиця 3.3

Індивідуальні завдання високого рівня складності

№ вар.	Завдання
1	Ввести два числа і визначити, що більше: сума квадратів чи квадрат суми цих чисел. Відповідь вивести у вигляді повідомлення
2	Ввести значення кута в радіанах і визначити, що більше: значення синуса чи косинуса цього кута. Відповідь вивести у вигляді повідомлення
3	Ввести три числа і визначити серед них середнє за значенням
4	Ввести три числа і визначити серед них найменше
5	Ввести координати точки $B(x$ та $y)$ і визначити: чи належить ця точка кривій $f(x) = 6x^7–4.5x^5+4x^2$ з допустимою похибкою eps = $10^{–3}$ (тобто $\lvert f(x) – y\rvert <$ eps)
6	Ввести координати точок $A(x_0, y_0)$ і $B(x_1, y_1)$ і визначити яка з цих точок – A чи B – є найбільш віддалена від початку координат (O(0,0)). Відповідь вивести у вигляді повідомлення
7	Ввести значення трьох сторін трикутника a, b та c і визначити, чи є цей трикутник прямокутним. Відповідь вивести у вигляді повідомлення
8	Ввести три числа і додатні з них піднести до квадрата, а від'ємні залишити без змін
9	Ввести координати точки $A(x$ та $y)$ і визначити: в якій чверті лежить ця точка. Відповідь вивести у вигляді повідомлення
10	Ввести координати точки x і y та визначити, чи лежить ця точка всередині кола з радіусом R. Центром кола є початок координат. Відповідь вивести у вигляді повідомлення

Закінчення табл. 3.3

№№	Завдання
11	Ввести три цілих числа (довжини сторін трикутника) і визначити, чи можна побудувати за цими числами трикутник
12	Ввести значення сторони квадрата A та радіус кола R і визначити, площа якої з цих фігур є більше. Відповідь вивести у вигляді повідомлення
13	Ввести значення трьох сторін двох трикутників – $a1,b1,c1$ і $a2,b2,c2$. Визначити, площа якого з них є більше. Відповідь вивести у вигляді повідомлення
14	Ввести координати точки B (x та y) і визначити: чи належить ця точка кривій $f(x) = 6\cos^2 x - 0.25x^5 + 3.2x^2 - 2.7$ з припустимою похибкою $\text{eps} = 10^{-3}$ (тобто $\lvert f(x) - y\rvert < \text{eps}$). Відповідь вивести у вигляді повідомлення
15	Ввести три числа і додатні з них піднести до куба, а від'ємні – замінити на 0
16	Ввести значення трьох сторін трикутника a, b, і c. Найменша зі сторін трикутника є стороною квадрата. Визначити, площа якої з цих фігур є більша
17	Ввести координати точки A(x та y) і визначити, чи належить ця точка до першої чверті. Відповідь вивести у вигляді повідомлення
18	Ввести три числа і вивести числа за модулем, більші за середнє арифметичне
19	Ввести радіанну міру кута і визначити більше зі значень тангенса або котангенса цього кута. Відповідь вивести у вигляді повідомлення
20	Ввести координати точки Q(x та y) і визначити, чи лежить ця точка на кривій $f(x) = 7\text{tg}^2 x - 0.31x^3 + 3.2x^2 - e^x$ з припустимою похибкою $\text{eps} = 10^{-3}$ (тобто $\lvert f(x) - y\rvert < \text{eps}$). Відповідь вивести у вигляді повідомлення
21	Ввести три числа і визначити найбільше з них
22	Ввести два числа і визначити, що є більше: різниця квадратів чи модуль квадрата різниці цих чисел. Відповідь вивести у вигляді повідомлення
23	Ввести координати точок $A(x_0, y_0)$ та $B(x_1, y_1)$ і визначити, яка з точок – A чи B – найменш віддалена від початку координат $O(0,0)$. Відповідь вивести у вигляді повідомлення
24	Ввести координати точки A(x та y) і визначити, чи лежить ця точка всередині тора, утвореного колами із радіусами r і R із центром у точці $O(0,0)$
25	Ввести координати точки B(x та y) і визначити, чи лежить ця точка на кривій $f(x) = \begin{cases} \sin^2 x^3 & \text{за} \quad \lvert x\rvert > 1; \\ \sqrt{6\arcsin x^7 + 4.5x^6 + 4x^2 + 2} & \text{за} \quad \lvert x\rvert \le 1 \end{cases}$ із припустимою похибкою $\text{eps} = 10^{-3}$ (тобто $\lvert f(x) - y\rvert < \text{eps}$). Відповідь вивести у вигляді повідомлення
26	Ввести координати точки A(x та y) і визначити, чи лежить ця точка в четвертій чверті. Відповідь вивести у вигляді повідомлення
27	Ввести значення трьох сторін трикутника a, b та c і визначити, чи є цей трикутник рівнобедреним. Відповідь вивести у вигляді повідомлення
28	Ввести три цілих числа a, b, c і визначити, чи є вони трійкою Піфагора ($c^2 = a^2 + b^2$). Відповідь вивести у вигляді повідомлення
29	Ввести координати точки A(x та y) і визначити, чи лежить ця точка в області, обмеженій параболою $y = 2 - x^2$ та віссю абсцис
30	Ввести координати точок $A_1(x_1, y_1)$, $A_2(x_2, y_2)$, $A_3(x_3, y_3)$ і визначити, чи лежать ці точки на одній прямій. Відповідь вивести у вигляді повідомлення

Самостійна робота № 3

Побітові логічні операції

Мета роботи: освоїти навички побітової роботи над числами.

Теоретичні відомості

1 Логічні побітові операції

Логічні побітові операції виконуються над цілими числами і опрацьовують число побітово. До цих операцій належать: побітове логічне заперечення (~), побітове логічне множення "І" (&), побітове логічне додавання "АБО" (|), побітове логічне "виключне АБО" (^) (табл. 3.4). Операнди логічних операцій мають бути будь-якого цілого типу.

Таблиця 3.4

Логічні побітові операції

Позначення	Операція	Приклади	
~	інверсія, НЕ	~X	~1011 = 0100
&	І	X & Y	1011 & 1010 = 1010
|	АБО	X | Y	1011 | 1010 = 1011
^	виключне АБО	X ^ Y	1011 ^ 1010 = 0001
<<	зсув ліворуч	X << 2	1011 << 2 = 1100
>>	зсув праворуч	Y >> 2	1011 >> 2 = 0010

Зауважимо, що в наведених у табл. 3.4 прикладах для простоти та наочності ми обмежились лише чотирма розрядами двійкових чисел.

Операція побітового логічного заперечення "НЕ" (~) інвертує кожен біт операнда: 0 – на 1, 1 – на 0, тобто продукує двійкове доповнення свого операнда ($\sim 1010_2 = 0101_2$), наприклад:

```
short  x = 987;
unsigned short  y = 0xAAAA;
y = ~y;           // У  результаті  y = 0x5555 (шістнадцяткове)
if (!(x < y));    // Результат виразу  0 (false)
```

Операція побітового логічного множення "І" (&) порівнює кожен біт першого операнда з відповідним бітом другого операнда і, якщо обидва порівнюваних біти є одиницями, то відповідний біт результату встановлюється в 1, інакше – в 0. Наприклад, $1010_2 \ \& \ 1001_2 = 1000_2$.

Операція побітового логічного додавання "АБО" (|) порівнює кожен біт першого операнда з відповідним бітом другого операнда, і, якщо хоча б один (чи обидва) з них дорівнює 1, то відповідний біт результату встановлюється в 1, інакше – біт результату дорівнює 0. Наприклад, $1010_2 \mid 1001_2 = 1011_2$.

Операція побітове логічне "виключне АБО" (^) порівнює кожен біт першого операнда з відповідним бітом другого операнда і, якщо обидва опрацьовувані біти мають однакове значення, результат дорівнює 0, у решті випадків – 1. Наприклад, 1010_2 ^ $1001_2 = 0011_2$.

Наведемо приклади виконання побітових логічних операцій.

```
short i=0x45FF, //  i= 0100 0101 1111 1111
r,  j=0x00FF;   //  j= 0000 0000 1111 1111
r = i ^ j;      //  r=0x4500 = 0100 0101 0000 0000
r = i | j;      //  r=0x45FF = 0100 0101 0000 0000
r = i & j       //  r=0x00FF = 0000 0000 1111 1111
short n,  i = 0xAB00,  j = 0xABCD;
n = i & j;      // У результаті  n = AB00 (шістнадцяткове)
n = i | j;      // У результаті  n = ABCD (шістнадцяткове)
n = i ^ j;      // У результаті  n = CD (шістнадцяткове)
```

Операції побітового зсуву >> і << виконують зсув бітів лівого операнда на кількість розрядів, зазначену правим операндом, відповідно праворуч чи ліворуч. Значення бітів, яких бракує, доповнюються нулями. При виконанні операції зсуву можуть втрачатися старші чи молодші розряди:

>> – побітовий зсув праворуч на задану кількість бітів (65 >> 2 = 16, оскільки число $65 = 41_{16} = 0100\ 0001_2$ після зсуву праворуч становить $0001\ 0000_2 = 10_{16} = 16$). Операція є еквівалентна до цілочислового поділу на 2 у відповідному степені (65 / 4 = 16);

<< – побітовий зсув ліворуч (13 << 2 = 20, оскільки число $13 = D_{16} = 1101_2$ після зсуву ліворуч становить $11\ 0100_2 = 34_{16} = 52$). Операція є еквівалентна до цілочислового множення на 2 у відповідному степені (13 * 4 = 52). Слід пам'ятати, що при цьому є можливе втрачання старших розрядів.

Наведемо кілька прикладів роботи операцій зсуву.

```
int  c = 12, m, n;
m = c >> 2;     // m=6
n = c << 2;     // n=24
c = n >> 3      // c=24/23=24/8=3
int  i = 0x1234, j, k ;
k = i << 4;     // k=0x0234
j = i << 8;     // j=0x3400
i = j >> 8;     // i=0x0034
unsigned  int x = 0x00AA, y = 0x5500, z;
z = (x << 8) + (y >> 8);
```

В останньому прикладі `x` зсувається ліворуч на 8 позицій, а `y` – праворуч на 8 позицій. Результати зсувів додаються, утворюючи величину `0xAA55`, котра присвоюється `z`.

Застосовування операцій `>>` та `<<` почергово до однієї й тієї самої змінної може спричинити змінення її значення через втрату розрядів.

Перетворювання, виконувані операціями зсуву, не забезпечують опрацювання ситуацій переповнення і втрати знаків. Знак першого операнда після ви-

конання операції зберігається. Результат операції зсуву є невизначений, якщо другий операнд буде від'ємний.

Побітові операції є зручні для організації зберігання у стислому вигляді інформації про стан on/off (ввімкнено/вимкнено). В одному байті можна зберігати 8 таких "прапорців". Якщо змінна `ch` є сховищем таких "прапорців", то перевірити, чи є "прапорець" третього біту у стані on, можна у такий спосіб:

```
if (ch & 4) . . .
```

Ця перевірка базується на двійковому поданні числа 4 = 0000 0100.

Зауважимо, що вираз `a ^ a` завжди повертає значення 0, а вираз `a ^ b ^ a` – значення `b`. Ці закономірності часто застосовуються у растровій графіці.

2 Пріоритет логічних операцій

Обчислення в логічних виразах виконуються зліва направо у відповідності з таким *пріоритетом* операцій:

1) `~`, `!`;
2) `>>`, `<<`;
3) `<`, `<=`, `>`, `>=`;
4) `==`, `!=`;
5) `&` (побітове "І");
6) `^` (побітове "виключне АБО");
7) `|` (побітове "АБО");
8) `&&` ("І");
9) `||` ("АБО").

Вирази у круглих дужках виконуються першочергово.

Приклад програмного побітового опрацювання чисел

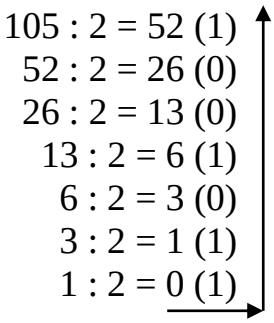
Приклад 1. Для числа 105 виконати перетворення у двійкову і шістнадцяткову системи числення, після чого виконати над ним такі побітові логічні операції:

1) інвертувати усі біти числа;
2) встановити (задати значення 1) для 5-ого біта числа;
3) визначити (вибрати окремо) значення 4-го біта числа;
4) скинути (встановити в 0) 0-й біт числа;
5) інвертувати 1-й біт числа;
6) логічний побітовий зсув на 2 біти праворуч.

Для усіх здобутих двійкових чисел виконати переведення до шіснадцяткової та десяткової систем числення.

Розробити програмний код для виконання усіх зазначених побітових операцій та перевірити правильність виконання виконаних обчислень.

Розв'язок: Переведемо число 105 до двійкової системи числення:



Результат: $105_{10} = 1101001_2$.

Переведення числа 105 до шістнадцяткової системи числення має вигляд:

105 : 16 = 6 (9)

Результат: $105_{10} = 69_{16}$.

Виконання побітових логічних операцій:

1) інвертувати усі біти числа

0110 1001 $=1001\ 0110_2 = 1\cdot2^7+1\cdot2^4+0\cdot2^3+1\cdot2^2+1\cdot2^1+0\cdot2^0=128+16+4+2=150$;

Проте, якщо число займає не один, а більше байтів, наприклад, два або чотири, то інвертуватимуться і старші біти числа з 0 на 1. Для двобайтного числа 105 (0000 0000 1001 0110) інвертованим буде 1111 1111 0110 1001 = 65 430. Це є логічним, оскільки 65 535 – 105 = 65 430;

2) визначити (вибрати окремо) значення 3-го біта числа:

$110\ \underline{1}001_2 \rightarrow 1$;

3) встановити (задати значення 1) для 4-го біта числа:

$11\underline{0}\ 1001_2 = 111\ 1001 = 2^6+2^5+2^4+2^3+2^0 = 64+32+16+8+1 = 121$;

4) скинути (встановити в 0) 0-й біт числа:

$110\ 100\underline{1}_2 = 110\ 1000_2 = 2^6+2^5+2^3 = 64+32+8 = 104$;

5) інвертувати 2-й біт числа:

$110\ 1\underline{0}01_2 = 110\ 1101_2 = 1\cdot2^6+1\cdot2^5+1\cdot2^3+1\cdot2^2+1\cdot2^0 = 64+32+8+4+1 = 109$;

6) логічний побітовий зсув на 2 біти праворуч:

$1101001_2 >>\ 2 = 11010_2 = 2^4+2^3+2^1 = 16+8+2 = 26$ $(105/2^2=105/4=26)$.

Програмний код для виконання усіх зазначених побітових операцій:

```
#include <iostream>
using namespace std;

void bin (int dec) // функція для переведення числа у двійкову систему числення
{
 int v, i, kol;  // Вага і номер формованого розряду, а також кількість розрядів
 if (dec > 255)
   { v = 65536; kol=16; } // kol - кількість розрядів
 else
   { v=256; kol=8; }      // v - вага старшого розряду
```

```
 for (i = 1; i <= kol; i++)
 {
    v = v / 2;            // Вага наступного розряду удвічі менше
    if (i % 4 == 1) printf(" ");
    if (dec >= v)
     { cout<<"1";
       dec -= v;
     }
    else printf("0");
  }
}

int main()
{
  setlocale(0,".1251");
  unsigned short x, xinv, x3, x4_1, x0_0, xinv_2, x_r2;
  cout << "Input number :";
  cin >> x;
  bin(x); // Виклик функції для переведення числа у двійкову систему числення
  cout << "\n\nBinary operations : \n";
  xinv = ~x;
  cout << "Invert number (2 byte) : " <<  xinv << " = "; bin(xinv);
  if (x<255)
  {
     xinv = xinv & 255;
     cout << "\nInvert number (1 byte) : " <<  xinv << " = ";
     bin(xinv);
  }
  int k = 16-3-1;
  x3 = x << k;
  x3 = x3 >> (k+3);
  cout << endl<< "3 bit = " << x3 << endl;
  x4_1 = x | 0x10; // або x|16
  cout << "Number with 4bit=1 : " << x4_1 << " = "; bin(x4_1);
  x0_0 = x & 0xFFFE;
  cout << "\nNumber with 0bit=0 : " << x0_0 << " = "; bin(x0_0);
  xinv_2=x^4;
  cout << "\nNumber after invert 2bit : " << xinv_2 << " = "; bin(xinv_2);
  x_r2 = x >> 2;
  cout << "\nNumber after >>2 : " << x_r2 << " = "; bin(x_r2);
  cout << endl;
  system("pause>>void");
  return 0;
}
```

Результати виконання програми:

```
Input number :105
 0110 1001

Binary operations :
Invert number (2 byte) : 65430 =  1111 1111 1001 0110
Invert number (1 byte) : 150 =  1001 0110
3 bit = 1
Number with 4bit=1 : 121 =  0111 1001
Number with 0bit=0 : 104 =  0110 1000
Number after invert 2bit : 109 =  0110 1101
Number after >>2 : 26 =  0001 1010
```

Контрольні запитання та завдання для самоконтролю

1) Назвати усі побітові логічні операції?
2) Як побітові логічні операції позначаються у мові С++?
3) Вказати пріоритет виконання логічних побітових операцій.
4) З якою операцією є еквівалентним побітовий зсув праворуч? побітовий зсув ліворуч?

Завдання до самостійної роботи

1) Перевести число, вибране відповідно до індивідуального варіанта у табл. 3.5, у двійкову та шіснадцяткову системи числення.

2) Над цим двійковим числом виконати такі побітові логічні операції:

– інвертувати усі біти числа;
– встановити (задати значення 1) для *n*-ого біта числа;
– визначити (вибрати окремо) значення *k*-го біта числа;
– скинути (встановити в 0) *m*-й біт числа;
– інвертувати *p*-й біт числа;
– логічний побітовий зсув на задану кількість розрядів за вказаним напрямом (ліворуч або праворуч);

де *n, k, m, p* – номери бітів для кожного варіанта. Усі дії виконувати над вихідним числом.

3) Результати виконання побітових операцій перевірити за допомогою відповідного програмного коду.

Таблиця 3.5

Індивідуальні варіанти завдань

№ варіанта	Десяткове число	Номер біта (*n*), який слід встановити	Номер біта (*k*), який слід визначити	Номер біта (*m*), який слід скинути	Номер біта (*p*), який слід інвертувати	Кількість бітів для зсуву та напрям
1	65	1	4	0	5	2, →
2	30	2	5	1	6	2, ←
3	16	3	6	2	7	3, ←
4	25	4	7	3	0	4, ←
5	87	5	0	4	1	3, →
6	63	6	1	5	2	1, →
7	94	7	2	6	3	1, ←
8	83	0	3	7	4	2, →
9	43	1	4	0	5	3, ←
10	64	2	5	1	6	2, ←
11	97	3	6	2	7	1, ←
12	84	4	7	3	0	1, →
13	11	5	0	4	1	3, →
14	20	6	1	5	2	4, ←
15	68	7	2	6	3	3, →
16	12	0	3	7	4	3, ←
17	33	1	4	0	5	1, →
18	44	2	5	1	6	2, ←
19	29	3	6	2	7	1, ←
20	35	4	7	3	0	2, →
21	56	5	0	4	1	1, →
22	34	6	1	5	2	3, ←
23	48	7	2	6	3	2, ←
24	67	0	3	7	4	1, ←
25	81	1	4	0	5	1, →
26	96	2	5	1	6	2, →
27	98	3	6	2	7	3, →
28	33	4	7	3	0	3, ←
29	21	5	0	4	1	4, ←
30	23	5	1	4	6	1, ←

Лабораторна робота № 4
Оператор вибору варіантів switch

Мета роботи: набути практичних навиків використання операторів вибору варіантів switch та безумовного переходу goto при створюванні програмних проєктів розгалуженої структури в С++.

Теоретичні відомості

1. Оператор безумовного переходу goto

Оператор goto (перейти до) дозволяє передавати керування у будь-яку точку коду (програми), котру позначено спеціальною міткою.

Синтаксис оператора goto:

goto <*мітка*>;

Мітку записують перед оператором, на який слід передати керування, і відокремлюють від нього символом двокрапки (:). Мітки в С++ не оголошують. В якості мітки може застосовуватись поєднання будь-яких літер латиниці і цифр, але розпочинатися мітка повинна з літери, наприклад: start, M1, second.

Застосовувати цей оператор слід обережно і помірковано, особливо при переході всередину блока чи циклу, оскільки це може призвести до непередбачуваних помилок. Тому в С++ оператор goto застосовується рідко і вважається застарілим. Проте, на практиці часто трапляються випадки, коли цей оператор значно спрощує код програми.

2. Оператор вибору варіантів switch

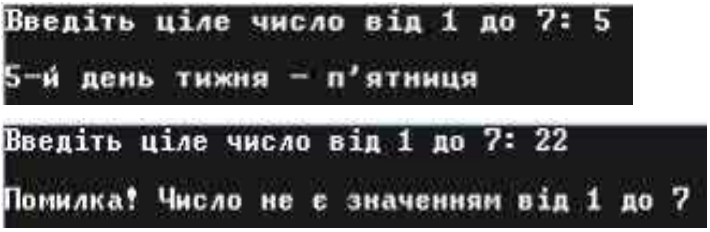
Формат оператора вибору варіантів switch:

```
switch  (<вираз>)
{  case <значення-мітка_1> :  {<послідовність операторів>; break;}
   ……….
   case <значення-мітка_n> : {<послідовність операторів>; break;}
  [ default:   <послідовність операторів>; ]
}
```

Спочатку обчислюється *вираз* у дужках. *Вираз* повинен мати цілий або символьний тип. Значення *виразу* порівнюється зі значеннями *міток* після ключових слів case. Якщо значення *виразу* збіглося зі значенням якої-небудь *мітки*, то виконується відповідна *послідовність операторів*, позначена цією *міткою* і записана після двокрапки, доки не зустрінеться оператор break. Якщо значення *виразу* не збіглося з жодною *міткою*, то виконуються оператори, які йдуть за ключовим словом default. Мітка default є необов'язковою конструкцією оператора switch, на що вказують квадратні дужки [] у форматі. Оператор break

здійснює вихід із switch. Якщо оператор break є відсутній наприкінці операторів відповідного case, то будуть почергово виконані всі оператори до наступного break чи то до кінця switch для всіх гілок case незалежно від значення їхніх міток.

Розглянемо роботу оператора switch на прикладі функції, яка за значенням введеного цілого числа виводить назву дня тижня.




```
#include <iostream>
#include <locale.h>
using namespace std;
int main()
{
  setlocale(0,".1251");
  int n; char *s="";
  cout << "Введіть ціле число від 1 до 7: ";  cin >> n;
  switch (n)
  {  case 1: s="понеділок"; break;
     case 2: s="вівторок"; break;
     case 3: s="середа";   break;
     case 4: s="четвер";   break;
     case 5: s="п'ятниця"; break;
     case 6: s="субота";   break;
     case 7: s="неділя";   break;
     default: { cout<<"\nПомилка! Число не є значенням від 1 до 7\n";
                system("pause>>void"); return 0;
   }          }
   cout << endl << n << "-й день тижня - " << s << endl;
   system("pause>>void");
   return 0;
}
```

Наведемо ще один приклад програми з використанням оператора switch для реалізації простого калькулятора в консольному режимі.

```
#include <iostream>
using namespace std;
int main()
{
 setlocale(0,".1251");
 double a,b,res;  char op;
 cout << "\n Введіть перший операнд  "; cin >> a;
 cout << "\n Введіть знак операції   ";  cin >> op;
 cout << "\n Введіть другий операнд  "; cin >> b;
 bool f=true;
 switch (op)
 { case '+' : res = a+b; break;
   case '-' : res = a-b; break;
   case '*' : res = a*b; break;
   case '/' : res = a/b; break;
   default: cout << "\n Невідома операція!\n "; f=false;
```

```
 }
 if (f) cout << "\n Результат : " << res<<endl;
 system("pause>>void");
 return 0;
}
```

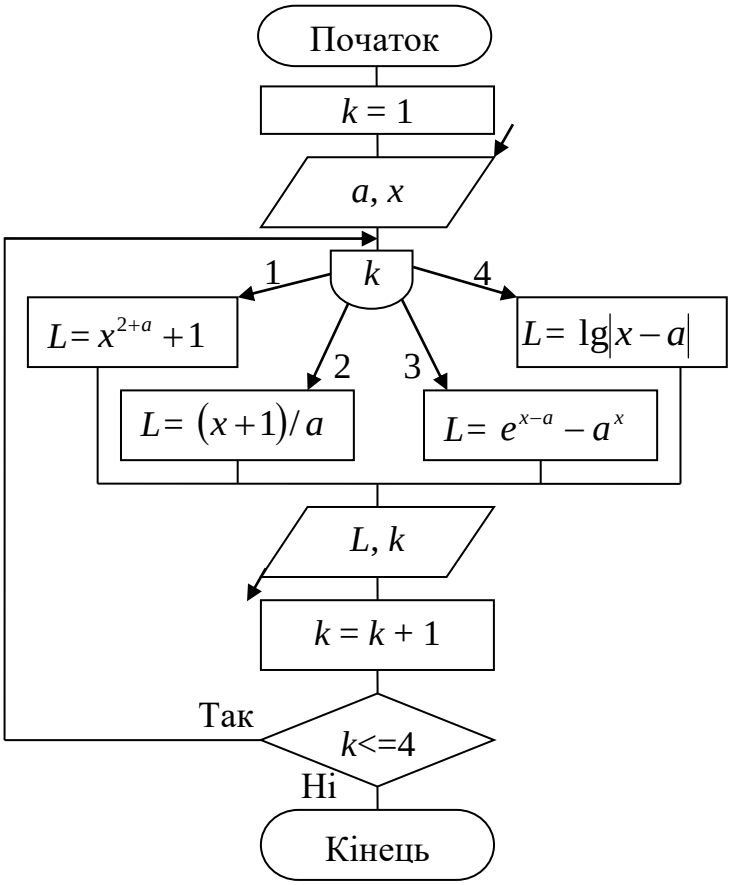
```
Введіть перший операнд  0.25
Введіть знак операції   *
Введіть другий операнд  50
Результат : 12.5
```

```
Введіть перший операнд  89
Введіть знак операції   ^
Введіть другий операнд  10
Невідома операція!
```

Приклади програм з розгалуженою структурою, зорганізованих за допомогою оператора switch

Приклад 1. Обчислити значення функції $L=\begin{cases} x^{2+a}+1 & \text{за } k=1; \\ (x+1)/a & \text{за } k=2; \\ e^{x-a}-a^{x} & \text{за } k=3; \\ \lg|x-a| & \text{за } k=4 \end{cases}$ для всіх значень параметра k.

Текст програми та схема алгоритму:




```
#include <iostream>
#include <math.h>
using namespace std;

int main()
{
 setlocale(0,".1251");
 double x, a, L;
 int k = 1;
 cout<<"Введіть значення a= ";
 cin>> a;
 cout<<"Введіть значення x= ";
 cin>> x;
 cout<< "\nРезультати:\n";
M1:
  switch (k)
  {
    case 1: { L = pow(x,2+a) + 1; break; }
    case 2: { L = (x+1)/a; break; }
    case 3: { L = exp(x-a) - pow(a,x); break; }
    case 4: { L = log10(fabs(x-a)); break; }
  }
```

```
 cout<< "L" << k << " = " << L << endl;
 k++;
 if(k <= 4) goto M1;
 system("pause>>void");
 return 0;
}
```

Результати роботи:

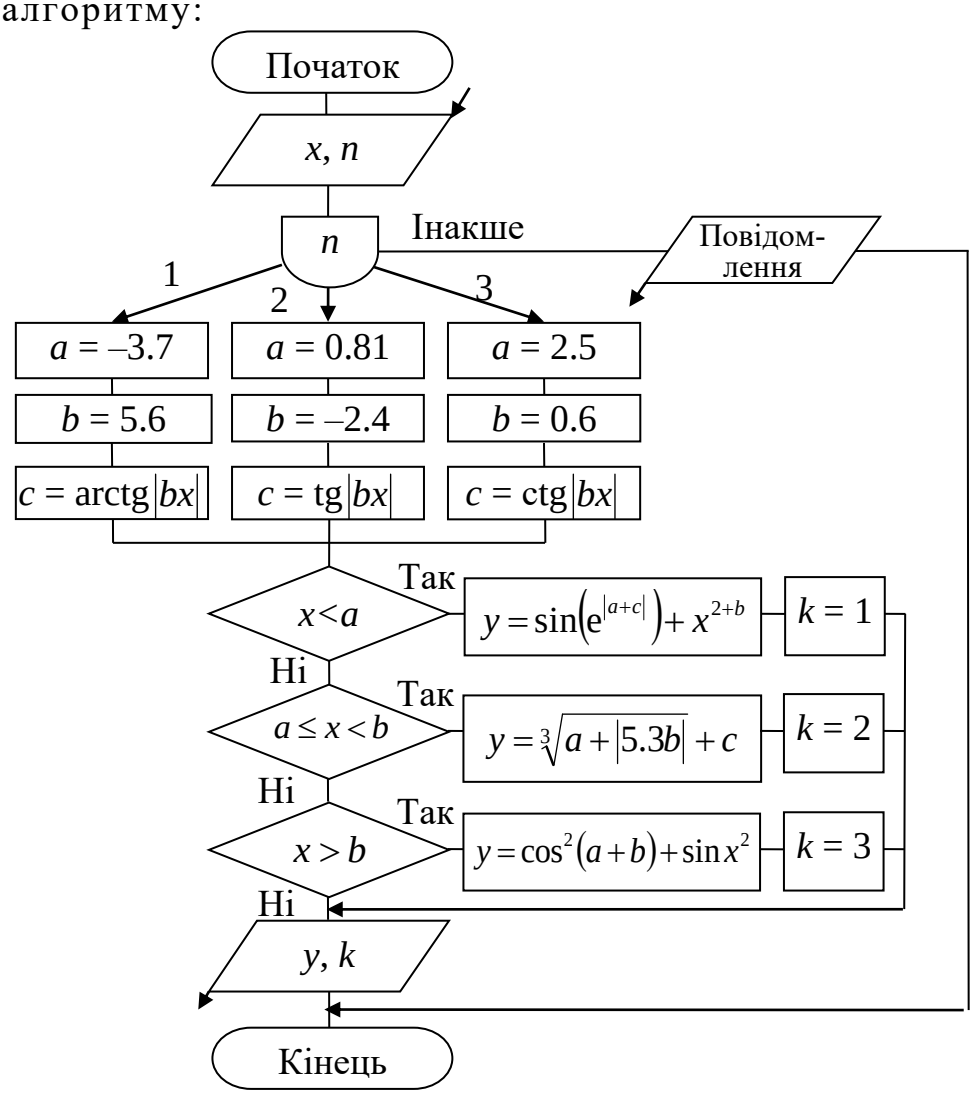
```
Введіть значення a= 0.8
Введіть значення x= 10.5

Результати:
L1  = 724.319
L2  = 14.375
L3  = 16317.5
L4  = 0.986772
```

Приклад 2. Ввести значення x та обчислити значення y для одного з трьох варіантів параметрів, номер якого слід ввести:

$$y=\begin{cases}\sin\left(e^{|a+c|}\right)+x^{2+b} & \text{за} \quad x<a;\\ \sqrt[3]{a+|5.3b|}+c & \text{за} \quad a\le x<b;\\ \cos^2(a+b)+\sin x^2 & \text{за} \quad x>b\end{cases}$$

1) $a = -3{,}7;\ b = 5{,}6;\ c = \operatorname{arctg}|bx|$;
2) $a = 0{,}81;\ b = -2{,}4;\ c = \operatorname{tg}|bx|$;
3) $a = 2{,}5;\ b = 0{,}6;\ c = \operatorname{ctg}|bx|$.

Схема алгоритму:



Текст програми:

```
#include <iostream>
```

```
#include <math.h>
using namespace std;
int main()
{
  system("chcp 1251 > null");
  system("color F0");
  double x, y, a, b, c;       int k, n;
  cout<<" Введіть номер варіанта (ціле значення) 1, 2 або 3: ";
  cin>>n;
  cout<<" Введіть довільне значення x= ";
  cin>>x;
  switch (n)
  { case   1:  a=-3.7;  b=5.6;  c=atan(fabs(b*x)); break;
    case   2:  a=0.81;  b=-2.4; c=tan(fabs(b*x)); break;
    case   3:  a=2.5;   b=0.6;  c=1/tan(fabs(b*x)); break;
    default: { cout<<" Некоректне значення варіанта!"<<endl;
               system("pause>>void");
               return 0;
  }          }
  cout<< endl << " Результати:" << endl;
  if (x<a)  { y=sin(exp(fabs(a + c)))+pow(x+b,2);  k=1;}
  else
    if (x>=a && x<b)  { y=pow(a+fabs(5.3*b),1.0/3)+c;  k=2;}
    else
      if (x>=b) { y=pow(cos(a+b),2)+sin(x*x);  k=3;}
  cout<<" y= " << y << "  була виконана умова №" << k << endl;
  system("pause>>void");
  return 0;
}
```

Результати роботи:

```
Введіть номер варіанта (ціле значення) 1, 2 або 3: 2
Введіть довільне значення x= 4.3

Результати:
y= -0.35149  була виконана умова №3
```

```
Введіть номер варіанта (ціле значення) 1, 2 або 3: 7
Введіть довільне значення x= 0.4
Некоректне значення варіанта!
```

Контрольні запитання та завдання для самоконтролю

1) Які оператори в C++ використовуються для організації розгалужень?
2) Як працює оператор `switch`?
3) У чому відмінність і схожість операторів `if` та `switch`?

4) Яким буде значення y після виконання фрагментів програми:

а)
```
double y=0; int n=1;
  switch (n)
  {case 1:  y=n/4.; break;
   case 2:  y=n*n;  break;
   case 3:  y=n;    break;
  }
```

б)
```
double y=0; int n=3;
  switch (n)
  {case 1:  y=n/4.;
   case 3:  y=n*n;
   case 5:  y=n+1;
  }
```

в)
```
double y=0; int n=4;
switch (n)
{case 2:  { y=n/4.; break; }
 case 5:  { y=n*n;  break; }
 case 9:  { y=n;    break; }
}
```

г)
```
double y=0; int n=1;
  switch (n)
  {case 1: { y=n/4; }
   case 3: { y=n*n; }
   case 5: { y=n+1; }
  }
```

Лабораторне завдання

1) У протоколі лабораторної роботи дати відповіді на контрольні запитання.

2) У протоколі лабораторної роботи скласти схеми алгоритмів і написати програми мовою C++ із застосуванням оператора варіантів `switch` для розв'язання завдань, поданих у табл. 4.1 … 4.2 відповідно до індивідуального варіанта.

3) Створити на комп'ютері програмні проєкти мовою C++ для реалізації написаних програм. Занести результати обчислень до протоколу.

Таблиця 4.1

Індивідуальні завдання базового рівня складності

№ вар.	Функція	№ вар.	Функція
1	$y=\begin{cases}2x+1 & \text{за } k=1;\\ \sqrt[3]{1-x^4} & \text{за } k=2;\\ \lg\lvert x+5\rvert & \text{за } k=3;\\ \ln\left\lvert\dfrac{1+x}{x^3+\cos x}\right\rvert & \text{за } k=4\end{cases}$	2	$y=\begin{cases}\dfrac{\operatorname{tg}\lvert 1+e^{x+1.2}\rvert}{x+\sin x} & \text{за } n=1;\\ \sqrt[4]{\cos\lvert\pi+x\rvert} & \text{за } n=2;\\ \dfrac{1+x^{x+1}-\lg x}{x^3+\ln\lvert x\rvert} & \text{за } n=3\end{cases}$
3	$y=\begin{cases}1/x+\operatorname{arctg}^2 x^3 & \text{за } M=1;\\ 2^{x-1}+\sin^2 x+\lg\lvert x\rvert & \text{за } M=2;\\ \sqrt{\lvert 1+x\rvert}-\sqrt[3]{x} & \text{за } M=3\end{cases}$	4	$y=\begin{cases}10^{-3}+\sin x^3 & \text{за } z=1;\\ \sqrt{1+x}+\sin^2 x & \text{за } z=2;\\ \lg(1/x+\sqrt{x}) & \text{за } z=3\end{cases}$
5	$t=\begin{cases}\sqrt{\lvert 2^x-x^2\rvert+0.5} & \text{за } k=1;\\ 1+\operatorname{arctg}(x) & \text{за } k=2;\\ \sqrt[5]{\pi^2+x^2} & \text{за } k=3;\\ \lg\lvert 6.5-x^4\rvert & \text{за } k=4\end{cases}$	6	$y=\begin{cases}2^{x+1}+1 & \text{за } k=1;\\ \sqrt[3]{e^{x^2}+x^4} & \text{за } k=2;\\ \lg\lvert\sin(\pi-x)\rvert & \text{за } k=3;\\ \operatorname{tg}\dfrac{1+x}{x^3+x^x} & \text{за } k=4\end{cases}$

Продовження табл. 4.1

№ вар.	Функція	№ вар.	Функція
7	$y=\begin{cases}\sqrt{x+1}-\cos^2 x & \text{за } k=1;\\ \mathrm{e}^{0.01}+\ln x^2, & \text{за } k=2;\\ \sqrt{x}+\sin^2(x-\pi) & \text{за } k=3;\\ x+\lg x & \text{за } k=4\end{cases}$	8	$y=\begin{cases}3{,}5x-7.3x^2\operatorname{ctg}x & \text{за } L=1;\\ 2{,}8\ln x+\mathrm{e}^{\sqrt{x}} & \text{за } L=2;\\ \sqrt[3]{3.4x}+x^2\sin x & \text{за } L=3;\\ 1.7^x+\cos^2 x^2 & \text{за } L=4\end{cases}$
9	$y=\begin{cases}\sin\mathrm{e}^{x+1.2} & \text{за } n=1;\\ \sqrt[5]{\lg\lvert 1+x\rvert}, & \text{за } n=2;\\ \operatorname{tg}\lvert\cos x+5\pi/4\rvert & \text{за } n=3;\\ \dfrac{1+x^{x+1}-x}{x^3+\ln\lvert x\rvert} & \text{за n}=4\end{cases}$	10	$y=\begin{cases}7.8x^3-\operatorname{tg}(3.1x^2+4x) & \text{за } k=1;\\ \mathrm{e}^{0.85\sqrt{x}}(x^2+3) & \text{за } k=2;\\ \sin(2x+\pi)+\mathrm{e}^{4x} & \text{за } k=3;\\ x\dfrac{\sqrt[3]{x+\cos(\pi/2+x)}}{x^{2^x}+0.1\cdot 10^{-3}} & \text{за } k=4\end{cases}$
11	$y=\begin{cases}\operatorname{arctg}(2x+1)+1 & \text{за } k=1;\\ \sqrt[3]{1+x^4} & \text{за } k=2;\\ \cos\left(\dfrac{\pi}{2}-x^x\right)+\mathrm{e}^{\lvert x+5\rvert} & \text{за } k=3;\\ \lg\dfrac{1+x}{x^3+\sqrt{\lvert x\rvert}} & \text{за } k=4\end{cases}$	12	$y=\begin{cases}\dfrac{4x^2t}{2x-3t+2} & \text{за } n=1;\\ 6.2x-\dfrac{\ln\sqrt{x^2+0.1}}{\sqrt{\lvert 2x-\cos x\rvert}} & \text{за } n=2;\\ 8.3t^3+x-0.2 & \text{за } n=3\end{cases}$
13	$y=\begin{cases}\sqrt[5]{x+1} & \text{за } k=1;\\ \operatorname{tg}(\cos x+\pi/2) & \text{за } k=2;\\ \mathrm{e}^{2x^2}+\sqrt{\lvert 1-x\rvert} & \text{за } k=3;\\ \sin^2(x^2+3) & \text{за } k=4;\\ \cos 3x^2 & \text{за } k=5\end{cases}$	14	$y=\begin{cases}2x^2+\lg\lvert x\rvert & \text{за } n=1;\\ \cos^2 x+2.8\sqrt[3]{x} & \text{за } n=2;\\ \sin^2\sqrt{\lvert x\rvert} & \text{за } n=3;\\ \ln\left\lvert\dfrac{x+1}{4}\right\rvert & \text{за } n=4\end{cases}$
15	$y=\begin{cases}\sqrt{x}+\mathrm{e}^x & \text{за } L=1;\\ \ln\lvert x\rvert+0.8 & \text{за } L=2;\\ x^2/\lg^2\lvert x\rvert & \text{за } L=3;\\ x\cos^2 x+\sqrt{x} & \text{за } L=4;\\ \mathrm{e}^{0.8x}+\lvert x\rvert & \text{за } L=5\end{cases}$	16	$y=\begin{cases}\cos^2(x+\pi/2) & \text{за } S=1;\\ \operatorname{ctg}^2\sqrt{x}+1/x & \text{за } S=2;\\ 2\sin x+\ln\lvert x\rvert & \text{за } S=3;\\ \dfrac{0{,}8x^2}{\mathrm{e}^x+x^x+x} & \text{за } S=4\end{cases}$
17	$y=\begin{cases}\mathrm{e}^{2x}-\sin^2 x & \text{за } S=1;\\ \cos^2 x+\ln\lvert x\rvert & \text{за } S=2;\\ \sin^2 x-\ln\lvert x\rvert & \text{за } S=3;\\ x+\sin\sqrt{x} & \text{за } S=4\end{cases}$	18	$y=\begin{cases}\cos^2 x & \text{за } S=1;\\ \sin x^2+1/x & \text{за } S=2;\\ 2\ln\lvert x\rvert+\mathrm{e}^x & \text{за } S=3;\\ 8x^2-\operatorname{arctg}x & \text{за } S=4\end{cases}$

Закінчення табл. 4.1

№ вар.	Функція	№ вар.	Функція
19	$S=\begin{cases}\dfrac{\cos(\pi-x^2)}{x+1} & \text{за } n=1;\\ \operatorname{tg}(\pi x^2) & \text{за } n=2;\\ \sqrt[3]{e^{x+1}-\ln\lvert x\rvert} & \text{за } n=3;\\ x^2-2^x & \text{за } n=4\end{cases}$	20	$y=\begin{cases}\sqrt{x}+e^x & \text{за } L=1;\\ \ln x+0.8 & \text{за } L=2;\\ x^2+\sin^2 x & \text{за } L=3;\\ x\cos x^2+\sqrt{x} & \text{за } L=4;\\ e^{0.8x}+\ln\lvert x\rvert & \text{за } L=5\end{cases}$
21	$y=\begin{cases}\sqrt[3]{x+1} & \text{за } k=1;\\ \sin(\cos x+\pi/2) & \text{за } k=2;\\ e^{1+x^2}+\lg\sqrt{\lvert 1-x\rvert} & \text{за } k=3;\\ \sin^3(x^2+\pi) & \text{за } k=4;\\ \arccos(3-x^2) & \text{за } k=5\end{cases}$	22	$y=\begin{cases}\sqrt{e^x-1} & \text{за } L=1;\\ \lg\lvert x\rvert+1/x & \text{за } L=2;\\ 2^{x-1}+\arcsin^2 x & \text{за } L=3;\\ x\cos^2 x+\sqrt{x} & \text{за } L=4;\\ \ln\sqrt{\lvert x+0.1\rvert} & \text{за } L=5\end{cases}$
23	$y=\begin{cases}\dfrac{\sqrt{x}+e^x}{e^{0.1x}+\lg\lvert x\rvert} & \text{за } L=1;\\ \ln x+\operatorname{ctg}\left(\sqrt{\lvert\pi-x\rvert}\right) & \text{за } L=2;\\ x^2+\sin^2 x & \text{за } L=3;\\ x\cos^2 x+\sqrt{x} & \text{за } L=4\end{cases}$	24	$y=\begin{cases}\sin(x-\pi/2) & \text{за } S=1;\\ \operatorname{tg}^3\lvert\sqrt{x}\rvert+x & \text{за } S=2;\\ \arcsin^2 x+\lg\lvert x\rvert & \text{за } S=3;\\ \dfrac{2{,}4-x^2}{e^x+x^x} & \text{за } S=4\end{cases}$
25	$y=\begin{cases}e^{x^2}+0.8x^2 & \text{за } K=1;\\ \ln x^2+\sin^2 x & \text{за } K=2;\\ \sqrt{\lvert x\rvert}+\lg\lvert x\rvert & \text{за } K=3;\\ x+\operatorname{tg}^2(x-\pi) & \text{за } K=4\end{cases}$	26	$S=\begin{cases}4\ln x^2-e^{\lvert x\rvert} & \text{за } K=1;\\ \operatorname{ctg}\left(\sqrt{\lvert\pi-x\rvert}\right) & \text{за } K=2;\\ \sin^2(x+\pi) & \text{за } K=3;\\ \operatorname{tg}\left(x+10^{-3}\right) & \text{за } K=4\end{cases}$
27	$y=\begin{cases}\sin(e^{1+x}+1)+x^2 & \text{за } K=1;\\ \sqrt{2x+\lvert\sin x\rvert}+x & \text{за } K=2;\\ 1/\cos x^2+x & \text{за } K=3;\\ 2x-\sin^2 x & \text{за } K=4\end{cases}$	18	$y=\begin{cases}2.5a+\sin x^2 & \text{за } N=1;\\ \lg^2\lvert x+1\rvert & \text{за } N=2;\\ \operatorname{tg}(x-\pi/4) & \text{за } N=3;\\ 2x+\dfrac{\sin x}{\sqrt{x}} & \text{за } N=4\end{cases}$
29	$y=\begin{cases}3x^2+\operatorname{arctg}x & \text{за } L=1;\\ 0{,}2\ln x+e^{\sqrt{\lvert x\rvert}} & \text{за } L=2;\\ \sqrt[3]{4-x}+x^3\sin x & \text{за } L=3;\\ 1/e^{x^2}+\cos^5 x & \text{за } L=4\end{cases}$	30	$y=\begin{cases}\sqrt{x^4+1}-\cos x & \text{за } k=1;\\ e^{0.1x}+\ln x^2 & \text{за } k=2;\\ \sqrt{5x}+\operatorname{ctg}^2(x-\pi) & \text{за } k=3;\\ x+\lg\lvert x\rvert & \text{за } k=4\end{cases}$

Таблиця 4.2

Індивідуальні завдання середнього рівня складності

№ вар.	Функції	Варіанти параметрів
1	$y=\begin{cases}\dfrac{(2u+1)^2}{7\pi+x} & \text{за} \quad u+x<-0.5;\\ \cos^2 u-\sin\dfrac{u}{3} & \text{за} \quad -0.5\le u+x\le 10^{-3};\\ \dfrac{\lg(u+x)-\mathrm{e}^x}{3.5x} & \text{за} \quad u+x>10^{-3}\end{cases}$	1 $u=\sin x$; 2 $u=\cos x$; 3 $u=\operatorname{tg} x$
2	$y=\begin{cases}abx-\cos^2(zx) & \text{за} \quad x<3.5a;\\ (a-x)^2-\ln(z+x) & \text{за} \quad 3.5a\le x\le b;\\ \sqrt{bx-a+zx^2} & \text{за} \quad x>b\end{cases}$	1 $a=0.4;\ b=2.3;\ z=e^{2x}$; 2 $a=0.2;\ b=0.8;\ z=e^{x}$; 3 $a=0.7;\ b=8.1;\ z=0.8$
3	$y=\begin{cases}\sin(bm+\cos(nx)) & \text{за} \quad \lvert bm\rvert>x^2;\\ \cos(bm-\sin x) & \text{за} \quad \lvert bm\rvert<x^2;\\ \sqrt{\mathrm{e}^{\lvert\cos x\rvert}+\sqrt{\lvert bmx\rvert}} & \text{за} \quad \lvert bm\rvert=x^2\end{cases}$	1 $b=-1.6;\ m=0.9;\ n=-1.4$; 2 $b=4.5;\ m=-2;\ n=2.2$; 3 $b=-4.5;\ m=0.5;\ n=-1.5$
4	$y=\begin{cases}a\sin^2 x+b\cos(zx) & \text{за} \quad x<-\ln(a);\\ a^b-\cos^3(a+zx) & \text{за} \quad -\ln(a)\le x\le b;\\ \sqrt{2.5a^3+(b-zx^2)^6} & \text{за} \quad x>b\end{cases}$	1 $a=0.2;\ b=0.5;\ z=\mathrm{e}^{ax}$; 2 $a=0.15;\ b=0.2;\ z=\mathrm{e}^{2ax}$; 3 $a=0.9;\ b=5;\ z=\mathrm{e}^{2.5ax}$
5	$y=\begin{cases}\sin(\mathrm{e}^{a+b})+x^2 & \text{за} \quad \mathrm{e}^{a+b}>\mathrm{e}^x;\\ \operatorname{arctg}(abc)+\sqrt[3]{x} & \text{за} \quad \mathrm{e}^{a+b}=\mathrm{e}^x;\\ \cos\left(\sqrt{\lvert x+abc\rvert}\right) & \text{за} \quad \mathrm{e}^{a+b}<\mathrm{e}^x\end{cases}$	1 $a=4{,}2;\ b=5.3;\ c=1.5$; 2 $a=-0.35;\ b=1.8;\ c=-1.8$; 3 $a=2.8;\ b=-0.6;\ c=2.0$
6	$y=\begin{cases}2.8\sin^2 ax-bx^3z & \text{за} \quad x<a;\\ z\cos(ax+b)^2+\ln(z) & \text{за} \quad a\le x\le b^2;\\ \mathrm{e}^{2.5ax}+zabx & \text{за} \quad x>b^2\end{cases}$	1 $a=-5;\ b=2.5;\ z=\ln\lvert bx^3\rvert$; 2 $a=3;\ b=5;\ z=\ln\lvert bx\rvert$; 3 $a=-10;\ b=3,\ z=\ln\lvert bx^2\rvert$
7	$y=\begin{cases}x\mathrm{e}^a+\mathrm{e}^{\lvert bc\rvert} & \text{за} \quad \lvert 1-x^2\rvert=a+c;\\ \sin^2 ax+\cos bc & \text{за} \quad \lvert 1-x^2\rvert>a+c;\\ \sqrt{ab^4+\sqrt[5]{cx^2}} & \text{за} \quad \lvert 1-x^2\rvert<a+c\end{cases}$	1 $a=3.2;\ b=-0.7;\ c=2.2$; 2 $a=10.5;\ b=-2.5;\ c=5.6$; 3 $a=5.4;\ b=3;\ c=2.6$
8	$y=\begin{cases}\ln\lvert mx+n\rvert & \text{за} \quad x^2>m+n;\\ \mathrm{e}^{\cos\lvert mx-n\rvert} & \text{за} \quad x^2=m+n;\\ \sqrt[3]{k^2+\cos^2 x} & \text{за} \quad x^2<m+n\end{cases}$	1 $k=3.1;\ m=5.15;\ n=-1.15$; 2 $k=0.78;\ m=-2.4;\ n=4.36$; 3 $k=1.1;\ m=0.8;\ n=0.41$

Продовження табл. 4.2

№ вар.	Функції	Варіанти параметрів
9	$y=\begin{cases} a\sin^2 x+b\cos(zx+a) & \text{за } x<a^3; \\ (a+bx)^2-\sin(a+zx) & \text{за } a^3\le x\le b; \\ \sqrt{x-(\sin(bx+z))} & \text{за } x>b \end{cases}$	1 $a=1.2;\ b=7.2;\ z=\mathrm{e}^x$; 2 $a=-1.5;\ b=3.2;\ z=\mathrm{e}^{2x}$; 3 $a=1.7;\ b=5.5;\ z=\mathrm{e}^3$
10	$y=\begin{cases} \sqrt[3]{b^2+\sqrt{\lvert x+c\rvert}} & \text{за } \lg a<x; \\ \cos(x-b-c) & \text{за } \lg a=x; \\ \sin(x+a-b) & \text{за } \lg a>x \end{cases}$	1 $a=0.1;\ b=9.8;\ c=11.12$; 2 $a=10;\ b=10.05;\ c=6.2$; 3 $a=100;\ b=3.03;\ c=7.12$
11	$y=\begin{cases} \ln(\lvert\lg\lvert kx+mn\rvert\rvert) & \text{за } 3x>\lvert m+n\rvert; \\ \sin(kmx)+\sqrt{\lvert nx\rvert} & \text{за } 3x=\lvert m+n\rvert; \\ \mathrm{e}^{\cos x}+\mathrm{e}^{m+n} & \text{за } 3x<\lvert m+n\rvert \end{cases}$	1 $k=4;\ m=-14.7;\ n=-0.6$; 2 $k=3;\ m=6.5;\ n=3.15$; 3 $k=5;\ m=-12;\ n=0.45$
12	$y=\begin{cases} \mathrm{e}^{ax}-3.5\cos^2(z+bx) & \text{за } x\le a; \\ a+\ln\lvert a+bx\rvert-2x & \text{за } a<x\le b^{3.5}; \\ a+\cos^{3.5}(a+bxz) & \text{за } x>b^{3.5} \end{cases}$	1 $a=-1;\ b=3.4;\ z=\operatorname{tg} bx$; 2 $a=-3.2;\ b=5.5;\ z=\operatorname{tg} bx^2$; 3 $a=-5.2;\ b=7.2;\ z=\operatorname{tg} bx^3$
13	$y=\begin{cases} x^2\mathrm{e}^{2k}+\ln\lvert rx\rvert & \text{за } \cos x=\cos(rs); \\ \sqrt[3]{x^2}+\sqrt{\lvert k+rsx\rvert} & \text{за } \cos x>\cos(rs); \\ \operatorname{arctg}(kx+rs) & \text{за } \cos x<\cos(rs) \end{cases}$	1 $k=1.33;\ r=0.85;\ s=3{,}5$; 2 $k=0.9;\ r=3{,}3;\ s=1.2$; 3 $k=1.57;\ r=0.75;\ s=2.15$
14	$y=\begin{cases} 2.5b^2+ax-4.5\cos xz & \text{за } x\le 5a; \\ (a^2-5.4x)^3+\ln(xz) & \text{за } x>b; \\ \sqrt{6.5b^2+(a-x^3z)} & \text{за } 5a<x\le b \end{cases}$	1 $a=0.5;\ b=4.5;\ z=\mathrm{e}^{ax}$; 2 $a=0.5;\ b=3.7;\ z=\mathrm{e}^{2ax}$; 3 $a=0{,}5;\ b=2.7;\ z=\mathrm{e}^{2.5ax}$
15	$y=\begin{cases} a\cos^2 x+b\sin zx & \text{за } x\le a; \\ \operatorname{tg}(ax+z)+\sin^2 bx & \text{за } a<x\le 1.5b; \\ \ln(ax-b)+z^2 & \text{за } x>1.5b \end{cases}$	1 $a=4.5;\ b=8.4;\ z=\operatorname{tg}(bx)^2$; 2 $a=8.2;\ b=15.2;\ z=\operatorname{tg}(bx)^2$; 3 $a=1.7;\ b=0.5;\ z=\operatorname{tg}(bx^2)$
16	$y=\begin{cases} 3.5\sin^2(bx+z)^3-\mathrm{e}^{3.5a} & \text{за } x\le a; \\ \ln(a+b^3x)+a & \text{за } a<x\le b^{2.5}; \\ \cos^2(a^b+xz)+a^2 & \text{за } x>b^{2.5} \end{cases}$	1 $a=0.1;\ b=0.5;\ z=\mathrm{e}^{2.5ax}$; 2 $a=1.2;\ b=2.5;\ z=\mathrm{e}^{2.5ax}$; 3 $a=2.5;\ b=1.2;\ z=\mathrm{e}^{2.5ax}$
17	$y=\begin{cases} \sqrt{\lvert ax-\cos^2 b^3x+5.1c^2\rvert} & \text{за } \lvert 1-x^2\rvert=a+c; \\ \mathrm{e}^{0.04x}+\ln\lvert b^5\cos x\rvert & \text{за } \lvert 1-x^2\rvert>a+c; \\ \cos^2(b^3x^2)+\ln\lvert bx-a^2\rvert & \text{за } \lvert 1-x^2\rvert<a+c \end{cases}$	1 $a=3.5;\ b=-0.73;\ c=2.5$; 2 $a=15{,}4;\ b=-5.6;\ c=3.5$; 3 $a=5.1;\ b=4;\ c=2.7$

Продовження табл. 4.2

№ вар.	Функції	Варіанти параметрів
18	$y=\begin{cases} a+\sin bx+\cos x^2 & \text{за} \quad x\le a; \\ \sqrt{a+bx}+\sin zx & \text{за} \quad a<x<\ln b; \\ \ln(a+bx+z) & \text{за} \quad x\ge \ln b \end{cases}$	1 $a=-1.2;\ b=0.75;\ z=\ln\lvert\operatorname{tg}(bx)\rvert$; 2 $a=0.4;\ b=2.4;\ z=\ln\lvert\operatorname{tg}(bx)\rvert$; 3 $a=1.1;\ b=6.1;\ z=\ln\lvert\operatorname{tg}(bx)\rvert$
19	$y=\begin{cases} \dfrac{(2z+1)^2}{3.71-x^2} & \text{за} \quad z>-0.5; \\ \sin^3 z-\sin\dfrac{z}{3\pi} & \text{за} \quad -0.5\le z\le 10^{-3}; \\ \dfrac{\operatorname{tg}(z+x)-\mathrm{e}^x}{3.5x} & \text{за} \quad z>10^{-3} \end{cases}$	1 $z=\arcsin x^3$; 2 $z=\arccos^2 x$; 3 $z=\operatorname{tg} x$
20	$y=\begin{cases} (3.5a-7.3bx+\sin(zx))^3 & \text{за} \quad x<-\ln\lvert a\rvert; \\ a^b-\cos^3(a+zx) & \text{за} \quad -\ln\lvert a\rvert\le x<b; \\ \sqrt{\lvert\operatorname{tg} a-x\rvert}-x^2 & \text{за} \quad x\ge b \end{cases}$	1 $a=6;\ b=3.2;\ z=\mathrm{e}^{1.5ax}$; 2 $a=3;\ b=6;\ z=\mathrm{e}^{1.5ax}$; 3 $a=2.7;\ b=1.8;\ z=\mathrm{e}^{1.5ax}$
21	$y=\begin{cases} \mathrm{e}^{ax}+f\cos^5 bx & \text{за} \quad x\le a; \\ a+\cos^2 bx-\ln(fx) & \text{за} \quad a<x\le b^2; \\ \cos^2(a+bfx) & \text{за} \quad x>b^2 \end{cases}$	1 $a=0.8;\ b=2.4;\ f=\mathrm{e}^{1.5ax}$; 2 $a=1{,}2;\ b=4.2;\ f=\mathrm{e}^{2ax}$; 3 $a=3.4;\ b=8.1;\ f=\mathrm{e}^{3ax}$
22	$y=\begin{cases} a+bx+\sin^2 zx^{3.5} & \text{за} \quad x<a; \\ a+\ln\lvert ab-zx^3\rvert+\ln x & \text{за} \quad a\le x\le b^2; \\ \sqrt{\lvert a+\operatorname{ctg}(zx)\rvert}+b\sin x & \text{за} \quad x>b^2; \end{cases}$	1 $a=0.3;\ b=0.9;\ z=\sin x^2$; 2 $a=4.3;\ b=3.15;\ z=\sin x^3$; 3 $a=6.5;\ b=3.5;\ z=\sin^2 x$
23	$y=\begin{cases} \ln\lvert bzx\rvert+za^{2.5} & \text{за} \quad a^3<x\le b; \\ ax^2+bz^a+\sin^2 zx & \text{за} \quad x>b; \\ \cos(ax+b)+\ln\lvert zx\rvert & \text{за} \quad x\le a^3 \end{cases}$	1 $a=1.5;\ b=6.4;\ z=\ln\lvert bx^3+1.5\rvert$; 2 $a=1.9;\ b=8.6;\ z=\ln\lvert bx^3+3\rvert$; 3 $a=0.6;\ b=2.4;\ z=\ln\lvert bx^3+1.8\rvert$
24	$y=\begin{cases} x\mathrm{e}^x+(z+7.7abx) & \text{за} \quad x<a; \\ \operatorname{tg}(ax+z)+\cos^2 bx & \text{за} \quad a\le x\le b^2; \\ \ln(\sin^2(a+bx+zx^2)) & \text{за} \quad x>b^2 \end{cases}$	1 $a=8.7;\ b=3.7;\ z=\operatorname{tg}(bx)$; 2 $a=9.3;\ b=3.5;\ z=\operatorname{tg}(abx)$; 3 $a=2.1;\ b=5.7;\ z=\operatorname{tg}(b^2x)$
25	$y=\begin{cases} a+z\cos^2(bx)^3 & \text{за} \quad x<a; \\ a+\sin^2 b^2+\ln(zx) & \text{за} \quad a\le x\le b; \\ \sqrt[3]{0.3b+\sqrt{\lvert(a-z^2-\cos x)\rvert}} & \text{за} \quad x>b \end{cases}$	1 $a=1.5;\ b=5.7;\ z=\ln\lvert\operatorname{tg}(bx)\rvert$ 2 $a=3.7;\ b=8.4;\ z=\ln\lvert\operatorname{tg}(bx)\rvert$ 3 $a=4.4;\ b=5.6;\ z=\ln\lvert\operatorname{tg}(bx)\rvert$

Закінчення табл. 4.2

№ вар.	Функції	Варіанти параметрів
26	$y=\begin{cases} a^2x^3+\sqrt{b^4+1.7} & \text{за} \quad \lvert x\rvert<0.2; \\ \operatorname{arctg}\left(2^x-\lvert p\rvert\right) & \text{за} \quad \lvert x\rvert=0.2; \\ \sqrt[3]{\ln\lvert a\rvert+4.3}+x & \text{за} \quad \lvert x\rvert>0.2 \end{cases}$	1 $a=0.5;\ b=1.5;\ p=-4;$ 2 $a=-1;\ b=0.5;\ p=-4;$ 3 $a=-2;\ b=0;\ p=-4$
27	$y=\begin{cases} c\sin\left(b^2x\right)+b\ln\left(cx+a\right) & \text{за} \quad x<a; \\ a+\ln\left(bx\right)-\sin^2\left(a+cx\right) & \text{за} \quad a\le x<b; \\ \sqrt{\lvert\cos\left(a+bx\right)+cx^2\rvert} & \text{за} \quad x\ge b \end{cases}$	1 $a=2.2;\ b=2.4;\ c=\ln\lvert bx\rvert;$ 2 $a=1.6;\ b=1.7;\ c=\ln\lvert bx\rvert;$ 3 $a=1.3;\ b=4.2;\ c=\ln\lvert b^2x\rvert$
28	$y=\begin{cases} \sin\left(\mathrm{e}^{a+b}\right)+x^2 & \text{за} \quad a+b>x; \\ \operatorname{arctg}\left(abc\right)+\sqrt[3]{x} & \text{за} \quad a+b=x; \\ \arcsin\left(\cos^2\left(\sqrt{\lvert x\rvert}\right)\right) & \text{за} \quad a+b<x \end{cases}$	1 $a=7{,}2;\quad b=-1.3;\ c=2.5;$ 2 $a=1.47;\quad b=3.81;\quad c=2.8;$ 3 $a=4.8;\quad b=10.6;\quad c=2.7$
29	$y=\begin{cases} \operatorname{ctg}\left(x^2\mathrm{e}^{3k}\right)+\ln\lvert r+x\rvert & \text{за} \quad x=rs; \\ \sqrt[5]{x^2}+\sqrt{\lvert\arcsin k\rvert} & \text{за} \quad x>rs; \\ \operatorname{arctg}\left(kx+\operatorname{tg}\left(rs\right)\right) & \text{за} \quad x<rs \end{cases}$	1 $k=-0.3;\ r=0.85;\ s=3{,}5;$ 2 $k=0.9;\quad r=3{,}3;\quad s=1.2;$ 3 $k=-0.7;\ r=0.75;\ s=2.15$
30	$y=\begin{cases} a^3+\operatorname{arctg}\left(\sin^3 bx\right)+\cos^2 x^2 & \text{за} \quad x\le a; \\ \sqrt{\left(a+bx\right)+2}+\sin zx & \text{за} \quad a<x<\ln b; \\ \operatorname{arctg}\left(a+bx+z\right) & \text{за} \quad x\ge\ln b \end{cases}$	1 $a=1.5;\ b=5.7;\ z=\operatorname{tg}\left(bx\right);$ 2 $a=3.7;\ b=8.4;\ z=\operatorname{tg}\left(bx\right);$ 3 $a=4.4;\ b=5.6;\ z=\operatorname{tg}\left(bx\right)$

Самостійна робота № 4
Програмування розгалужень

Мета роботи: закріпити практичні навички використання операторів вибору варіантів `switch` та безумовного переходу `goto`, а також набути навиків застосування умовної операції `?:` при створюванні програмних проєктів розгалуженої структури в С++.

Теоретичні відомості

Умовна операція ?:

Формат:

<умова> ? *< операнд1>* : *< операнд2>*;

Якщо *умова* має ненульове значення, результатом буде значення *операнда1*, інакше – значення *операнда2*. Зауважимо, що обчислюється лише один з операндів, а не обидва.

Наприклад:

```
j = (i<0) ? (-i) : (i);
```

У результаті *j* набуде абсолютного значення *i*, оскільки, якщо *i* є менше нуля, то *j* присвоїться *-i*, а, якщо *i* більше або дорівнює нулю, то *j* присвоїться *i*.

Для визначення більшого з двох чисел *x* та *y* слід записати оператор:

```
max = (x > y) ? x : y;
```

Для знакопочережного ряду вираз $(-1)^i$ можна записати без використання математичної функції, суттєво зекономивши пам'ять і час виконання, за допомогою умовної операції: `(i%2) ? -1 : 1`

Умовні операції можна вкладати одна в одну. Наприклад, залежно від значення числа можна формувати коректну кінцівку рядка:

```
int k;
cout<<"Введіть значення кількості комп'ютерів -  ";
cin>> k;
int n=(k>20)? k%10 : k;
cout<<"В класі є " << k << " комп'ютер"
    << ((!n || n>4) ? "ів." : (n>1) ? "и." : ".")<<endl;
```

```
Введіть значення кількості комп'ютерів -  3
В класі є 3 комп'ютери.
```

```
Введіть значення кількості комп'ютерів -  15
В класі є 15 комп'ютерів.
```

```
Введіть значення кількості комп'ютерів -  21
В класі є 21 комп'ютер.
```

Приклад програми з розгалуженою структурою

Приклад. Для будь-якого варіанта з трьох чисел визначити і вивести найбільше з них:

1) $a = 3$; $b = 17$; $c = -4.7$;
2) $a = 8$; $b = -9.1$; $c = 89.1$;
3) $a = 13.6$; $b = 6.5$; $c = -6.2$.

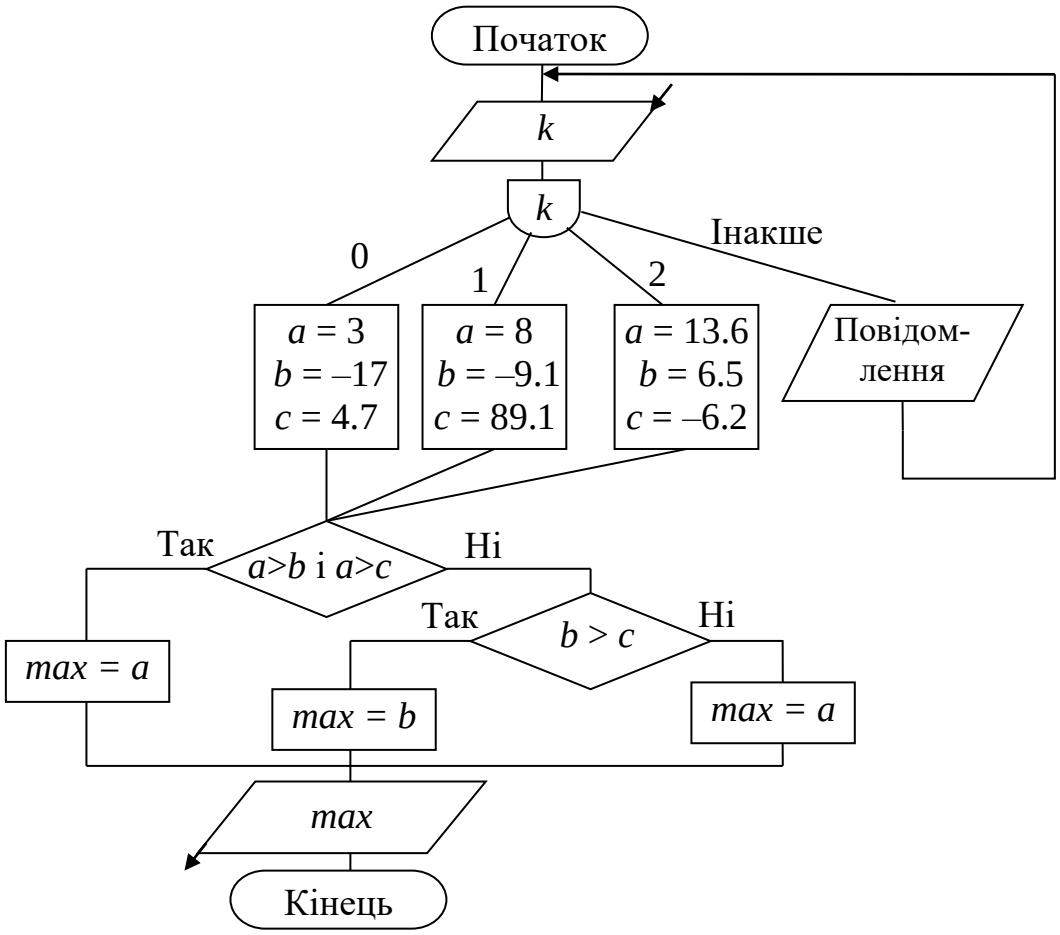
Тут конструкцію визначення максимального з трьох чисел можна організувати двома вкладеними одна в одну умовними операціями `?:`

```
max = (a>b && a>c)? a : (b>c)? b : c;
```

або двома вкладеними один в один операторами `if-else`, але така конструкція є більш громіздкою:

```
if(a>b && a>c) max=a;
else if(b>c) max=b;
     else max=c;
```

Схема алгоритму:



Текст програми:

```
#include <iostream>
#include <math.h>
using namespace std;

int main()
```

```
{
   setlocale(0,".1251");
   double a, b, c, max;
  MM:
   cout<<" Введіть номер варіанта (ціле значення) 1, 2 або 3: ";
   cin>>k;
   switch(k)
    { case 0:  a=3;  b=-17;   c=4.7;  break;
      case 1:  a=8;  b=-9.1;  c=89.1; break;
      case 2:  a=13.6; b=6.5; c=-6.2; break;
      default: { cout<<" Введіть значення 1, 2 або 3!"<<endl;
                 goto MM;
    }          }
   max = (a>b && a>c)? a : (b>c)? b : c;
   cout<<"max= "<< max << endl;
   system("pause>>void");
   return 0;
 }
```

Результати роботи:

```
Введіть номер варіанта (ціле значення) 1, 2 або 3: 1
max= 4.7
```

Контрольні запитання та завдання для самоконтролю

1) Яке призначення умовної операції (?:)?
2) Яким буде значення змінної `f` після виконання операторів:

```
int f = 1, n = 3, i = 2;
M1: if (i > n)  goto PP;
    f = f * i; i++; goto M1;
PP: ;
```

3) У наведеному фрагменті програми:

```
int  nom = pow(2, 3);
switch (nom)
{ case  2 : y=d;     break;
  case  8 : y=d*exp(x); break;
  case 10 : y=d*x;   break;
}
```

оператор `switch` обчислюватиме вираз … (записати увесь вираз).

Завдання до самостійної роботи

1) Дати відповіді на контрольні запитання.

2) Скласти схеме алгоритмів і написати програми мовою С++ із застосуванням оператора варіантів `switch` для розв'язання завдань, поданих у табл. 4.3 відповідно до індивідуального варіанта.

3) Створити на комп'ютері програмні проєкти мовою С++ для реалізації написаних програм. Занести результати обчислень до протоколу.

Таблиця 4.3

Індивідуальні завдання

№ вар.	Завдання	Варіанти параметрів
1	Визначити, чи лежить точка А в області, обмеженій параболою $y = 2 - x^2$ та віссю абсцис. Відповідь вивести у вигляді повідомлення	1 $x = 3.5$; $y = 7.2$; 2 $x = -0.5$; $y = 1.2$; 3 $x = 0.72$; $y = -3.12$
2	З трьох чисел визначити і вивести на екран середнє за значенням з них	1 $a = 3$; $b = 3.5$; $c = -2.1$; 2 $a = 2.1$; $b = -6.55$; $c = 0.1$; 3 $a = -9$; $b = -3.7$; $c = -0.1$
3	Для точок з координатами x та y визначити, чи лежать вони всередині кола з радіусом R, якщо центром кола є початок координат	1 $x = 3$; $y = -7$; $R = 5$; 2 $x = 12$; $y = 11$; $R = 16$; 3 $x = -9$; $y = 6$; $R = 11$
4	Задано значення трьох сторін трикутника – a, b та c. Визначити, чи є цей трикутник прямокутним	1 $a = 3$; $b = 3.5$; $c = -2.1$; 2 $a = 2.1$; $b = -6,55$; $c = 0.1$; 3 $a = -9$; $b = -3,7$; $c = -0.1$
6	Задано значення трьох чисел – A, B, C. Подвоїти ті числа, для яких $A + B + C > 0$, а якщо це не так, – замінити їх на нулі	1 $A = -3$; $B = 3,5$; $C = 0,1$; 2 $A = 58$; $B = 27$; $C = -87$; 3 $A = -8$; $B = -35$; $C = 42$
7	Для координат точок $A(x_0, y_0)$ та $B(x_1, y_1)$ визначити, яка з точок – A чи B – найменш віддалена від початку координат ($O(0,0)$)	1 $x_0 = 2$; $y_0 = 2$; $x_1 = -4$; $y_1 = 0$; 2 $x_0 = 8$; $y_0 = 9$; $x_1 = 12$; $y_1 = 1$; 3 $x_0 = -3$; $y_0 = 0.9$; $x_1 = 2$; $y_1 = 3$
8	Для трикутників зі значеннями сторін – a, b та c визначити, чи є вони рівнобедреними	1 $a = 3$; $b = 3.5$; $c = 1.1$; 2 $a = 3$; $b = 6.55$; $c = 6.55$; 3 $a = 0.9$; $b = 0.9$; $c = 0.9$
9	Для трьох цілих чисел (a, b, c) визначити, чи є вони трійкою Піфагора ($c^2 = a^2 + b^2$)	1 $a = 3$; $b = 5$; $c = 4$; 2 $a = 3$; $b = 8$; $c = 11$; 3 $a = 13$; $b = 5$; $c = 12$
10	Для трьох точок – $A_1(x_1, y_1)$, $A_2(x_2, y_2)$ та $A_3(x_3, y_3)$ – визначити, чи лежать ці точки на одній прямій	1 $x_1 = 2$; $y_1 = 2$; $x_2 = 4$; $y_2 = 0$; $x_3 = -2$; $y_3 = 6$; 2 $x_1 = 8$; $y_1 = 9$; $x_2 = 4$; $y_2 = 0$; $x_3 = 5$; $y_3 = 1$
11	Перевірити числа A та B і змінити їхній знак на протилежний, якщо вони мають різні знаки, а якщо це не так, – замінити їх на нулі	1 $A = -3$; $B = 3.5$; 2 $A = 58$; $B = 27$; 3 $A = -8$; $B = -35$
12	З трьох чисел визначити і вивести на екран найменше з них	1 $a = 23$; $b = 17$; $c = 47$; 2 $a = 9$; $b = -8.1$; $c = 9.1$; 3 $a = 36$; $b = 65$; $c = 62$
13	Для трьох чисел – x, y, z – визначити середнє арифметичне та вивести на екран ті з чисел, які за модулем є більші за середнє арифметичне	1 $x = 3.2$; $y = -7$; $z = 0.5$; 2 $x = 2.3$; $y = 3$; $z = 2.5$; 3 $x = 23$; $y = -34$; $z = 89.5$

Продовження табл. 4.3

№ вар.	Завдання	Варіанти параметрів
14	Для точки з координатами x та y визначити, чи лежить вона на кривій $f(x)=\begin{cases} x^3 & \text{за} \quad |x|>1; \\ \sqrt{2-x^2} & \text{за} \quad |x|\le 1. \end{cases}$. Похибка складає eps = 10^{-3}, тобто $\lvert f(x)-y\rvert <$ eps	1 x = -3; y = 27; 2 x = -2; y = 56; 3 x = -1; y = 1; 4 x = -3; y = -27
15	Для точки з координатами x та y визначити, чи лежить вона в четвертій координатній чверті	1 x = -2.8; y = 0.7; 2 x = 0; y = -9.5; 3 x = 2; y = -51
16	Для трикутника зі сторонами a, b та c, найменша зі сторін якого є стороною квадрата, визначити, площа якої фігури є більша	1 a = 3; b = 5; c = 4; 2 a = 13; b = 8; c = 11; 3 a = 10; b = 5; c = 12
17	З трьох чисел – x, y, z – визначити і вивести на екран ті з цих чисел, які є менші за їхнє середнє арифметичне	1 x = -2; y = 1.2; z = 9.5; 2 x = 0.5; y = 2; z = -0.15; 3 x = 0.4; y = 2.2; z = 9.5
18	З трьох чисел – x, y та z – визначити і вивести на екран ті з цих чисел, які за модулем є більші за число π	1 x = -7.2; y = 3.14; z =-2.5; 2 x = -4; y = -3; z = 9.15; 3 x = 3.14; y =-3.4; z = 0.59
19	Для трьох цілих чисел (довжин сторін трикутника) визначити, чи можна побудувати трикутник з цими сторонами	1 a = 8; b = 13.5; c = 1.1; 2 a = 3; b = 3.56; c = 0.55; 3 a = 1.9; b = 0.9; c = 0.9
20	Для точок $A(x_0, y_0)$ та $B(x_1, y_1)$ визначити, яка з них – А чи В – є найменш віддалена від початку координат ($O(0,0)$)	1 x_0 = 3; y_0 = 3; x_1 = -6; y_1 = 0; 2 x_0 = 8; y_0 = 9; x_1 = 12; y_1 = 1; 3 x_0 = 3; y_0 = 0.9; x_1 = 2; y_1 = 3
21	З трьох чисел – a, b, c – додатні піднести до квадрата, а від'ємні – залишити без змін	1 a = 0; b = 1.5; c = -31.1; 2 a = 2; b = -1.56; c = 2.55; 3 a = -1.9; b = 2.9; c = -2.9
22	З трьох цілих чисел – a, b, c – знайти і вивести на екран непарні числа	1 a = 2; b = 9; c = 474; 2 a = 3; b = 0; c = 27; 3 a = 4; b = 11; c = 30
23	Для трьох чисел – a, b, c – визначити кількість коренів рівняння $ax^2 + bx + c = 0$	1 a = 1; b = 8; c = 16; 2 a = -8; b = 29.7; c = 0.11; 3 a = 2.5; b = 5; c = 3
24	Для точок з координатами x та y визначити, чи лежать вони за межами кола з радіусом R, якщо центром кола є початок координат	1 x = 78; y = -71; R = 85; 2 x = 2; y = 11; R = 13; 3 x = -7; y = 6; R = 11
25	З трьох цілих чисел – a, b, c – знайти і вивести на екран числа, які діляться на 3 без остачі	1 a = 2; b = 9; c = 474; 2 a = 3; b = 0; c = 27; 3 a = 4; b = 10; c = 30

Закінчення табл. 4.3

№ вар.	Завдання	Варіанти параметрів
26	З трьох цілих чисел – *a*, *b*, *c* – знайти і вивести на екран числа, які завершуються числом 5	1 *a* = 550; *b* = 175; *c* = -251; 2 *a* = 872; *b* = -56; *c* = -255; 3 *a* = -1995; *b* = 259; *c* = 89
27	З трьох чисел знайти і вивести на екран середнє за абсолютним значенням з них	1 *a* = 3; *b* = -3.5; *c* = -2.1; 2 *a* = 2.1; *b* = -6.55; *c* = 0.1; 3 *a* = -9; *b* = -3.7; *c* = 11.1
28	Для точки з координатами *x* та *y* визначити, чи лежить вона в першій координатній чверті	1 *x* = 12.8; *y* = 0.7; 2 *x* = 0; *y* = -9.5; 3 *x* = -12; *y* = -51
29	З трьох цілих чисел – *a*, *b*, *c* – знайти і вивести на екран парні числа.	1 *a* = 2; *b* = 9; *c* = 474; 2 *a* = 3; *b* = 0; *c* = 27; 3 *a* = 4; *b* = 10; *c* = 30
30	Визначити для трьох варіантів координат точок з координатами *x* та *y* квадрант, в якому вони розміщені	1 *x* = 0; *y* = -2.7; 2 *x* = -2.43; *y* = -2.2; 3 *x* = 0.13; *y* = 0.74

Лабораторна робота № 5
Програмування циклів. Оператор циклу з параметром for

Мета роботи: набути практичних навиків організації циклічних обчислень у С++ з використанням оператора циклу з параметром **for**.

Теоретичні відомості

1. Циклічні алгоритми

Обчислювальний процес називається *циклічним*, якщо він неодноразово повторюється, доки виконується певна задана умова. Блок повторюваних операторів називають *тілом* циклу. Існують три різновиди операторів циклу:

- оператор циклу `for`;
- оператор циклу з передумовою `while`;
- оператор циклу з післяумовою `do-while`.

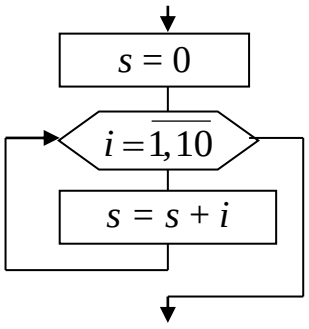
2. Оператор циклу з параметром `for`

Синтаксис оператора:

for (<*ініціалізація*>; <*умова*>; <*модифікації*>) <*тіло циклу*>;

Конструктивно цей оператор складається з трьох основних блоків, розміщених у круглих дужках і відокремлених один від одного крапкою з комою (;), та команд (тіла циклу), які мають багаторазово виконуватись у цьому циклі. На початку виконання оператора циклу одноразово у блоці *ініціалізації* задаються початкові значення змінних (параметрів), які керують циклом. Потім перевіряється *умова* і, якщо вона виконується (`true` або має ненульове значення), то виконується *команда* (чи група команд в операторних дужках {}) тіла циклу. *Модифікації* змінюють параметри циклу і, в разі істинності умови, виконування циклу триває. Якщо *умова* не виконується (`false` або дорівнює нулю), відбувається вихід із циклу, і керування передається на оператор, який іде за оператором `for`. Суттєвим є те, що перевірка умови виконується на початку циклу. Це означає, що тіло циклу може не виконуватись жодного разу, якщо *умова* спочатку є хибна. Кожне повторення (крок) циклу називається *ітерацією*.

Простий приклад для обчислення суми $S = \sum_{i=1}^{10} i$ проілюструє використання оператора `for`:




```
int  s = 0;
for(int i = 1; i <= 10; i++)  s += i;
```

Цей оператор можна прочитати так: “виконати команду `s += i` 10 разів (для значень `i` від 1 до 10 включно, де `i`

на кожній ітерації збільшується на 1)". У наведеному прикладі є два присвоєння початкових значень: `s=0` та `i=1`, умова продовження циклу: (`i<=10`) і змінення параметра: `i++`. Тілом циклу є команда `s += i`.

Порядок виконання комп'ютером цього циклу є такий:

1) присвоюються початкові значення (`s=0`, `i=1`);
2) перевіряється умова (`i<=10`);
3) якщо умова є істинна (`true`), виконується команда (чи команди) тіла циклу: до суми, обчисленої на попередній ітерації, додається нове число;
4) параметр циклу збільшується на 1.

Далі повертаємось до пункту 2. Якщо умову у пункті 2 не буде виконано (`false`), відбудеться вихід із циклу.

В операторі можливі конструкції, коли є відсутній той чи інший блок: *ініціалізація* може бути відсутня, якщо початкове значення задати попередньо; *умова* – якщо припускається, що умова є завжди істинна, тобто слід неодмінно виконувати тіло циклу, доки не зустрінеться оператор `break`; а *модифікації* – якщо приріст параметра здійснювати в тілі циклу або взагалі це є непотрібне. Тоді сам вираз блока пропускається, але крапка з комою (;) неодмінно має залишитись. Можливою є наявність *порожнього* оператора (оператор є відсутній) у тілі циклу. Наприклад, суму з попереднього прикладу можна обчислити в інший спосіб:

```
for(int s = 0,  i = 1;  i <= 10;  s += i++);
```

У цьому прикладі є відсутній *оператор*, а блок *ініціалізації* вмістить два оператори, розділених операцією "кома", які задають початкові значення змінних `s` та `i`.

Розглянемо використання циклу `for` для обчислення факторіала $F = n!$ (нагадаємо, що факторіал обчислюється за формулою $n! = 1 \cdot 2 \cdot 3 \ldots (n-2)(n-1)n$, наприклад: $4! = 1 \cdot 2 \cdot 3 \cdot 4 = 24$). Наведемо три аналогічні за дією форми запису оператора `for`:

```
1) int F=1, n=5; for(int i=1; i<=n; i++) F *= i;
2) int F, i, n=5; for(F=1, i=1; i<=n; F *= i++);
3) int F=1, i=1, n=5; for( ; i<=n; ) F *= i++;
```

Для завчасного початку чергової ітерації циклу можна використати оператор переходу до наступної ітерації `continue`, наприклад:

```
for(i = 0; i < 20; i++)
 { if(a[i] == 0)  continue;
   a[i] = 1/a[i]; }
```

Для дострокового виходу з циклу застосовують оператори `break` (вихід з конструкції), `goto` (безумовний перехід) і `return` (вихід з поточної функції).

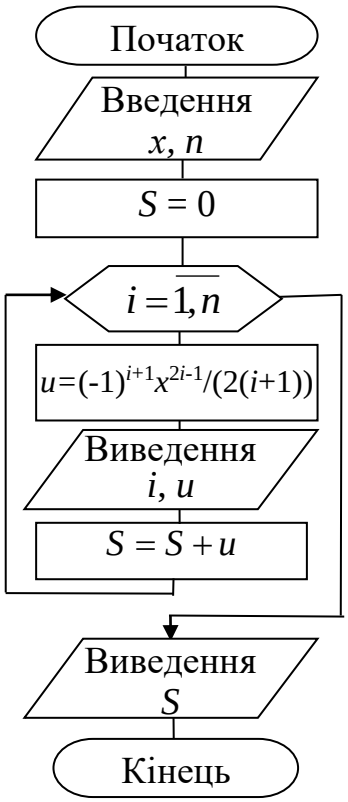
Розглянемо кілька прикладів розв'язування задач, в яких є доцільне використання оператора циклу `for`. Для усіх програм розроблено алгоритмічні схеми (блок-схеми), які визначають порядок виконування дій, і наведено форми з результатами обчислень.

Приклади програм з циклічною структурою, зорганізованих за допомогою оператора for

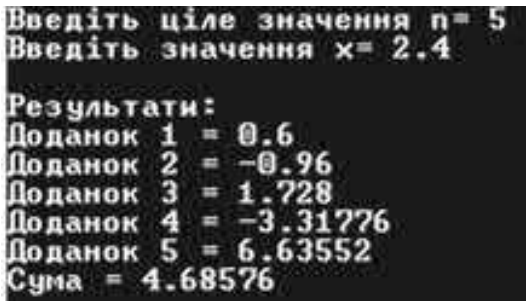
Приклад 1. Обчислення суми ряду. Ввести ціле число n і дійсне число x, обчислити $s = \frac{x}{4} - \frac{x^3}{6} + \frac{x^5}{8} - \ldots = \sum_{i=1}^{n} \frac{(-1)^{i+1} x^{2i-1}}{2(i+1)}$.

Текст програми та блок-схема:

```
#include <iostream>
#include <math.h>
using namespace std;
int main()
{
 setlocale(0,".1251");
 double x, u, s=0;
 int i, n;
 cout<<"Введіть ціле значення n= ";
 cin>> n;
 cout<<"Введіть значення x= ";
 cin>> x;
 cout<< "\nРезультати:\n";
 for(i=1; i<=n; i++)
 { u = pow(-1,i+1.0)*pow(x,2*i-1)/(2*(i+1));
   cout<< "Доданок " << i << " = " << u << endl;
   s+=u;
 }
 cout<< "Сума = " << s << endl;
 system("pause>>void");
 return 0;
}
```

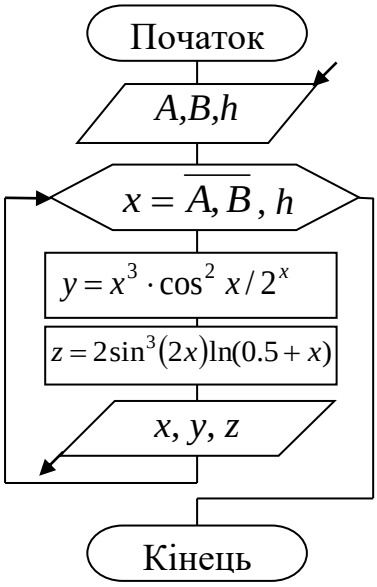


Результати роботи:



Приклад 2. Дослідження функцій на певному проміжку (табулювання). Скласти схему алгоритму і програму табулювання функцій (дослідження на певному проміжку) $y(x) = \frac{x^3 \cdot \cos^2 x}{2^x}$ та $z(x) = 2\sin^3(2x)\ln(0.5 + x)$, змінюючи x на проміжку [1, 10] з кроком $h = 0{,}5$.

Текст програми та блок-схема:




```
#include <iostream>
#include <math.h>
using namespace std;
int main()
{
  setlocale(0,".1251");
  double x, y, z, A, B, h;
  cout<<"Введіть початкове значення X= ";
  cin >> A;
  cout<<"Введіть кінцеве значення X= ";
  cin >> B;
  cout<<"Введіть крок для X= ";
  cin >> h;
  cout<<"\n  X \t  Y(X) \t  Z(X) \n";
  for(x=A; x <=B+0.1*h; x+=h)
  {
    y = x*x*x*pow(cos(x),2)/pow(2,x);
    z = 2*pow(sin(2*x),3)*log(0.5+x);
    cout<<"x = "<< x << " \t "<< y << " \t "<< z << endl;
  }
}
```

Результати роботи:

```
Введіть початкове значення X= 1
Введіть кінцеве значення X= 10
Введіть крок для X= 0.5
  X           Y(X)           Z(X)
x = 1       0.145963       0.609679
x = 1.5     0.00597069     0.00389602
x = 2       0.346356       -0.794348
x = 2.5     1.77282        -1.93744
x = 3       3.30779        -0.0546576
x = 3.5     3.32334        0.78624
x = 4       1.709          2.91314
x = 4.5     0.178948       0.225305
x = 5       0.314313       -0.548956
x = 5.5     1.84634        -3.58341
x = 6       3.1115         -0.578331
x = 6.5     2.89379        0.288681
x = 7       1.52305        3.91732
x = 7.5     0.28003        1.14365
x = 8       0.0423405      -0.10214
x = 8.5     0.614769       -3.90493
x = 9       1.182          -1.90704
x = 9.5     1.17741        0.0155043
x = 10      0.68754        3.57838
```

Контрольні запитання та завдання для самоконтролю

1) Який процес називають циклічним?
2) Які оператори циклу використовуються в мові C++?
3) Скільки разів виконуватиметься оператор усередині циклу, тобто вкажіть значення s:
```
for(int k=-1, s=0; k<=5; k++) s++;
```
4) Назвіть помилки в таких фрагментах програм:

а)
```
int k, m=2, n=3;
  for(k=1; k<=n; k++)n=n+m;
```

б)
```
int n=-7, m=2;
  for(int k=n; k<=m; k--)k++;
```

5) Вкажіть значення m після виконання фрагментів програми:

а)
```
int k, m=1;
  for(k=1; k<=5; k++)m++;
```

б)
```
int m=1, n=5;
  for(int k=n; k>=1; k--)m*=k;
```

Лабораторне завдання

1) У протоколі лабораторної роботи дати відповіді на контрольні запитання.

2) У протоколі лабораторної роботи скласти схему алгоритму і написати програму мовою C++ із застосуванням оператора `for` для розв'язання завдань, поданих у табл. 5.1 відповідно до індивідуального варіанта (див. приклад програми 1).

3) У протоколі лабораторної роботи скласти схему алгоритму і програму табулювання (дослідження) функцій $y = f(x)$ та $z = f(x)$, змінюючи x на заданому проміжку із заданим кроком, які вибрати з табл. 5.2 відповідно до індивідуального варіанта (див. приклад програми 2).

4) Створити на комп'ютері програмні проєкти мовою C++ для реалізації написаних програм. Занести результати обчислень до протоколу.

Таблиця 5.1

Індивідуальні завдання "Обчислення суми ряду"

№ вар.	Варіанти завдань
1	Ввести натуральне число n і дійсне число x, обчислити $s = 1 + x + \frac{x^2}{2} + \frac{x^3}{3} + \ldots = \sum_{i=1}^{n} \frac{x^i}{i}$
2	Ввести натуральне число n та обчислити $s = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \ldots = \sum_{i=1}^{n} \frac{(-1)^{i+1}}{2i-1}$
3	Ввести два натуральні числа n та m ($n < m$), вивести всі кратні 4 числа від n до m та обчислити їхню суму
4	Ввести ціле число n і дійсне x, обчислити $s = \cos(x) + \frac{\cos(2x)}{2} + \frac{\cos(3x)}{3} + \ldots = \sum_{i=1}^{n} \frac{\cos(ix)}{i}$
5	Ввести ціле число n і дійсне число a, обчислити $s = 1 - a + a^2 - a^3 + \ldots = \sum_{i=0}^{n} (-a)^i$

Продовження табл. 5.1

№ вар.	Варіанти завдань
6	Ввести натуральне число *n* та обчислити $s = 1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + ... = \sum_{i=1}^{n} \frac{1}{i}$
7	Ввести двоцифрове число *N*, вивести всі непарні числа від 1 до *N* та обчислити їхню суму
8	Ввести два натуральні числа *n* та *m* (*n* < *m*), обчислити $s = \sum_{i=n}^{m} i$
9	Ввести ціле число *n* і дійсне *x*, обчислити $s = -\frac{1}{x} + \frac{3}{x^2} - \frac{5}{x^3} - ... = \sum_{i=1}^{n} \frac{(-1)^i (2i-1)}{x^i}$
10	Ввести двоцифрове число *N*, вивести всі кратні 3 числа від 1 до *N* та обчислити їхню суму
11	Ввести ціле число *n* і дійсне число *x*, обчислити $s = -x + \frac{x^3}{2} - \frac{x^5}{3} + ... = \sum_{i=1}^{n} \frac{(-1)^i x^{2i-1}}{i}$
12	Ввести ціле число *n* і дійсне число *x*, обчислити $s = \cos(x) + \cos(3x^3) + ... = \sum_{i=1}^{n} \cos((2i-1)x^{2i-1})$
13	Ввести ціле число *n* і дійсне число *x*, обчислити $s = \frac{1}{2} + \frac{x^2}{5} + \frac{x^3}{8} + ... = \sum_{i=1}^{n} \frac{x^{i-1}}{3i-1}$
14	Ввести два натуральні числа *n* та *m* (*n* < *m*), вивести всі парні числа від *n* до *m* та обчислити їхню суму
15	Ввести ціле число *n* і дійсне *x*, обчислити $s = 1 - \frac{2}{x^2} + \frac{3}{x^4} - ... = \sum_{i=1}^{n} \frac{(-1)^{i+1} i}{x^{2i}}$
16	Ввести ціле число *n* і дійсне *x*, обчислити $s = \sin(x) + \frac{\sin^2(x)}{4} + \frac{\sin^3(x)}{7} + ... = \sum_{i=1}^{n} \frac{\sin^i(x)}{3i-2}$
17	Ввести ціле число *n* і дійсне число *x*, обчислити $s = x - \frac{x^3}{3} + \frac{x^5}{5} - ... = \sum_{i=0}^{n} \frac{(-1)^{i-1} x^{2i+1}}{2i+1}$
18	Ввести двоцифрове число *N*, вивести всі парні числа від 1 до *N* та обчислити їхню суму
19	Ввести ціле число *n* і дійсне число *x*, обчислити $s = (x+1) + \frac{(x+2)^3}{4} + \frac{(x+3)^5}{9} - ... = \sum_{i=1}^{n} \frac{(x+i)^{2i-1}}{i^2}$
20	Ввести двоцифрове число *N*, вивести всі кратні 3 числа від 1 до *N* та обчислити їхню суму
21	Ввести ціле число *n* і дійсне *x*, обчислити $s = \sin(1-x) + \frac{\sin(2-x)}{4} + \frac{\sin(3-x)}{9} + ... = \sum_{i=1}^{n} \frac{\sin(i-x)}{i^2}$

Закінчення табл. 5.1

№ вар.	Варіанти завдань
22	Ввести ціле число n і дійсне число x, обчислити $s = \frac{1}{(x+1)} - \frac{3}{(x+2)^2} + \frac{5}{(x+3)^3} - ... = \sum_{i=1}^{n} (-1)^{i+1} \frac{2i-1}{(x+i)^i}$
23	Ввести ціле число n і дійсне число x, обчислити $s = -1 + \frac{x}{2} + \frac{x^2}{7} + \frac{x^3}{14} ... = \sum_{i=1}^{n} \frac{x^{i-1}}{i^2 - 2}$
24	Ввести натуральне число N до 10, вивести всі степені числа 2 від 1 до N та обчислити їхню суму
25	Ввести ціле число n і дійсне x, обчислити $s = \frac{\cos(2x)}{3} + \frac{\cos(4x)}{15} + \frac{\cos(6x)}{35} ... = \sum_{i=1}^{n} \frac{\cos(2ix)}{(2i-1)(2i+1)}$.
26	Ввести ціле число n і дійсне x, обчислити $s = \frac{4}{x} + \frac{9}{2x^3} + \frac{16}{3x^5} + ... = \sum_{i=1}^{n} \frac{(i+1)^2}{ix^{2i-1}}$
27	Ввести натуральне число n (до 10) і дійсне x, вивести всі степені числа x від 1 до n та обчислити суму цих чисел
28	Ввести ціле число n і дійсне x, обчислити $s = \frac{1}{2} + \frac{\sin(x)}{3} + \frac{\sin^2(x)}{4} + ... = \sum_{i=0}^{n} \frac{\sin^i(x)}{i+2}$
29	Ввести ціле число n і дійсне x, обчислити $s = \frac{x}{2} + \frac{x^3}{12} + \frac{x^5}{30} ... = \sum_{i=1}^{n} \frac{x^{2i-1}}{2i(2i-1)}$
30	Ввести ціле число n і дійсне x, обчислити $s = 2 + \frac{3x}{9} + \frac{4x^2}{25} + \frac{5x^3}{49} ... = \sum_{i=1}^{n} \frac{(i+1)x^{i-1}}{(2i-1)^2}$

Таблиця 5.2

Індивідуальні завдання
"Дослідження функцій на певному проміжку (табулювання)"

№ вар.	Функція $y = f(x)$	Функція $z = f(x)$	Проміжок дослідження
1	$\sin(x)/x^2$	$\cos(x)/x$	$x \in [0{,}5;\ 11]$, $h = 0{,}3$
2	$\operatorname{arctg}(x + 3{,}1)$	e^x	$x \in [-6;\ 1]$, $h = 0{,}2$
3	$e^{3(x-0.6)}$	$\arcsin(x)$	$x \in [-1;\ 1]$, $h = 0{,}05$
4	$\sqrt{\lvert\sin(x + \pi/4)\rvert}$	$\sin x^2 + \cos x$	$x \in [-4;\ 10]$, $h = 0{,}4$
5	$\operatorname{tg}\sqrt{x}$	$x/(x-3)^2$	$x \in [4{,}5;\ 18{,}5]$, $h = 0{,}4$
6	$1/e^x$	$\lg(x/2 + 0{,}1)$	$x \in [0;\ 7]$, $h = 0{,}2$
7	$\operatorname{tg}(x/3)\cdot\sin(x - 1{,}2)$	$2{,}5\sin(x/2)$	$x \in [-2;\ 5]$, $h = 0{,}2$
8	$1/x$	$(x/3)^2$	$x \in [0{,}5;\ 4]$, $h = 0{,}1$
9	$\cos(1{,}5x)\cdot\lg(2{,}5x)$	$e^{\frac{1}{\sqrt{x}}} \sin(x)$	$x \in [3{,}5;\ 10{,}5]$, $h = 0{,}2$

Закінчення табл. 5.2

№ вар.	Функція y = f(x)	Функція z = f(x)	Проміжок дослідження
10	$\cos(x)/x$	$\cos(x/2)$	$x\in[0{,}3;\ 7{,}3]$, $h = 0{,}2$
11	e^x	$1{,}5\cos(x - \pi/4\cdot e^x)$	$x\in[-6;\ 1]$, $h = 0{,}2$
12	$\arcsin(x)$	$\cos(1/(x + \pi/3))$	$x\in[-1;\ 1]$, $h = 0{,}05$
13	$\sin^2(x)\cdot\cos(x - \pi)$	$\cos(x)/x$	$x\in[0{,}5;\ 11]$, $h = 0{,}3$
14	$\sin x^2+\cos x$	$\dfrac{\sin x}{\lg(x^2 + 2)}$	$x\in[-4;\ 10]$, $h = 0{,}4$
15	$x/(x - 3)^2$	$\lvert\cos(x/3)\rvert$	$x\in[4{,}5;\ 18{,}5]$, $h = 0{,}4$
16	$\lg(x/2 + 0{,}1)$	$\cos((x + 2\pi)e^x)$	$x\in[0;\ 7]$, $h = 0{,}2$
17	$2{,}5\sin(x/2)$	$\sin(x)/\ln(x + 4)$	$x\in[-2;\ 5]$, $h = 0{,}2$
18	$(x/3)^2$	$\cos(x + \pi/3) + 1{,}8$	$x\in[0{,}5;\ 4]$, $h = 0{,}1$
19	$e^{\frac{1}{\sqrt{x}}}\sin(x)$	$\operatorname{tg}\sqrt{x}\cdot\sin\left(x - \dfrac{\pi}{2}\right)$	$x\in[3{,}5;\ 10{,}5]$, $h = 0{,}2$
20	$\cos(x/2)$	$\sin(x + \pi/2)\cdot\cos(1/x)$	$x\in[0{,}3;\ 7{,}3]$, $h = 0{,}2$
21	$\sin^2(x)\cdot\cos(x - \pi)$	$\sin(x)/x^2$	$x\in[0{,}5;\ 11]$, $h = 0{,}3$
22	$1{,}5\cos(x - \pi/4\cdot e^x)$	$\operatorname{arctg}(x + 3{,}1)$	$x\in[-6;\ 1]$, $h = 0{,}2$
23	$\cos(1/(x + \pi/3))$	$e^{3(x-0.6)}$	$x\in[-1;\ 1]$, $h = 0{,}05$
24	$\dfrac{\sin x}{\lg(x^2 + 2)}$	$\sqrt{\lvert\sin(x + \pi/2)\rvert}$	$x\in[-4;\ 10]$, $h = 0{,}4$
25	$\lvert\cos(x/3)\rvert$	$\operatorname{tg}\sqrt{x}$	$x\in[4{,}5;\ 18{,}5]$, $h = 0{,}4$
26	$\cos((x + 2\pi)e^x)$	$1/e^x$	$x\in[0;\ 7]$, $h = 0{,}2$
27	$\operatorname{tg}(x/3)\cdot\sin(x - 1{,}2)$	$\sin(x)/\ln(x + 4)$	$x\in[-2;\ 5]$, $h = 0{,}2$
28	$1/x$	$\cos(x + \pi/3) + 1{,}8$	$x\in[0{,}5;\ 4]$, $h = 0{,}1$
29	$\cos(1{,}5x)\cdot\lg(2{,}5x)$	$\operatorname{tg}\sqrt{x}\cdot\sin(x - \pi/2)$	$x\in[3{,}5;\ 10{,}5]$, $h = 0{,}2$
30	$\cos(x)/x$	$\sin(x + \pi/2)\cdot\cos(1/x)$	$x\in[0{,}3;\ 7{,}3]$, $h = 0{,}2$

Самостійна робота № 5

Циклічне опрацювання послідовностей чисел

Мета роботи: набути практичних навиків програмного опрацювання числових послідовностей за допомогою циклів.

Теоретичні відомості

Існує коло задач, в яких необхідно певним чином опрацьовувати задану числову послідовність, причому для обчислення результату досить переглянути послідовність один раз. Наприклад, щоб обчислити середнє арифметичне заданої послідовності чисел, можна підсумовування чисел і підрахунок їхньої кількості поєднати з введенням чисел. Тоді не потрібно буде зберігати всю послідовність в пам'яті комп'ютера (у вигляді масиву), достатньо мати одну скалярну змінну цілого або дійсного типу і по черзі присвоювати їй значення, які вводяться.

Числова послідовність може задаватися зі зазначенням кількості чисел або мати якусь ознаку кінця.

Приклади програм

Приклад 1. Ввести шість дійсних чисел та визначити найбільше з них.

Розв'язок. Алгоритм пошуку максимального числа послідовності:

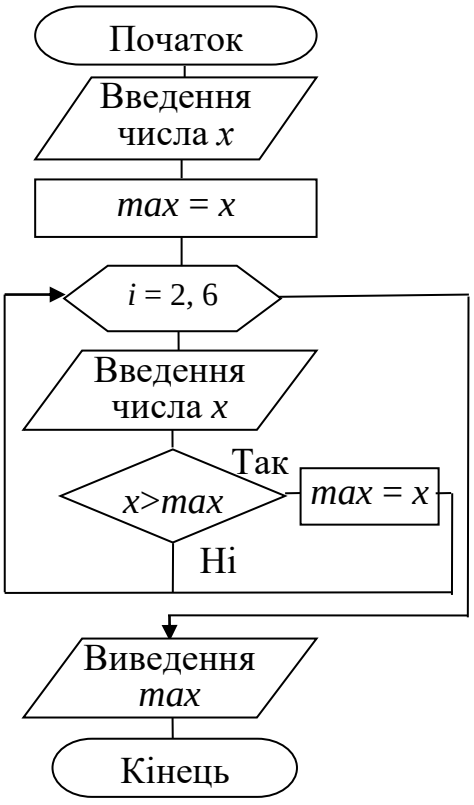
1) ввести перше число x;

2) вважати, що воно є максимальним:

```
max = x;
```

3) у циклі почергово вводити решту чисел. Кожне з введених чисел порівнювати зі значенням max і, якщо число x буде більшим за max, запам'ятати його значення як max:

```
if (x>max) max=x;
```



Текст програми та блок-схема:

```
#include <iostream>
using namespace std;

int main()
{
 setlocale(0,".1251");
 int i;  double x, max;
 cout << "Ввести 1 число: ";
 cin >> x;
```

```
 max=x;
 for (i=2; i<=6; i++)
 { cout << "Ввести "<< i <<" число ";
   cin >> x;
   if (x>max) max=x;
 }
 cout << "Найбільше число: " << max << endl;
 system("pause>>void");
 return 0;
}
```

Результати виконання програми:

```
Ввести 1 число: 5.5
Ввести 2 число: 0.2
Ввести 3 число: -7
Ввести 4 число: 3.1
Ввести 5 число: 15.2
Ввести 6 число: -1.4
Найбільше число: 15.2
```

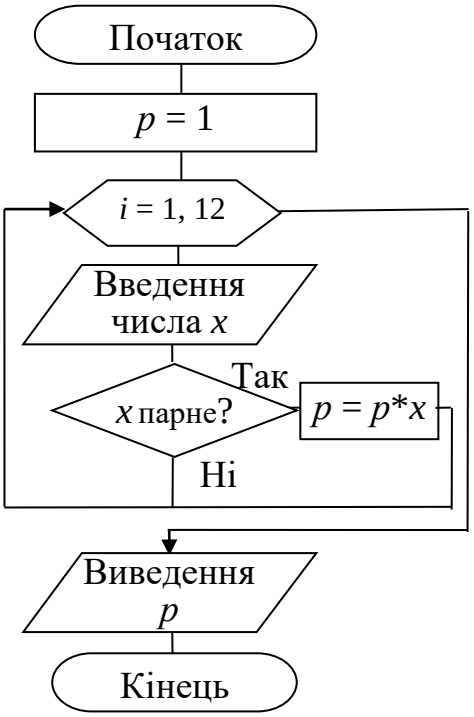
Приклад 2. Ввести 12 цілих чисел та обчислити добуток парних елементів з цієї числової послідовності.

Розв'язок. У циклі, який повторюватиметься 12 разів, виконувати такі дії:

– виводити запрошення для введення наступного числа;

– вводити нове число;

– перевіряти введене число на парність і, якщо це так, то число перемножити на добуток.



Текст програми та блок-схема:

```
#include <iostream>
using namespace std;
int main()
{
 setlocale(0,".1251");
 int i, x, p=1;                    // Спочатку добуток дорівнює 1.
 for(i=1; i<=12; i++)              // У циклі 12 разів повторити дії:
 { cout << "Ввести " << i << " число:"; // вивести запрошення,
   cin >> x;                       // ввести число, i,
   if(x%2==0 && x!=0) p *= x;      // якщо число парне, перемножити його.
 }
 cout << "Добуток парних чисел: " << p << endl;
 system("pause>>void");
 return 0;
}
```

Результат виконання програми:

```
Ввести 1 число: 3
Ввести 2 число: 1
Ввести 3 число: -7
Ввести 4 число: 6
Ввести 5 число: 3
Ввести 6 число: -2
Ввести 7 число: 4
Ввести 8 число: 7
Ввести 9 число: 11
Ввести 10 число: -4
Ввести 11 число: -5
Ввести 12 число: 3
Добуток парних чисел: 192
```

У цієї програми є один суттєвий недолік. Якщо парних чисел немає, то результатом обчислення добутку буде виведено 1. Замість цього бажано вивести повідомлення, що парних чисел введено не було. Для цього слід організувати підрахунок кількості введених парних чисел. Якщо після циклу ця кількість становитиме 0, то слід вивести повідомлення. Для обчислення кількості слід оголосити окрему цілу змінну, значення якої спочатку є 0 (парних чисел ще нема). Якщо введене число є парне, кількість збільшиться на 1.

Текст програми:

```
#include <iostream>
using namespace std;
int main()
{
 setlocale(0,".1251");
 int i, x, p=1, k=0;
 for(i=1; i<=12; i++)
 {
   cout << "Ввести " << i << " число: ";
   cin >> x;
   if(x%2==0 && x!=0)          // Якщо число парне,
   { p *= x;                   // обчислюватиметься добуток
     k++;                      // і кількість таких чисел.
   }
 }
if(k > 0) // Якщо кількість парних чисел є більше за 0, виведеться на екран
  cout << "Добуток парних чисел: " << p << endl; // добуток,
else        // інакше – виведеться повідомлення, що парних чисел немає.
  cout << "Парних чисел немає." << endl;
system("pause>>void");
return 0;
}
```

Результат виконання програми:

```
Ввести 1 число: 3
Ввести 2 число: 1
Ввести 3 число: -7
Ввести 4 число: 5
Ввести 5 число: 3
Ввести 6 число: -7
Ввести 7 число: 3
Ввести 8 число: 7
Ввести 9 число: 11
Ввести 10 число: -17
Ввести 11 число: -5
Ввести 12 число: 3
Парних чисел немає.
```

Приклад 3. Ввести послідовність цілих чисел і визначити перший від'ємний елемент.

Текст програми:

```
#include <iostream>
using namespace std;
int main()
{
 setlocale(0,".1251");
 int i, n, x, v=0;
 cout << "Ввести кількість чисел: ";  cin >> n;
 cout << "Ввести " << n << " чисел: ";
 for(i=1; i<=n; i++)
 { cin >> x;
   if (x<0)                    // Якщо число від'ємне,
   { v=x;                      // запам'ятати його у змінній v
     cout << "Перший від'ємний елемент: " << v << endl; // добуток,
     break;                    // і припинити перевірку решти чисел.
    }
 }
 if(v == 0)   cout << "Від'ємних чисел немає"<< endl;
 system("pause>>void");
 return 0;
}
```

Результати виконання програми:

```
Ввести кількість чисел: 8
Ввести 8 чисел: 90 0 -8 7 -69 56 -83 4
Перший від'ємний елемент: -8

Ввести кількість чисел: 6
Ввести 6 чисел: 3 0 18 37 69 56
Від'ємних чисел немає
```

Приклад 4. Ввести послідовність дійсних чисел (a_1, a_2, a_3, ...) і обчислити $\min(|a_2 - a_1|, |a_3 - a_2|, \ldots)$.

Текст програми:

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
 setlocale(0,".1251");
 int i, n;
 double min=1e3, x0, x;
 cout << "Ввести кількість чисел: ";
 cin >> n;
 cout << "Ввести " << n << " чисел: ";
 cin >> x;
 for(i=2; i<=n; i++)
 {
   x0=x;
   cin >> x;
   if (min>fabs(x-x0)) min=fabs(x-x0)
 }
 cout << "min= " << min << endl;
 system("pause>>void");
 return 0;
}
```

Результати роботи:

```
Ввести кількість чисел: 10
Ввести 10 чисел: 2.5  30  -89  0  3.67  -45  90  78  3  5.2
min= 2.2
```

Завдання до самостійної роботи

1) Скласти схеми алгоритмів і написати програми мовою С++ із застосуванням оператора циклу для опрацювання числових послідовностей з розв'язання завдань, поданих у табл. 5.3 і 5.4 відповідно до індивідуального варіанта (див. приклади програм 1 … 4).

2) Створити на комп'ютері програмні проєкти мовою С++ для реалізації написаних програм. Занести результати обчислень до протоколу.

Таблиця 5.3

Індивідуальні завдання базового рівня складності

№ вар.	Варіанти завдань
1	Ввести 7 дійсних чисел та обчислити добуток елементів цієї послідовності, значення яких є менше за 6
2	Ввести 10 дійсних чисел та обчислити кількість додатних елементів
3	Ввести 6 дійсних чисел та обчислити суму від'ємних елементів
4	Ввести 5 дійсних чисел і визначити найменше та найбільше серед них
5	Ввести 8 дійсних чисел та обчислити середнє арифметичне ненульових
6	Ввести 9 дійсних чисел та обчислити суму елементів, абсолютне значення яких не перевищує 5
7	Ввести 11 дійсних чисел та обчислити кількість елементів послідовності, значення яких є більше за значення першого елемента
8	Ввести 6 дійсних чисел та обчислити добуток елементів послідовності, значення яких перебувають у діапазоні [3, 6]
9	Ввести 8 дійсних чисел та обчислити середнє арифметичне додатних
10	Ввести 7 дійсних чисел та обчислити суму квадратів тих чисел, модуль яких не перевищує 3
11	Ввести 14 цілих чисел та обчислити кількість ненульових елементів
12	Ввести 9 дійсних чисел та визначити мінімальний елемент послідовності
13	Ввести 6 цілих чисел та обчислити добуток ненульових елементів
14	Ввести 10 цілих чисел та обчислити середнє арифметичне елементів послідовності, значення яких перебувають у діапазоні [10, 20]
15	Ввести 8 дійсних чисел та обчислити кількість елементів, значення яких перебувають у діапазоні [5, 10]
16	Ввести 7 цілих чисел та визначити суму модулів усіх від'ємних елементів
17	Ввести 9 дійсних чисел та обчислити добуток додатних елементів, значення яких не перевищує 4
18	Ввести 12 дійсних чисел та обчислити кількість додатних і кількість від'ємних елементів послідовності
19	Ввести 8 цілих чисел та обчислити середнє арифметичне абсолютних (за модулем) значень усіх елементів послідовності
20	Ввести 6 дійсних чисел та віднайти максимальний і мінімальний елементи та визначити наскільки максимальний елемент є більшим за мінімальний
21	Ввести 11 цілих чисел та обчислити суму тільки двоцифрових елементів
22	Ввести 9 цілих чисел та обчислити добуток непарних елементів
23	Ввести 14 цілих чисел та обчислити кількість елементів, кратних до числа 3
24	Ввести 7 цілих чисел та обчислити середнє арифметичне парних елементів
25	Ввести 6 дійсних чисел та обчислити суму елементів, значення яких є меншим за значення першого елемента послідовності
26	Ввести 9 цілих чисел та обчислити добуток одноцифрових елементів
27	Ввести 8 цілих чисел та визначити найменший з непарних додатних елементів цієї послідовності

Закінчення табл. 5.3

№ вар.	Варіанти завдань
28	Ввести 11 цілих чисел та обчислити середнє арифметичне елементів, кратних до числа 3
29	Ввести 7 цілих чисел та обчислити добуток елементів, кратних до числа 5
30	Ввести 10 цілих чисел та визначити найбільший з парних додатних елементів цієї послідовності

Таблиця 5.4

Індивідуальні завдання середнього рівня складності

№ вар.	Варіанти завдань
1	Ввести послідовність дійсних чисел та обчислити кількість елементів, які більше попереднього елемента послідовності
2	Ввести послідовність дійсних чисел та обчислити суму лише тих елементів цієї послідовності, значення яких є меншими за перший елемент
3	Ввести послідовність дійсних чисел та перевірити, чи є вона упорядкованою за спаданням
4	Ввести послідовність натуральних чисел (a_1, a_2, a_3, ...) та обчислити $\min(a_1+a_2, a_2+a_3, ...)$
5	Ввести послідовність цілих чисел та визначити різницю між найменшим і першим числами послідовності
6	Ввести послідовність дійсних чисел (a_1, a_2, a_3, ...) та обчислити $\min(a_1, a_3, a_5, ...) + \max(a_2, a_4, a_6, ...)$
7	Ввести послідовність дійсних чисел (a_1, a_2, a_3, ...) та обчислити $\max(\|a_1 - a_2\|, \|a_2 - a_3\|, \ldots)$
8	Ввести послідовність цілих чисел та визначити різницю між найбільшим і першим числами послідовності
9	Ввести послідовність дійсних чисел (a_1, a_2, a_3, ...) та обчислити $a_1*a_2 + a_2*a_3 + ... + a_{n-1}*a_n$
10	Ввести послідовність дійсних чисел (a_1, a_2, a_3, ...) та обчислити $(a_2 - a_1)*(a_3 - a_2)* ... * (a_n - a_{n-1})$
11	Ввести послідовність дійсних чисел та обчислити середнє арифметичне елементів послідовності, значення яких є меншими за перший елемент
12	Ввести послідовність цілих чисел та перевірити, чи є в ній однакові сусідні числа
13	Ввести послідовність цілих чисел та з'ясувати, чи складають числа зростаючу послідовність
14	Ввести послідовність цілих чисел та визначити різницю між найбільшим і найменшим числами послідовності
15	Ввести послідовність натуральних чисел та обчислити кількість і суму тих членів послідовності, які діляться на 5 і не діляться на 7

Закінчення табл. 5.4

№ вар.	Варіанти завдань
16	Ввести послідовність натуральних чисел та обчислити подвоєну суму всіх додатних членів послідовності
17	Ввести послідовність дійсних чисел та обчислити суму від'ємних і кількість додатних елементів послідовності
18	Ввести послідовність дійсних чисел та віднайти елементи, які за значенням є найближчими, тобто різниця між якими є найменшою
19	Ввести послідовність цілих чисел та обчислити відсотковий вміст від'ємних, нульових та додатних чисел
20	Ввести послідовність цілих чисел та перевірити чи є в ній числа, однакові зі значенням першого елемента цієї послідовності
21	Ввести послідовність натуральних чисел та визначити перший нульовий елемент
22	Ввести послідовність натуральних чисел та обчислити кількість членів послідовності, які мають парні порядкові номери і є непарними числами
23	Ввести послідовність дійсних чисел та обчислити суму квадратів лише тих елементів, значення яких є меншими за перший елемент
24	Ввести послідовність натуральних чисел та обчислити кількість трицифрових чисел
25	Ввести послідовність цілих чисел та обчислити суму елементів до першого від'ємного значення
26	Ввести послідовність дійсних чисел та обчислити кількість лише тих елементів, значення яких відрізняються від першого елемента на 10
27	Ввести послідовність цілих чисел та обчислити добуток до першого нульового значення
28	Ввести послідовність натуральних чисел та обчислити суму залишків від ділення цих чисел на 2
29	Ввести послідовність дійсних чисел та визначити останній від'ємний елемент
30	Ввести послідовність натуральних чисел та обчислити кількість двоцифрових чисел

Лабораторна робота № 6

Вкладені цикли

Мета роботи: набути практичних навиків створення програм із вкладеними циклами.

Теоретичні відомості

Цикли можуть бути вкладені один в одного. При використанні вкладених циклів треба складати програму в такий спосіб, щоб внутрішній цикл повністю вкладався в тіло зовнішнього циклу, тобто цикли не повинні перетинатися. Своєю чергою, внутрішній цикл може містити власні вкладені цикли. Імена параметрів зовнішнього та внутрішнього циклів мають бути різними. Припускаються такі конструкції:

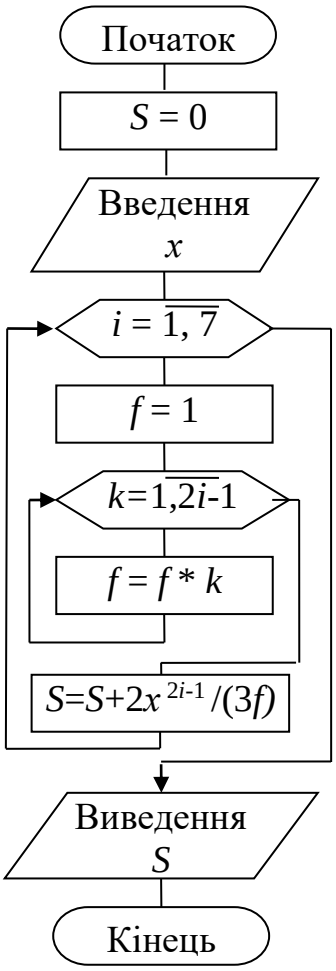
```
for(k=1; k<=10; k++)
 {  . . .
   for(i=1; i<=10; i++)
    { . . .
        for(m=1; m<=10; m++)
         {    ...
         }
    }
 }
```

Приклади проєктів програм із вкладеними циклами

Приклад 1. Обчислити суму ряду $S = \sum_{i=1}^{7} \frac{2x^{2i-1}}{3(2i-1)!}$,

де $i = 1, 2, \ldots, 7$.

Розв'язок. Для обчислення суми s треба підсумувати сім доданків, для обчислення кожного з яких слід сформувати вкладений цикл для обчислення факторіалів $(2i-1)!$. У наведеній програмі кожний доданок обчислюється в окремій змінній u.



Текст програми та блок-схема:

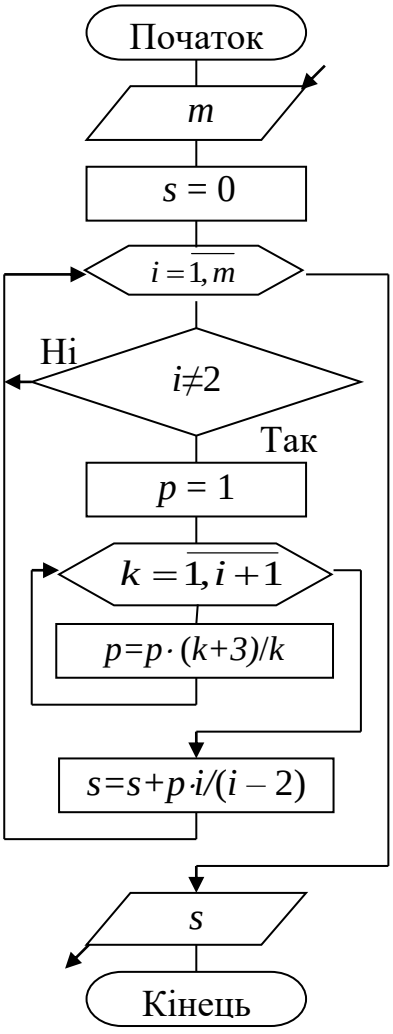
```
#include <iostream>
#include <math.h>
using namespace std;
```

```
int main()
{
 setlocale(0,".1251");
 double s=0, u, x;
 int i, k, f;
 cout<<"Введіть значення x = ";
 cin>>x;
 for (i=1; i<=7; i++)
 { f = 1;
   for (k=1; k<=2*i-1; k++) f *= k;
   u = 2*pow(x,2*i-1)/(3*f);
   s += u;
 }
 cout<<"\nСума = "<<s;
 cin.get();  cin.get();
 return 0;
 }
```

Результати:

```
Введіть значення x = 2.45
Сума = 3.83416
```

Приклад 2. Обчислити $S=\sum_{i=1}^{m}\frac{i}{i-2}\prod_{k=1}^{i+1}\frac{k+3}{k}$, значення m ввести з екрана. З обчислень вилучити доданки і множники, які дорівнюють нулю в чисельнику або знаменнику.



Розв'язок. В цьому прикладі програми наведені цикли є вкладеними один в одного, оскільки параметр внутрішнього циклу `k` залежить від параметра зовнішнього циклу `i` (`k` змінюється від `1` до `i+1`). Добуток $\prod_{k=1}^{i+1}\frac{k+3}{k}$ є співмножником доданка і обчислюється у внутрішньому циклі у змінній `P`. Оскільки внутрішній цикл складається лише з одного оператора, то операторні дужки `{ }` не є обов'язковими.

Перед зовнішнім циклом для обчислення суми слід обнулити змінну `S`, в якій будуть накопичуватись доданки, а перед зовнішнім циклом для обчислення добутку змінній `P` слід присвоїти значення `1`.

Оскільки при обчислюванні добутку беруть участь лише цілі числа, то, щоб при діленні не втратити дробову частину, слід перетворити чисельник до дійсного типу; для цього можна дописати крапку до числа 3: `(k+3.0)/k`.

Текст програми та блок-схема:

```
#include <iostream>
using namespace std;
int main()
{
 setlocale(0,".1251");
 int i, k, m;
 cout << "Введіть значення m = "; cin >> m;
  double S = 0, p;
  for (i = 1; i <= m; i++)
    if ( i != 2)
    { p = 1;
      for(k = 1; k <= i+1; k++)
        p *= (k+3.) / k;
      S += i/(i-2.) * p;
    }
 cout << "\nСума = " << S;
 cin.get();  cin.get();
 return 0;
}
```

Результати:

```
Введіть значення m = 10
Сума = 1874.05
```

Контрольні запитання та завдання для самоконтролю

1) Охарактеризуйте правила організації вкладених циклів.
2) Назвіть номер фрагмента програми з вкладеним циклом

а)
```
for(k=1;k<=10;k++)
   p=k;
  for(j=1;j<=5;j++)
   s+= p*j;
```

б)
```
for(k=1;k<=10;k++)
{ p=k;
  for(j=1;j<=5;j++)
   s+= p*j;  }
```

в)
```
for(k=1;k<=10;k++)
{ p=k; }
for(j=1;j<=5;j++)
{ s+= p*j;  }
```

3) Якого значення набуде змінна s після виконання операторів

а)
```
for(s=0,k=1; k<=3; k++)
    for(j=1; j<=k; j++) s+=j;
```

б)
```
for(s=0,k=1; k<=2; k++)
    for(m=k; m<=2; m++) s+=m;
```

Лабораторне завдання

1) У протоколі лабораторної роботи дати відповіді на контрольні запитання.

2) У протоколі лабораторної роботи скласти схеми алгоритмів і написати програми мовою С++ для розв'язання завдань, поданих у табл. 6.1 … 6.2 відповідно до індивідуального варіанта (див. приклади програм 1 … 2).

3) Створити на комп'ютері програмні проєкти мовою С++ для реалізації написаних програм. Занести результати обчислень до протоколу.

Таблиця 6.1

Індивідуальні завдання базового рівня складності

(довільне значення x слід ввести)

№	Функція	№	Функція	№	Функція	№	Функція
1	$y=\sum_{k=1}^{10}\frac{x^k}{(k+1)!}$	2	$y=\sum_{i=1}^{6}\frac{(-1)^i x^{2i}}{(3i-1)!}$	3	$y=\sum_{i=1}^{10}\frac{(-1)^{i+1} i!}{2^{2i-1}\sin x}$	4	$y=\sum_{i=1}^{5}\frac{(2i-1)!}{x^{2i-1}}$
5	$y=\sum_{k=1}^{7}\frac{k!\cos(\pi k-x)}{\ln x}$	6	$y=\prod_{k=1}^{10}\frac{x^k}{(2k)!}$	7	$y=\sum_{k=1}^{7}\frac{(2k-1)!}{2^k x^{k-1}}$	8	$y=\sum_{i=1}^{5}(-1)^{i+1}\frac{\cos x^i}{2i!}$
9	$y=\sum_{i=1}^{5}(-1)^{i+1}\frac{\sin x^i}{(2i-1)!}$	10	$y=\sum_{i=1}^{3}\frac{x^{2i}(2i-1)!}{2^i}$	11	$y=\sum_{i=1}^{6}\frac{(-1)^{i+1}x^{2i}}{(2i-1)!}$	12	$y=\sum_{i=1}^{9}\frac{(-1)^i tgx^2}{(2i+1)!}$
13	$y=\sum_{i=1}^{11}\frac{(-1)^i x^i}{i!\cos\left(i+\frac{\pi}{4}\right)}$	14	$y=\sum_{k=1}^{6}\frac{k!}{(1+x)^k}$	15	$y=\prod_{k=1}^{5}\frac{(k+1)!}{x^{k+2}}$	16	$y=\sum_{k=1}^{5}\frac{x^{k+2}}{k!}$
17	$y=\prod_{k=1}^{5}\frac{(-1)^k(2k)!}{4.5x^{2k-1}}$	18	$y=\sum_{i=1}^{6}\frac{\mathrm{tg}(x-\pi)^2}{i!}$	19	$y=\sum_{k=1}^{8}\frac{(-1)^k x^{2k-1}}{2^k k!}$	20	$y=\sum_{k=1}^{5}\frac{(-1)^k x^{3k-2}}{(k+1)!}$
21	$y=\sum_{i=1}^{11}\frac{(-1)^i x^{3i}}{(2i-1)!}$	22	$y=\sum_{k=1}^{8}\frac{2k!\cos(\pi-x)}{x^{2k-1}}$	23	$y=\sum_{k=1}^{7}\frac{(-1)^k\cdot k!}{\sqrt{x}+\sin x}$	24	$y=\prod_{k=1}^{5}\frac{(-1)^k k!}{x^{k+2}}$
25	$y=\sum_{k=1}^{6}\frac{(-1)^k\cdot x^{4k+1}}{(2k)!}$	26	$y=\sum_{k=1}^{6}(-1)^k\frac{x^k}{k!}$	27	$y=\sum_{k=1}^{5}\frac{(-1)^{k-1}\cdot x^{k+2}}{k!}$	28	$y=\sum_{i=1}^{10}\frac{(2i-1)x^{i+1}}{2i!}$
29	$y=\sum_{i=1}^{5}\frac{(-1)^i x^{2i}}{(i+1)!\cos x}$	30	$y=\sum_{i=1}^{7}\frac{(i+1)x^i}{i!}$				

Таблиця 6.2

Індивідуальні завдання високого рівня складності

№	Функція	№	Функція	№	Функція
1	$S=\sum_{k=1}^{n}\frac{(k+1)}{(k-5)}\prod_{m=1}^{k+1}\frac{m-2}{m^2-9}$	2	$Z=\prod_{j=-4}^{k}\frac{(j+2)}{j-3}\sum_{i=j}^{k+5}\left(\frac{\sqrt[5]{i+5}}{i-11}\right)$	3	$S=\sum_{k=0}^{n}\frac{(-2)^{k+1}}{(k-5)}\prod_{i=1}^{k+1}\frac{i}{i^2-16}$
4	$Z=\prod_{j=-2}^{k}\frac{j}{j-1}\sum_{i=j}^{k}\frac{i}{i+5}$	5	$W=\sum_{i=1}^{k}\frac{(-1)^i(i+1)!}{i^2-4}$	6	$Y=\sum_{n=1}^{k}\frac{(-1)^{2-n}\left(n^2-9\right)^2}{(n-2)(n+1)!}$
7	$W=\sum_{i=1}^{k}\frac{(-1)^i}{(i-4)^2}\prod_{n=1}^{i+2}\frac{n^2-4}{n+2}$	8	$L=\prod_{j=1}^{k}\frac{(j-5)}{j-2}\sum_{i=k}^{12}\frac{\sqrt{i+5}}{i-1}$	9	$Q=\sum_{k=1}^{n}\frac{(k-1)^{k+1}(k-3)}{(k+1)!}$
10	$Z=\prod_{t=1}^{k}\frac{k-t-1}{\cos(t)-3}\sum_{i=1}^{t}\left(\frac{3i-2}{i-7}\right)$	11	$P=\prod_{j=1}^{k}\frac{(j-6)j}{j-3}\sum_{i=j}^{12}\frac{\sqrt[3]{i+5}}{i-11}$	12	$A=\prod_{j=1}^{k}\frac{j-3}{(j-4)j}\sum_{i=0}^{j}\frac{\sqrt{i+4}}{i-1}$
13	$P=\prod_{j=2}^{k}\frac{(j-6)j}{(j-3)(j-1)!}$	14	$Q=\sum_{k=1}^{n}\frac{(-1)^k(k-3)^2}{k!}$	15	$U=\prod_{t=1}^{k}\frac{\sin(t)}{t-3}\sum_{i=1}^{t}\left(\frac{i+2}{i-7}\right)$

Закінчення табл. 6.2

№	Функція	№	Функція	№	Функція
16	$S=\sum_{k=1}^{n}\frac{(-3)^{3k-1}}{(k-2)^{3k+1}}\prod_{m=1}^{k+n}\frac{m+3}{m^2-25}$	17	$Z=\prod_{j=-4}^{k}\frac{j+2}{j(j-3)}\sum_{i=j}^{k+5}\left(\frac{i+5}{i-11}\right)$	18	$Y=\sum_{i=1}^{k}\frac{(i-4)^i}{(3-i)^2}\prod_{n=i}^{2+k}\frac{n+0.8}{n-i}$
19	$G=\prod_{j=-3}^{k}\frac{(j-1)}{4j-3}\sum_{i=j}^{k+5}\left(\frac{i+5}{1-i+j}\right)$	20	$D=\sum_{i=-2}^{k}\frac{(-2^i)\sin^2(i+3)}{(i+3)!}$	21	$R=\sum_{i=0}^{k}\frac{(1-i)^i}{(i+3)}\prod_{n=i}^{2k}\frac{n-i}{n+2}$
22	$Q=\sum_{k=1}^{n}\frac{(k-1)^{k+1}(k-7)}{k!}$	23	$W=\sum_{i=1}^{k}\frac{(-1)^i(i-3)^2}{i!}$	24	$A=\prod_{j=1}^{k}\frac{(j^2-4)j}{j-k+1}\sum_{i=j}^{9}\frac{i-3}{i-7}$
25	$P=\prod_{j=1}^{k}\frac{j-6}{j-3}\sum_{i=j}^{10}\frac{\sin(i+5)}{i-4}$	26	$Z=\sum_{n=-2}^{k}\frac{(n+1)\lvert n-9\rvert}{(n+3)!}$	27	$p=\sum_{i=1}^{n}\frac{(2i+1)(i-3)}{(2i-1)!}$
28	$W=\sum_{i=1}^{k}\frac{(i-1)^i}{(i+3)^2}\prod_{n=1}^{i+1}\frac{n^2-1}{n-2}$	29	$Y=\sum_{i=-1}^{k}\frac{(k-i)^i(i+2)!}{i^2-4}$	30	$F=\sum_{n=0}^{k}\frac{(n+2^k)\lvert n-4\rvert}{(n)!}$

Самостійна робота № 6

Циклічне опрацювання чисел

Мета роботи: закріпити практичні навички використання циклів.

Приклади програм

Приклад 1. Визначити і вивести всі дільники введеного натурального числа.

Текст програми:

```
#include <iostream>
int main()
{ system("cls");          // Очистити екран
  system("chcp 1251");
  system("color F0");     // Встановити білий фон і чорний текст
  unsigned int x, i;
  std::cout << "Введіть x: ";  std::cin  >> x;
  std::cout << std::endl << "Дільники: ";
  for (i=1; i<=x; i++)
   if (x%i == 0)              // Якщо ділиться без остачі,
     std::cout << i << "\t";  // вивести черговий дільник
  std::cin.get();
  std::cin.get();
  return 0;
}
```

Результати виконання програми:

```
Введіть x: 45

Дільники: 1   3   5   9   15   45
```

Приклад 2. Визначити кількість цифр введеного натурального числа.

Текст програми:

```
#include <iostream>
using namespace std;
int main()
{
  unsigned int N, kol;
  cout << "N= ";   cin >> N;
  for (kol=1; N/10>0; kol++,N/=10);
  cout << "kol= " << kol << endl;
  system("pause>>void");
  return 0;
}
```

```
Введіть x: 45

Дільники: 1    3       5       9       15      45
```

```
Введіть x: 121

Дільники: 1    11      121
```

Приклад 3. Визначити найменшу та найбільшу цифру введеного натурального числа.

Текст програми:

```
#include <iostream>
using namespace std;
int main()
{
  unsigned int N, z, min, max;
  cout<<"N="; cin>>N;
  for (max=min=N%10; N>0; N/=10)
  {  z=N%10;
     if (z>max) max=z;
     if (z<min) min=z;
  }
  cout << "min=" << min << endl << "max=" << max << endl;
  system("pause>>void");
  return 0;
}
```

```
N=550308
min=0
max=8
```

Завдання до самостійної роботи

1) Скласти схеми алгоритмів і написати програми мовою С++ із застосуванням оператора циклу для опрацювання числових послідовностей з розв'язання завдань, поданих у табл. 6.3 відповідно до індивідуального варіанта.

2) Створити на комп'ютері програмні проєкти мовою С++ для реалізації написаних програм. Занести результати обчислень до протоколу.

Таблиця 6.3

Індивідуальні завдання

№ вар.	Варіанти завдань
1	Ввести ціле додатне число (>10). Найти найбільшу цифру цього числа
2	Ввести цілі числа M та N ($M < N$). Вивести усі непарні числа з цього проміжку, кратні трьом
3	Ввести дійсне число A і ціле число N (> 0). Обчислити A у степені N: $A^N = A \cdot A \cdot \ldots \cdot A$ (числа А перемножуються N разів)
4	Ввести дійсне число A і ціле число N (> 0). Обчислити усі цілі степені числа A від 1 до N
5	Ввести дійсне число A і ціле число N (> 0). Обчислити $1 + A + A^2 + A^3 + \ldots + A^N$
6	Ввести дійсне число A (> 1). Вивести найменше з цілих чисел, для яких сума $1 + 1/2 + \cdots + 1/N$ буде більшою за A, і саму цю суму
7	Ввести ціле число N (> 0). Обчислити добуток $1 \cdot 2 \cdot \ldots \cdot N$. Для уникнення цілочислового переповнення обчислювати цей добуток у дійсній змінній і вивести його як дійсне число

Закінчення табл. 6.3

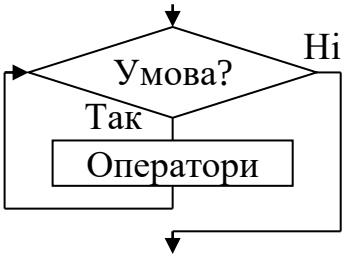
№ вар.	Варіанти завдань
8	Ввести ціле число N (> 0). Обчислити добуток $2 \cdot 1/(2) \cdot 1/(3) \cdot \ldots \cdot 1/(N)$
9	Ввести дійсне число X і ціле N (> 0). Обчислити $1 + X + X^2/2 + \ldots + X^N/N$
10	Ввести дійсне число A і ціле N (> 0). Обчислити $1 - A + A^2 - A^3 + \ldots + (-1)^N A^N$
11	Ввести дійсне число X і ціле число N (> 0). Обчислити значення $X - X^3/3 + X^5/5 - \ldots + (-1)^N X^{2N+1}/(2N+1)$
12	Ввести дійсне число X і ціле число N (> 0). Обчислити значення $1 - X^2/2 + X^4/4 - \ldots + (-1)^N X^{2N}/(2N)$
13	Ввести дійсне число X ($\|X\| < 1$) і ціле число N (> 0). Обчислити значення $X - X^2/2 + X^3/3 - \ldots + (-1)^{N-1} X^N/N$. Саме це значення є приблизним значенням функції ln() у точці $1+X$
14	Ввести дійсне число X ($\|X\| < 1$) і ціле число N (> 0). Обчислити значення $X - X^3/3 + X^5/5 - + (-1)^N X^{2N+1}/(2N+1)$. Саме це значення є приблизним значенням функції arctg() у точці X
15	Ввести ціле число N. Вивести усі непарні числа від 1 до N, кратні п’яти
16	Ввести цілі числа M та N ($M < N$). Вивести усі парні числа з цього проміжку, кратні трьом
17	Ввести ціле додатне число N (>10). Ввести числа від 10 до N, кратні п’яти
18	Вивести квадрати чисел від 11 до 99.
19	Для введеного числа n в одному циклі обчислити $n!$ та 2^n
20	Ввести ціле додатне число (>10). Найти першу цифру цього числа
21	Ввести ціле додатне число (>10). Найти суму цифр цього числа
22	Ввести ціле додатне число (>10). Поміняти місцями першу та останню цифри цього числа, наприклад, з числа 12348 отримати число 82341
23	Ввести ціле додатне число (>10). Найти найменшу цифру цього числа
24	Ввести ціле додатне число і визначити, чи є воно симетричним, тобто таким числом (паліндром, перевертиш), як наприклад числа: 123321, 989
25	Ввести ціле додатне число і переформувати його навпаки, наприклад, з числа 1248 отримати число 8421
26	Ввести ціле додатне число (>100). Позбавитись у ньому від другої цифри числа, наприклад з числа 1234 отримати число 134
27	Ввести ціле додатне число (>100). Позбавитись у ньому від першої цифри числа, наприклад з числа 1234 отримати число 234
28	Вивести усі прості двоцифрові додатні числа (які мають тільки два дільники – 1 та саме це число)
29	Ввести два цілі числа A та B ($A < B$). Найти всі цілі числа, розташовані поміж цими числами (включно з цими числами), у порядку зростання їхніх значень, а також кількість таких чисел
30	Ввести два цілі числа A та B ($A < B$). Найти всі цілі числа, розташовані поміж цими числами (не враховуючи ці числа), у порядку зменшення їхніх значень, а також кількість таких чисел

Лабораторна робота № 7
Оператори циклу while та do-while

Мета роботи: набути практичних навиків організації циклічних обчислень у C++ з використанням операторів циклу з передумовою **`while`** та післяумовою **`do-while`**.

Теоретичні відомості

Оператори з передумовою та післяумовою використовуються для організації циклів і є альтернативними операторові `for`. Звичайно цикл з передумовою використовується, якщо кількість повторювань заздалегідь є невідома або немає явно вираженого кроку змінювання параметра циклу. А для багаторазових повторювань тіла циклу відомим є вираз умови, за істинності якої цикл продовжує виконання. Цю умову слід перевіряти кожного разу перед черговим повторенням.

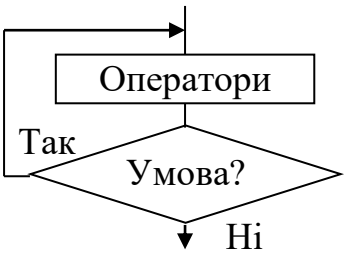


Синтаксис циклу з передумовою:

```
while (<умова>)
   { <тіло циклу> };
```

Послідовність операторів (тіло циклу) виконується, доки *умова* є істинна (`true`, має ненульове значення), а вихід з циклу здійснюється, коли *умова* стане хибною (`false`, матиме нульове значення). Якщо *умова* є хибною при входженні до циклу, то *послідовність операторів* не виконуватиметься жодного разу, а керування передаватиметься до наступного оператора програми.

Цикл з післяумовою використовується, якщо є потреба перевіряти умову кожного разу після чергового повторення. Відмінність циклу з передумовою від циклу з післяумовою полягає в першій ітерації: цикл з післяумовою завжди виконується принаймні одноразово незалежно від умови.



Синтаксис циклу з післяумовою:

```
do {
     <тіло циклу>
   } while (<умова>);
```

Послідовність операторів (*тіло циклу*) виконується один чи кілька разів, доки *умова* стане хибною (`false` чи дорівнюватиме нулю). Якщо *умова* є істинна (ненульова), то оператори тіла циклу виконуються повторно. Оператор циклу `do-while` використовується в тих випадках, коли є потреба виконати тіло циклу хоча б одноразово, оскільки перевірка умови здійснюється після виконання операторів.

Якщо тіло циклу складається з одного оператора, то операторні дужки `{}` не є обов’язкові.

Оператори `while` та `do-while` можуть завчасно завершитись при виконанні операторів `break`, `goto`, `return` усередині тіла циклу.

Варто зауважити, що в тілі циклу слід передбачати змінювання параметрів, які беруть участь в умові, інакше умову виходу з циклу ніколи не буде виконано й відбуватиметься зациклювання.

Наведемо відмінності роботи різних операторів циклу на прикладі обчислення суми всіх непарних чисел у діапазоні від 10 до 100:

1) з використанням оператора `for`

```
int i, s=0;
for (i=11; i<100; i += 2) s += i;
```

2) з використанням оператора `while`

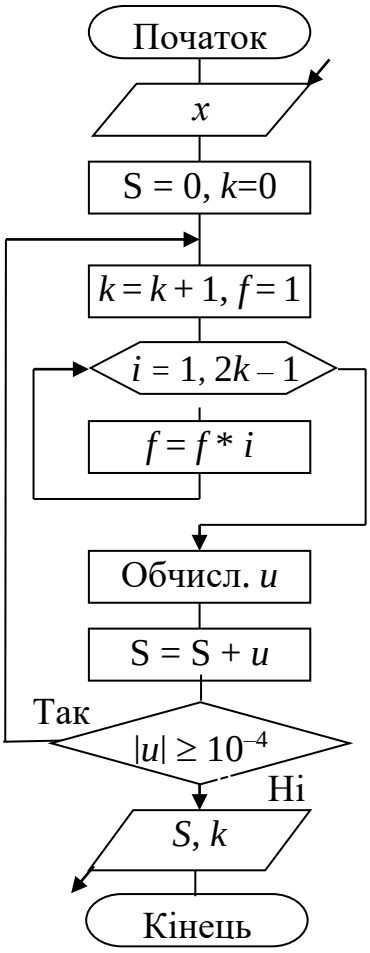
```
int s=0, i=11;
while (i<100) { s += i; i += 2; }
```

3) з використанням оператора `do-while`

```
int s=0, i=11;
do { s += i; i += 2;} while (i<100);
```

Приклади програм

Приклад 1. Обчислити суму ряду $S = \sum_{k=1}^{\infty} \frac{x^{2k+1}}{3^k (2k-1)!}$, підсумовуючи члени ряду, значення яких за модулем є більше за задану точність $\varepsilon = 10^{-4}$. Визначити кількість доданків. Значення x $(-2 < x < 2)$ вводити з клавіатури.



Розв'язок. У цій програмі недоцільно використовувати оператор циклу з параметром `for`, оскільки кількість повторень циклу є наперед невідома. Доцільним буде використання оператора циклу з післяумовою `do-while`, оскільки на момент першої перевірки умови вже треба знати значення першого доданка.

Текст програми та блок-схема:

```
#include <iostream>
#include <math.h>
using namespace std;
int main()
{
 setlocale(0,".1251");
 double x, f, u, s=0;
 int i, k=0;
 cout << "Введіть значення x= ";
 cin >> x;
 cout << "\nРезультати:\n";
```

```
 do                                          // Цикл з післяумовою
 { k++;                                      // Збільшення змінної  k на 1
  for(i=1,f=1; i<=2*k-1; i++) f *= i;        // Обчислення факторіалу
  u=pow(x, 2*k+1)/(pow(3.,k)*f);             // Обчислення k-го доданка
  s+=u;                                      // Підсумовування доданків
 } while(fabs(u)>=1e-4);
 cout << "Сума = " << s << endl;
 cout << "Кількість доданків = " << k << endl;
 system("pause>>void");
 return 0;
}
```

Результати роботи:

```
Введіть значення x= 2.5

Результати:
Сума = 7.21478
Кількість доданків = 6
```

Приклад 2. Обчислити суму ряду $y = \sum_{k=1}^{\infty} \frac{(-1)^{k+1} x^{2k-1}}{(k+3)(2k)!}$, підсумовуючи члени ряду, значення яких за модулем є більше за задану точність ε. Визначити кількість доданків. Значення x $(-2 < x < 2)$ і $\varepsilon = 10^{-4}$ вводити з клавіатури.

Розв'язок. Множник $(-1)^{k+1}$ за непарних k = 1, 3, 5, … дорівнює 1, а за парних k = 2, 4, … – дорівнює (–1). Отже, поданий ряд є знакозмінним, де всі парні доданки будуть від'ємними, а всі непарні – зі знаком "+". Для наочності й контролю правильності у розв'язку окремо наведемо всі доданки.

Для того щоби зробити алгоритм програми більш оптимальним, його можна удосконалити, але для цього слід обчислити рекурентний множник. Це дозволить у даній програмі позбавитись від вкладеного циклу для обчислення факторіалу й оператора перевірки на парність k.

```
u = x/8;
s = u;
do
{
  k++;
  cout<< "доданок " << k <<   ":  " << u << endl;
  r = -x*x*(k+2)/((3+k)*(2*k) *(2*k-1));
  u *= r ;
  s += u;
}
while(fabs(u) >= eps);
```

Зупинимось більш докладно на виведенні рекурентної формули.

Подамо суму ряду $y=\sum_{k=1}^{\infty}\frac{(-1)^{k+1}x^{2k-1}}{(k+3)(2k)!}$ у вигляді $y=\sum_{k=1}^{\infty}u_k$, де $u_k=\frac{(-1)^{k+1}x^{2k-1}}{(k+3)(2k)!}$. Рекурентний множник R – це співвідношення двох поряд розміщених членів ряду:

$$u_2=R\cdot u_1,\quad u_3=R\cdot u_2,\quad \ldots,\quad u_k=R\cdot u_{k-1},$$

звідки

$$\boxed{R=\frac{u_k}{u_{k-1}}}.$$

Для визначення R в цю формулу слід підставити $u_k=\frac{(-1)^{k+1}x^{2k-1}}{(k+3)(2k)!}$ та u_{k-1}. Для обчислення u_{k-1} підставимо у вираз для u_k $(k-1)$ замість k:

$$u_{k-1}=\frac{(-1)^{k}x^{2(k-1)-1}}{(k+2)(2k-2)!}=\frac{(-1)^{k}x^{2k-3}}{(k+2)(2k-2)!};$$

$$R=\frac{u_k}{u_{k-1}}=u_k=\frac{(-1)^{k+1}x^{2k-1}}{(k+3)(2k)!}\cdot\frac{(k+2)(2k-2)!}{(-1)^{k}x^{2k-3}}=\frac{(-1)^{k+1}\cdot x^{2k-1}\cdot(k+2)\cdot(2k-2)!}{(-1)^{k}\cdot x^{2k-3}\cdot(k+3)\cdot(2k)!}=$$

$$=\frac{(-1)^{k+1-k}\cdot x^{2k-1-(2k-3)}\cdot(k+2)}{(k+3)}\cdot\frac{1\cdot2\cdot\ldots\cdot\cancel{(2k-2)}}{1\cdot2\cdot\ldots\cdot\cancel{(2k-2)}\cdot(2k-1)\cdot2k}=-\frac{x^2(k+2)}{2k(k+3)(2k-1)}.$$

Крім рекурентного множника R, треба обчислити перший член ряду за $k=1$:

$$u_1=\frac{(-1)^{2}x}{4\cdot(2)!}=\frac{x}{8}.$$

Текст програми:

```
#include <iostream>
#include <math.h>
using namespace std;
int main()
{
 setlocale(0,".1251");
 double x, u, r, s=0, eps;
 int k=0;
 cout <<"Введіть значення x= ";
 cin >> x;
 cout <<"Введіть значення точності ";
 cin >> eps;
 cout << "\nРезультати:\n";
 u = x/8;
 s = u;
 do
   {
```

```
      ++k;
      cout<< "доданок " << k <<    ":  " << u << endl;
      r = -x*x*(k+2)/((3+k)*(2*k) *(2*k-1));
      u *= r ;
      s += u;
    }
 while (fabs(u) >= eps);
 cout << "Сума = " << s << endl;
 cout << "Кількість доданків = " << k << endl;
 system("pause>>void");
 return 0;
}
```

Результати роботи:

```
Введіть значення x= 2.4
Введіть значення точності 0.0001

Результати:
доданок 1: 0.3
доданок 2: -0.648
доданок 3: 0.248832
доданок 4: -0.0398131
доданок 5: 0.00351005
доданок 6: -0.0001196563
Сума = -0.13566
Кількість доданків = 6
```

Контрольні запитання та завдання для самоконтролю

1) Які оператори циклу використовуються у мові С++?
2) Назвати правильну, на Ваш погляд, послідовність номерів, для запису елементів оператора циклу `while`:
   а) логічний вираз умови;
   б) оператори тіла циклу;
   в) `while`.
3) Вказати значення `n` після виконання фрагментів програми:
   а) `int k=0, n=17;`
      `while(k<7){ k++;  n--; }`
   б) `int m=1, n=1;`
      `while(m<5){ m+=2; n=n*m; }`
4) Якими є структура і порядок виконання оператора циклу `do-while`?
5) Записати трьома операторами циклу варіанти обчислення $S = \sum_{i=1}^{6} i^2$.
6) Вказати значення `s` після виконання операторів

```
s=0.5; i=0;
while(i<5) i++;
s+=1.0/i;
```

Лабораторне завдання

1) У протоколі лабораторної роботи дати відповіді на контрольні запитання.

2) У протоколі лабораторної роботи скласти схему алгоритму і програму для обчислення суми нескінченного ряду, підсумовуючи члени ряду, значення яких за модулем перевищують задану точність $\varepsilon = 10^{-4}$. Визначити кількість доданків. Обчислення виконати для x ($-2 < x < 2$), яке ввести. Завдання вибрати з табл. 7.1 відповідно до індивідуального варіанта (див. приклад програм 1 … 2).

3) У протоколі лабораторної роботи скласти схему алгоритму і програму для обчислення цієї самої суми (табл. 7.1) за більш оптимальним алгоритмом з обчисленням рекурентного множника, що дозволить позбавитись у програмі від вкладеного циклу та можливого переповнення при обчисленні факторіалу.

4) Створити на комп'ютері програмні проєкти мовою C++ для реалізації написаних програм. Занести результати обчислень до протоколу.

Таблиця 7.1

Індивідуальні завдання середнього рівня складності

№	Функція $f(x)$	№	Функція $f(x)$	№	Функція $f(x)$
1	$\sum_{k=1}^{\infty}\frac{(-1)^k x^{2k+1}}{(2k+1)!}$	2	$\sum_{k=1}^{\infty}\frac{(-1)^{k+1} x^{2k}}{k!2^{k-1}}$	3	$\sum_{k=1}^{\infty}\frac{(-1)^{k-1} x^{k}}{k!}$
4	$\sum_{k=1}^{\infty}\frac{(-1)^k x^{2k+1}}{k(2k+1)!}$	5	$\sum_{k=1}^{\infty}\frac{(-1)^k x^{2k-1}}{k(k+3)!}$	6	$\sum_{k=1}^{\infty}\frac{(-1)^k x^{2k+1}}{2k(k+1)!}$
7	$\sum_{k=1}^{\infty}\frac{(-1)^{k+1} x^{2k}}{(2k-1)!}$	8	$\sum_{k=1}^{\infty}\frac{(-1)^{k-1} x^{3k-1}}{(2k)!}$	9	$\sum_{k=1}^{\infty}\frac{(-1)^{k+1} x^{k+3}}{k^2(k+2)!}$
10	$\sum_{k=1}^{\infty}\frac{(-1)^{k+1} x^{2k}}{(2k+1)!}$	11	$\sum_{k=1}^{\infty}\frac{(-1)^{k-1} x^{3k-1}}{(k+2)k!}$	12	$\sum_{k=1}^{\infty}\frac{(-1)^{k} x^{3k-1}}{(k+3)(3k)!}$
13	$\sum_{k=1}^{\infty}\frac{(-1)^{k-1} x^{3k+1}}{3k(k+1)!}$	14	$\sum_{k=1}^{\infty}\frac{(-1)^{k} x^{2(k+1)}}{(k+2)k!}$	15	$\sum_{k=1}^{\infty}\frac{(-1)^{k+1} x^{3(k-2)}}{(k+3)(3k)!}$
16	$\sum_{k=1}^{\infty}\frac{(-1)^k x^{2k+1}}{2^k(2k-1)!}$	17	$\sum_{k=1}^{\infty}\frac{(-1)^{k+1} x^{k}}{(k+4)!}$	18	$\sum_{k=1}^{\infty}\frac{(-1)^{k} x^{3k-1}}{(k+1)!k^2}$
19	$\sum_{k=1}^{\infty}\frac{(-1)^{k+1} x^{2k}}{k!2^k}$	20	$\sum_{k=1}^{\infty}\frac{(-1)^{k} x^{3k-1}}{2k(k+3)!}$	21	$\sum_{k=1}^{\infty}\frac{(-1)^{k+1} x^{k+2}}{k(2k+1)!}$
22	$\sum_{k=1}^{\infty}\frac{(-1)^k x^{2(k+1)}}{(2k)!}$	23	$\sum_{k=1}^{\infty}\frac{(-1)^k x^{2k+1}}{k(2k+1)!}$	24	$\sum_{k=1}^{\infty}\frac{(-1)^{k-1} x^{3k-2}}{2k(k+3)!}$
25	$\sum_{k=1}^{\infty}\frac{(-1)^k x^k (2k-1)}{(3k-2)!}$	26	$\sum_{k=1}^{\infty}\frac{(-1)^{k+1} x^{k-1}}{(2k-1)(k+1)!}$	27	$\sum_{k=1}^{\infty}\frac{(-1)^{k-1} x^{3k-2}}{2^{k+1}k!}$
28	$\sum_{k=1}^{\infty}\frac{(-1)^{k+1} x^{k-1}}{(2+k)(2k+1)!}$	29	$\sum_{k=1}^{\infty}\frac{(-1)^{k-1} x^{3k+1}}{3k(2k-1)!}$	30	$\sum_{k=1}^{\infty}\frac{(-1)^{k} x^{2k-1}}{2k(2k+1)!}$

Самостійна робота № 7
Застосування циклів

Мета роботи: закріпити практичні навички програмної організації циклічних обчислень у мові С++.

Приклади програм

Приклад 1. Перевірити чи є введене натуральне число паліндромом.

Паліндром – це число (слово чи фраза), яке однаково читається в обох напрямках, або, інакше кажучи, будь-який симетричний відносно своєї середини набір символів. Наприклад, числа 404, 12521 – паліндроми.

Текст програми:

```
#include <iostream>
using namespace std;
int main()
{
 setlocale(0,".1251");
 unsigned long long n,m,d;
 cout<<"N="; cin>>n;
 d = 0; m = n;
 while ( m!=0)
  { d = d*10 + m%10;
    m /= 10;
  }
 if (d==n) cout<<"Так, число є паліндромом"<<endl;
 else cout<<"Ні, число не є паліндромом "<<endl;
 system("pause>>void");
 return 0;
}
```

```
N=25052
Так, число є паліндромом
```

Приклад 2. Написати програму обчислення факторіала введеного цілого числа за допомогою оператора `while`.

Порівняно з обчисленням факторіала оператором `for` чи засобами рекурсивної функції, наведене розв'язання є самим лаконічним і елегантним.

```
#include <iostream>
using namespace std;
int main()
{ int f = 1, n;
  cout << "Введіть ціле число\n";  cin >> n;
  while ( n >= 1 ) f *= n--;
  cout <<"Факторіал цього числа "<< f << endl;
  return 0;
}
```

```
Введіть ціле число
5
Факторіал цього числа 120
```

Приклад 3. Написати програму обчислення найбільшого спільного дільника (НСД) двох цілих чисел, тобто реалізувати алгоритм Евкліда.

Текст програми:

```
#include <iostream>
using namespace std;
int main()
{ setlocale(0,".1251");
  printf("Обчислення найбільшого спільного дільника двох цілих чисел.");
  printf("\nВведіть в один рядок два числа і натисніть <Enter>");
  printf("\n -> ");
  int nod;        // Найбільший спільний дільник (НДС)
  int n1,n2;      // Числа, НСД яких треба обчислити
  int r;          // Остача від ділення n1 на n2
  scanf("%i %i", &n1, &n2) ;
  printf("НСД чисел %i і %i - це ", n1, n2);
  while (n1 % n2)
  { r = n1 % n2; // Остача  від ділення
    n1 = n2;
    n2 = r;
  }
  nod = n2;
  printf("%i\n", nod);
  system("pause>>void");
  return 0;
}
```

Результати виконання програми:

```
Обчислення найбільшого спільного дільника двох цілих чисел.
Введіть до одного рядка два числа і натисніть [Enter]
 -> 121 33
НСД чисел 121 і 33 – це 11
```

Контрольні запитання та завдання для самоконтролю

1) Чим схожі і чим відрізняються оператори `while` та `do-while`?
2) Чи є взаємозамінними оператори циклу у мові С++?
3) Вказати значення у після виконання фрагментів програми:

а)
```
int i=1, y=1;
do {y*=i++;} while(y<7);
```

б)
```
int k=1;  float y=0;
do{k+=2;y+=1./k;}while(k<5);
```

Завдання до самостійної роботи

1) Дати відповіді на контрольні запитання.

2) Скласти схему алгоритму і написати програмний код мовою С++ для розв’язання індивідуального завдання з табл. 7.2.

3) Створити на комп’ютері програмний проєкт мовою С++ для реалізації написаних програм. Занести результати обчислень до протоколу.

Таблиця 7.2

Індивідуальні завдання

№ вар.	Завдання
1	Ввести натуральне число та визначити суму цифр числа
2	Ввести натуральне число та визначити першу цифру числа, наприклад, для числа 3406 цифра 3
3	Ввести натуральне число та визначити суму всіх його дільників
4	Ввести натуральне число та визначити кількість парних цифр числа
5	Ввести натуральне число та визначити найбільшу цифру числа
6	Ввести натуральне число та змінити порядок цифр числа, наприклад, було 1234, стало 4321
7	Ввести натуральне число та визначити його цифри, які кратні 3
8	Ввести натуральне число та дописати в нього по 1 на початок і кінець
9	Ввести натуральне число та визначити скільки разів його перша цифра зустрічається у числі
10	Ввести натуральне число та визначити кількість його дільників менших 10
11	Ввести натуральне число та визначити чи є воно степенем числа 3
12	Ввести натуральне число та поміняти місцями першу й останню цифру числа, наприклад, з числа 1234 отримати 4231
13	Ввести натуральне число та дописати до нього таке само число, наприклад, з числа 1234 отримати 12341234
14	Ввести натуральне число та визначити кількість його парних дільників
15	Приписати до введеного числа таке само число, наприклад, з числа 1234 здобути 12341234
16	Ввести натуральне число та визначити кількість непарних цифр числа
17	Ввести натуральне число та визначити найменшу цифру числа
18	Ввести натуральне число та визначити його цифри, які кратні 5
19	Ввести натуральне число та визначити чи є воно степенем числа 2
20	Ввести натуральне число та визначити кількість його непарних дільників
21	Вивести всі двоцифрові числа, які діляться на 5 і містять цифру 5
22	Вивести всі чотирицифрові числа, які при діленні на 47 дають в залишку 43, а при діленні на 43 дають в залишку 47
23	Ввести натуральне число та визначити кількість його дільників
24	Ввести натуральне число та визначити суму парних дільників
25	Вивести всі трицифрові числа, кратні 7, і сума цифр яких також кратна 7
26	Вивести всі двоцифрові числа, які діляться на 9 або містять цифру 9
27	Вивести всі двоцифрові числа, сума квадратів цифр яких ділиться на 13
28	Ввести натуральне число та визначити суму непарних дільників
29	Ввести натуральне число та поміняти місцями першу і другу цифру числа, наприклад, з числа 1234 отримати 2134
30	Ввести натуральне число та продублювати кожну з цифр числа двічі, наприклад, з числа 1234 отримати 11223344

ТЕМА 3
МОДУЛЬНА ОРГАНІЗАЦІЯ ПРОГРАМ

Лабораторна робота № 8
Організація функцій у С++

Мета роботи: набути практичних навиків програмної організації функцій засобами С++ для опрацювання різної кількості вхідних і вихідних даних.

Теоретичні відомості

Призначення та поняття функцій

Коли програма стає завеликою за обсягом і складною для сприймання, є сенс розділити її за змістом на невеликі логічні частини, називані функціями, кожна з яких виконуватиме певне завдання. Унаслідок цього програма стане більш легкою і для розуміння при створюванні, і для процесу налагодження. Окрім того, створення функції позбавляє потреби створювання двох, а то й більшої кількості, майже однакових фрагментів програмного коду для розв'язання однакових завдань за різних вхідних даних. Розподіл програми на функції є базовим принципом структурного програмування.

Функція – це незалежна іменована частина програми, яка може багаторазово викликатися з інших частин програми, маніпулювати даними та повертати результати. Кожна функція має власне ім'я, за яким здійснюють її виклик. Розрізнюють два основні різновиди функцій:

1) стандартні вбудовані функції, які є складовою частиною мови програмування, наприклад: `sin()`, `pow()` тощо;

2) функції, створювані користувачем для власних потреб.

Створювана у програмі функція повинна бути *оголошеною* і *визначеною*. Оголошення функції має бути написаним у програмі раніш за її використання. Визначення може перебувати у будь-якому місці програми, за винятком тіла (середини) інших функцій.

Оголошення функції складається з ***прототипу*** (заголовка) і має форму

[*клас*] <*тип результату*> <*ім'я функції*>
(<*перелік формальних аргументів із зазначенням типу*>) ;

Наприклад, оголошення функції обчислення середньоарифметичного двох цілих чисел може мати вигляд

```
double seredne (int a, int b);
```

де `seredne` – ім'я функції;

`a` і `b` – формальні вхідні аргументи (параметри), які за виклику набувають значень фактичних параметрів;

`double` – тип функції, який безпосередньо є типом результату виконання операторів функції. Результат повертається оператором `return`.

Прототип функції може бути розміщено як безпосередньо у програмі, так і в заголовному файлі. У разі, коли визначення функції розміщено у програмі раніш за точку виклику, писати окремо оголошення й окремо реалізацію цієї функції немає потреби й оголошення функції можна уникнути.

Крім оголошення, кожна функція повинна мати визначення (реалізацію).

Визначення (реалізація) функції, крім заголовка, містить тіло функції (команди, які виконує функція) і команду повернення результату `return`:

<тип функції> ***<ім'я функції>*** (*<перелік аргументів>*)
{ *<тіло функції>*
`return` *<результат обчислень>*`;`
}

Наприклад, визначення функції `seredne` може бути таким:

```
double seredne (int a, int b)
{ return (a+b)/2.0;
}
```

Наведена функція обчислює середньоарифметичне двох цілих чисел `a` та `b` і повертає його значення у точку виклику за допомогою оператора `return`.

Виклик функції `seredne()` може бути в один зі способів:

```
int  a=5, x=2, y=-3; double z, s1, s2;
1) z = seredne(5, 4);        // z = 4.5
2) s1 = seredne(a, 11);      // s1 = 8
3) s2 = seredne(x, y);       // s2 = -0.5
```

У першому наведеному виклику числа 5 та 4 є фактичними параметрами, які підставляються у дужки замість формальних параметрів `a` та `b`:

```
double seredne(int a, int b)
                      ↑     ↑
       z = seredne( 5,     4);
```

У другому виклику першим фактичним параметром є змінна `a`, а другим – число 11. У третьому – обидва фактичних параметри є змінними `x` та `y`.

Якщо функції виконують певні обчислення й дії, які не потребують повернення результатів, за їхній тип вказують тип `void` (тобто порожній, без типу). У таких функціях оператор `return` може бути відсутнім або записуватись без значення, яке повертається.

Основні правила організації функцій

1) Кількість фактичних параметрів функції має збігатися з кількістю її формальних параметрів.

2) Типи змінних, які є фактичними параметрами, мають збігатися з типами відповідних формальних параметрів, або мати можливість бути перетвореними до типів формальних параметрів.

3) Імена змінних, які є фактичними параметрами, можуть не збігатися з іменами формальних параметрів.

4) Фактичними параметрами можуть бути константи, змінні чи вирази.

5) У прототипі функції необов'язково задавати імена формальних параметрів, достатньо їх оголосити, наприклад, у такий спосіб:

```
double seredne(int, int);
```

6) Якщо функція не отримує жодного параметра, слід у заголовку функції ставити порожні дужки (опустити дужки не можна) або записати у дужках `void`:

```
double func();      або      double func(void);
```

7) Тип значення, яке повертає функція, може бути яким-завгодно, але не може бути масивом чи функцією. Наприклад, таке оголошення є помилкове:

```
int [10] func();  // Помилка!
```

8) С++ дозволяє існування у програмі функцій з однаковим ім'ям, але різною кількістю параметрів – перевантажуванні функції.

Способи передавання параметрів до функції. Передавання декількох результатів

Існують способи передавання параметрів до функції: *за значенням* і *за адресою*. При передаванні параметрів (числових змінних) **за значенням** оператори функції працюють з цими числовими значеннями і ніяким чином не можуть повідомити (передати) в головну програму нові значення цих змінних у разі їхнього змінення. За замовчуванням параметри будь-якого типу, крім масиву й функції, передаються до функції за значенням.

При передаванні параметрів **за адресою** передаватимуться не самі значення змінних, а *їхні адреси*. Тобто функція здійснює доступ до комірок пам'яті за цими адресами, а головна програма "бачить" всі змінювання параметрів-змінних так само, як і функція. При цьому існують два різновиди передавання параметрів за адресою: за допомогою *вказівника* (*) і *за посиланням* (&).

При передаванні **вказівника** всередині функції треба використовувати операцію розіменування (розадресації), що робить менш зрозумілим код функції і більш складним налагодження функції. При передаванні **за посиланням** до функції передається посилання, тобто синонім імені, внаслідок чого всередині функції всі звертання до параметра неявно розіменовуються.

Розглянемо приклад функції з передаванням трьох параметрів за значенням, за допомогою вказівника і за посиланням:

```
void f(int i, int* j, int& k)
{ i++; (*j)++; k++;
}
int main()
{ int i=1, j=2, k=3;
  cout << "i  j  k\n";
  cout << i << ' ' << j <<' '<< k << '\n';
  f(i, &j, k);
  cout << i << ' ' << j << ' ' << k;
}
```

Результат роботи цієї програми:

```
i j k
1 2 3
1 3 4
```

Перший параметр `i` передається до функції за значенням, а його змінення у функції не впливає на вихідне значення. Другий параметр `j` передається за адресою за допомогою вказівника, при цьому для передавання до функції адреси фактичного параметра використовується операція отримання адреси `&j`, а для отримання його значення у функції потрібна операція розіменування (розадресації) `*j`. Третій параметр `k` передається за адресою за допомогою посилання. Це дозволяє передавати фактичний параметр `k` без операції отримання адреси і використовувати його у функції без розадресації.

Якщо функція має повернути понад одне значення, їх можна передати параметрами за посиланням, і ті значення, яких вони набудуть у тілі функції, можна "побачити" і в головній програмі (див. приклад програми 7).

При опрацюванні масивів у функціях їх передавання в якості аргументів завжди здійснюється за адресою, при цьому передається адреса першого елемента (початок масиву), а доступ до кожного з елементів масиву здійснюється як певний зсув (обчислюваний через індекси) від початку масиву. При цьому інформація про кількість елементів втрачається і слід передавати його розмірність через окремий параметр. Докладніше опрацювання масивів у функціях буде розглянуто в подальших лабораторних роботах.

Приклади програм з функціями

Приклад 1. Обчислити значення виразу $x = \frac{\sqrt{6}+6}{2} + \frac{\sqrt{13}+13}{2} + \frac{\sqrt{21}+21}{2}$.

Розв'язок. Усі три доданки заданої формули схожі один на одного, кожен з них можна записати за допомогою спільної формули: $\frac{\sqrt{a}+a}{2}$, де a – число, яке у першому доданку дорівнює 6, у другому – 13, а у третьому – 21. Слушним є створення функції обчислення цієї формули, параметром якої буде a, і тричі викликати цю функцію для кожного з доданків: 6, 13, 21.

Текст функції та її виклику в основній програмі:

```
#include <iostream>
#include <math.h>
using namespace std;

double f (double a)
{ return (sqrt(a)+a)/2;
}
int main ()
{ setlocale(0,".1251");
```

```
  double x = f(6) + f(13) + f(21);
  cout<<" Результат x = "<< x << endl;
  system ("pause>>void");
  return 0;
}
```

Результати роботи:

```
Результат x = 25.32
```

Приклад 2. Обчислити значення виразу $x = \frac{13+\sqrt{7}}{7+\sqrt{13}} + \frac{15+\sqrt{12}}{12+\sqrt{15}} + \frac{21+\sqrt{32}}{32+\sqrt{21}}$.

Розв'язок. Формула цього прикладу схожа на формулу попереднього завдання і відрізняється лише тим, що функція тут матиме два параметри.

Текст функції та основної програми:

```
include <iostream>
#include <math.h>
using namespace std;
double f(double a, double b)
{ return (a+sqrt(b))/(b+sqrt(a));
}
int main ()
{
  setlocale(0,".1251");
  double x = f(13,7) + f(15,12) + f(21,32);
  cout<<" Результат x = "<< x << endl;
  system ("pause>>void");
  return 0;
}
```

Результати роботи:

```
Результат x = 3.37
```

Приклад 3. Створити функцію обчислення найбільшого спільного дільника (НСД) двох натуральних чисел та визначити за її допомогою НСД для двох і трьох введених чисел.

Розв'язок. Обчислення НСД можна здійснити різними способами.

Спосіб 1. Розкладання чисел на прості множники (див. функцію **nsd1()**) полягає у простому перебиранні всіх чисел з перевіркою на подільність без залишку обох чисел. Перше число, для якого виконається умова перевірки, і буде НСД. Недоліком цього способу є мала швидкість роботи.

Спосіб 2. Існує декілька алгоритмів Евкліда обчислення НСД пари додатних цілих чисел, які використовують різні рекурентні співвідношення. В самому простому випадку **алгоритм Евкліда** формує нову пару: з меншого числа та різниці між більшим і меншим числами. Процес повторюється, доки числа не стануть однаковими. Знайдене число і є найбільшим спільним дільником вихідної пари (див. функцію **nsd2()**).

Спосіб 3. **Бінарний алгоритм Евкліда** більш швидкий, ніж звичайний алгоритм Евкліда, оскільки замість повільних операцій віднімання виконуються зсуви (див. функції **`nsd3()`** з оператором циклу `while` та **`nsd4()`** – з циклом `do-while`).

Спосіб 4. **Рекурсивний алгоритм Евкліда** (див. функцію **`nsd5()`**) дозволяє позбавитись від циклічного оператора, який організував чергову ітерацію, за рахунок повторного (рекурсивного) виклику цієї самої функції, але вже зі зменшеними на значення зсуву парою чисел.

Наведемо програмний код для всіх названих способів визначення НСД, а в головній програмі організуємо їх виклик для однієї і тієї самої пари введених чисел.

Текст функції та основної програми:

```
#include <iostream>
using namespace std;

int nsd1( int a,  int b)
{
  int n=1;
  if(a > b) n = b; else n = a;
  n++;
  do  n--;
  while( a%n != 0 || b%n != 0);
  return n;
}
int nsd2( int a,  int b)
{
  while( a!= b)
   if (a > b) a -= b; else b -= a;
  return a;
}
int nsd3( int a,  int b)
{
  while ( a && b )
    if ( a>=b ) a %= b;
    else  b %= a;
  return a | b;
}

int nsd4( int a,  int b)
{ int n;
  do
  { n = a % b;
    a = b;
    b = n;
  } while (b > 0);
 return a;
}
```

```
int nsd5( int a,  int b)
{
  if(!b) return a; else return nsd5(b, a % b) ;
 // або ще коротший  запис:   return b ? nsd5 (b, a % b) : a;
}
int main()
{
  system("chcp 1251 > null");
  system("color F0");
  int x, y;
  cout <<" Введіть два натуральних (цілих додатних) числа \n --> ";
  cin >> x >> y;
  if (x<=0 || y<=0) cout << "Некоректні дані!" << endl;
  else
  { cout<<"НСД:  "<< nsd1(x,y) <<endl;
    cout<<"НСД:  "<< nsd2(x,y) <<endl;
    cout<<"НСД:  "<< nsd3(x,y) <<endl;
    cout<<"НСД:  "<< nsd4(x,y) <<endl;
    cout<<"НСД:  "<< nsd5(x,y) <<endl;
  }
  system ("pause>>void");
  return 0;
}
```

Результати роботи:

```
 Введіть два натуральних (цілих додатних) числа
 --> 125   80
НСД:  5
НСД:  5
НСД:  5
НСД:  5
НСД:  5
```

```
 Введіть два натуральних (цілих додатних) числа
 --> 150       500
НСД:  50
НСД:  50
НСД:  50
НСД:  50
НСД:  50
```

Щоб скористатись будь-якою з наведених функцій для обчислення НСД для трьох введених чисел x, y та z в основній програмі, слід викликати її оператором:

```
cout << "НСД:  " << nsd1(x, nsd1(z,y)) <<endl;
```

```
 Введіть три натуральних (цілих додатних) числа
 --> 135       99      450
НСД:  9
```

Приклад 4. Створити функцію, яка записуватиме число у зворотному порядку (наприклад, 1208 → 8021), та "перевернути" за її допомогою три введені числа.

Текст функції та основної програми:

```
#include <iostream>
using namespace std;
int reverse(int n)
{ int a=0;
  do
  { a = a*10 + n%10;
    n = n/10 ;
  } while(n);
  return a;
}
int main()
{ setlocale(0,".1251");
  int x;
  cout <<"Введіть ціле додатне число \n --> ";  cin >> x ;
  cout<<"Перевернуте число:  "<< reverse(x) <<endl;
  system ("pause>>void");
  return 0;
}
```

Результати роботи:

```
Введіть ціле додатне число
-->  14089
Перевернуте число:   98041
```

Приклад 5. Створити функцію, яка переставлятиме місцями першу й останню цифри числа, та за її допомогою змінити таким чином два введені числа.

Текст функції та основної програми:

```
#include <iostream>
using namespace std;

int first_last(int n)
{ int a = n, i = 1, p = n % 10;
  while (n >= 10)
  { i *= 10;
    n /= 10;
  }
  return a - n*i - p + n + p*i;
}
int main()
{ setlocale(0,".1251");
  int x;
  cout <<" Введіть ціле додатне число \n --> ";  cin >> x ;
  cout<<"Перевернуте число:  "<< first_last(x) <<endl;
  system ("pause>>void");
```

```
  return 0;
}
```

Результати роботи:

```
 Введіть ціле додатне число
-->  120569
Перевернуте число:   920561
```

Приклад 6. Обчислити значення виразу $z = \dfrac{2 \cdot 5! + 3 \cdot 8!}{6! + 4!}$.

Розв'язок. У формулі чотири рази зустрічається факторіал. Тому доречно обчислити значення факторіалу у функції і викликати його відповідно чотири рази для різних параметрів.

Текст функції та основної програми:

```
long long fact(int n)
{ long long c=1;
  for(int i=1; i<=n; i++) c*=i;
  return c;
}
int main()
{
  setlocale(0,".1251");
  double z = (2.0*fact(5)+3*fact(8))/(fact(6)+fact(4));
  cout<<" z=  "<< z <<endl;
  system ("pause>>void");
  return 0;
}
```

Результат роботи:

```
z=162.9032
```

Приклад 7. Обчислити за допомогою окремої функції значення виразів $y = \operatorname{arctg}^2 z;\ \ z = \sqrt{x^2 + a};\ \ a = \lg\left|\dfrac{x+b}{e^{b+0.1}}\right|$, значення x та b ввести.

Розв'язок. Обчислити у функції три результуючі значення, одне з яких можна передати через оператор `return`, а решту – за посиланням. Крім того, слід зважити на порядок обчислення виразів: спочатку обчислити значення a, тоді z (яке залежіть від a) і лише згодом – значення y, яке залежіть від z.

Текст функції та основної програми:

```
#include <iostream>
#include <math.h>
using namespace std;
double f(double x, double b, double& a, double& z)
{
  a=log10(fabs((x+b)/exp(b+0.1)));
  z=sqrt(x*x+a);
```

```
  return pow(atan(z),2);
}
int main()
{
  setlocale(0,".1251");
  double x, b, a, z;
  cout <<" Введіть значення x= ";  cin >> x ;
  cout <<" Введіть значення b= ";  cin >> b ;
  cout<<" y=  "<< f(x,b,a,z) <<endl;
  cout<<" a=  "<< a << endl <<" z=  "<< z <<endl;
  system ("pause>>void");
  return 0;
}
```

Результат роботи:

```
Введіть значення x= 2.6
Введіть значення b= 0.3
y= 1.46549
a= 0.28868
z= 2.65494
```

Контрольні запитання для самоконтролю

1) Які дії виконує у функції оператор `return`?
2) Який тип має функція, яка не повертає значення?
3) Яку інформацію про функцію надає такий її заголовок:
   ```
   int fun(double x)
   ```
4) В який спосіб можна передавати з функції декілька результатів?
5) Чи повинні збігатися імена відповідних формальних і фактичних параметрів функції?

Лабораторне завдання

1) У протоколі лабораторної роботи дати відповіді на контрольні запитання.

2) У протоколі лабораторної роботи скласти схеми алгоритмів і написати програми мовою С++, розв’язання завдань яких організувати у функціях. Тобто основна програма повинна містити введення вхідних даних, виклик функції, в якій реалізовано розв’язання, та виведення результатів. Завдання згідно з індивідуальним варіантом вибрати в табл. 8.1 … 8.2.

3) Створити на комп’ютері програмні проєкти мовою С++ для реалізації написаних програм. Занести результати обчислень до протоколу.

Таблиця 8.1

Варіанти завдань для створення функції з одним результатом

№ вар.	Завдання
1	Створити функцію обчислення середнього геометричного трьох чисел за формулою $\sqrt[3]{x \cdot y \cdot z}$ та за допомогою цієї функції визначити для трьох введених чисел значення середнього геометричного
2	Створити функцію визначення найменшого з двох чисел й обчислити для двох введених чисел a та b значення $z = \min(a, 2b) + \min(2a - b, b)$
3	Створити функцію обчислення $\frac{x + \sin y}{\sin x + y}$ та визначити $y = \frac{1+\sin 4}{\sin 1 + 4} + \frac{7+\sin 5}{\sin 7 + 5} + \frac{3+\sin 2}{\sin 3 + 2}$ за допомогою цієї функції
4	Створити функцію обчислення факторіалу та за її допомогою визначити число різних сполучень із n елементів по m ($n > m$) за формулою $C_n^m = \frac{n!}{m!(n-m)!}$
5	Для трикутника з довжинами сторін a, b і c значення медіани для сторони a обчислюється за формулою $0.5\sqrt{2b^2 + 2c^2 - a^2}$. Створити функцію для обчислення медіани та за її допомогою визначити медіани для всіх сторін трикутника
6	Створити функцію обчислення $\frac{\sqrt{x} + x}{y + \sqrt{y}}$ та визначити $\frac{\sqrt{5}+5}{7+\sqrt{7}} + \frac{\sqrt{12}+12}{15+\sqrt{15}} + \frac{\sqrt{22}+22}{32+\sqrt{32}}$ за допомогою цієї функції
7	Створити функцію для обчислення $\mathrm{sh}(x) = (\mathrm{e}^x - \mathrm{e}^{-x})/2$ та за її допомогою визначити $\mathrm{sh}(x)\,\mathrm{tg}(x+1) - \mathrm{ctg}^2(2 + \mathrm{sh}(x-1))$ для введеного значення x
8	Створити функцію визначення найбільшого з двох чисел й обчислити для двох введених чисел a та b значення $y = \max(2a - b, 2b - a) + \max(a, b)$
9	Створити функцію обчислення суми цифр числа та визначити за її допомогою суми цифр для кожного із трьох введених чисел
10	Створити функцію визначення абсолютного значення (за модулем) числа (без використання стандартних математичних функцій) та обчислити за її допомогою модуль значення для трьох введених чисел
11	Створити функцію обчислення степеню натурального числа з натуральним показником (без використання функції pow) та обчислити за її допомогою значення a^x та x^a для введених натуральних чисел a та x
12	Створити функцію обчислення кількості цифр числа та визначити за її допомогою кількість цифр для кожного із трьох введених чисел
13	Створити функцію обчислення третього степеню значення дійсного числа (без використання функції pow) та визначити для трьох введених чисел значення їх кубів за допомогою цієї функції

Продовження табл. 8.1

№ вар.	Завдання
14	Створити функцію обміну значеннями двох дійсних змінних та за її допомогою поміняти місцями введені x та y
15	Створити функцію визначення кількості цифр числа та за її допомогою обчислити розмірності трьох введених чисел
16	Створити функцію обчислення факторіалу та визначити $z = \frac{3 \cdot 7! - 2 \cdot 6!}{5! + 3!}$ за допомогою цієї функції
17	Визначити, якою кількістю способів можна розсадити 4, 5 та 8 студентів у ряд, якщо число всіх перестановок із n елементів обчислюється як $n!$
18	Скількома можливими способами можна вибрати з 6, 7 та 9 людей делегацію у складі 3 осіб, якщо число всіх сполучень із n елементів по k, де $1 \le n \le k$, обчислюється за формулою $C_n^k = \frac{n!}{k!(n-k)!}$
19	Створити функцію обчислення середнього геометричного чотирьох чисел за формулою $\sqrt[4]{a \cdot b \cdot c \cdot d}$ та за допомогою цієї функції визначити для чотирьох введених чисел значення середнього геометричного
20	Створити функцію обчислення $\frac{\sqrt{x} + x}{2}$ та визначити $\frac{\sqrt{5}+5}{2} + \frac{\sqrt{11}+11}{2} + \frac{\sqrt{17}+17}{2}$ за допомогою цієї функції
21	Створити функцію обчислення найбільшого спільного дільника (НСД) двох натуральних чисел за алгоритмом Евкліда та за допомогою неї визначити НСД чотирьох введених чисел
22	Створити функцію обчислення гіпотенузи прямокутного трикутника за значеннями двох катетів та за допомогою цієї функції визначити для двох введених значень катетів значення гіпотенузи
23	Створити функцію обчислення найбільшого спільного дільника (НСД) двох натуральних чисел за алгоритмом розкладання чисел на прості множники та за допомогою неї визначити НСД трьох введених чисел
24	Створити функцію для обчислення $\mathrm{ch}(x) = (\mathrm{e}^x + \mathrm{e}^{-x})/2$ та за її допомогою визначити $\mathrm{ch}(x)\,\mathrm{ctg}^2(x+1) - \mathrm{tg}(0.5 - \mathrm{ch}(x+1))$ для введеного значення x
25	Створити функцію обчислення середнього геометричного двох чисел за формулою $\sqrt{x \cdot y}$ та за допомогою цієї функції визначити для двох введених чисел значення середнього геометричного
26	Створити функцію визначення найбільшої цифри числа та за її допомогою обчислити найбільші цифри для трьох введених чисел
27	Створити функцію обчислення значення катета прямокутного трикутника за значенням гіпотенузи та одного з катетів і за допомогою цієї функції визначити значення катета для двох введених чисел

Закінчення табл. 8.1

№ вар.	Завдання
28	Створити функцію обчислення другого степеня дійсного числа (без використання функції pow) та визначити для трьох введених чисел значення їх квадратів за допомогою цієї функції
29	Створити функцію обчислення $\frac{x-\cos y}{\sin x+y}$ і визначити $z=\frac{2-\cos 3}{\sin 2+3}+\frac{11-\cos 5}{\sin 11+5}-\frac{3-\cos 9}{\sin 3+9}$ за допомогою цієї функції
30	Створити функцію обчислення найбільшого спільного дільника (НСД) двох натуральних чисел за бінарним алгоритмом Евкліда та за допомогою неї визначити НСД п'яти введених чисел

Таблиця 8.2

Варіанти завдань для створення функції з трьома результатами

№ вар.	Обчислювані значення	Вхідні параметри
1	$y=a\sin^2 b+b\cos^2 a;\ a=\sqrt[3]{\lvert b+c\rvert};\ b=\sqrt{x}$	*x, c*
2	$y=a^2+b^2;\ a=\ln\lvert x\rvert;\ b=\mathrm{e}^k+a$	*x, k*
3	$y=\mathrm{e}^x+5.8^c;\ c=a^2+\sqrt{b};\ a=b^3+\ln\lvert b\rvert$	*x, b*
4	$y=\sqrt[3]{\lvert a-b\rvert};\ a=\lg x;\ b=\sqrt{x^2+t^2}$	*x, t*
5	$y=a^3/b^2;\ a=\mathrm{e}^{\sqrt{\lvert x\rvert}};\ b=\left(\sin p^2+x^3\right)$	*x, p*
6	$y=p^2+t^4;\ p=x^2-\sqrt{\lvert x\rvert};\ t=\sqrt[3]{x+a^2}$	*x, a*
7	$y=c^3/\cos c;\ c=a^2+b^2;\ a=\sqrt{\lvert x\rvert}+\mathrm{e}^{\sqrt{b}}$	*x, b*
8	$y=\sin^3(a+b);\ a=t^3+\sqrt{b};\ b=\lg^2\lvert x\rvert$	*x, t*
9	$y=\mathrm{arctg}^3 x^2;\ x=p+k;\ k=\sqrt{p+t^2}$	*t, p*
10	$y=\cos^2(a+\sin b);\ a=\sqrt{\lvert x\rvert};\ b=x^4+m^2$	*m, x*
11	$y=\sin^3 a+\cos^2 x;\ a=c+k^2;\ c=\mathrm{arctg}\lvert x\rvert$	*k, x*
12	$y=\mathrm{e}^{\sqrt{\lvert x\rvert}}+\cos x;\ x=a+c^3;\ a=\sin^5 b$	*b, c*
13	$y=a\cos x-b\sin x;\ x=\sqrt[3]{a-b};\ a=t^2 b$	*t, b*
14	$y=\sqrt{x}\sin a+\sqrt{b}\cos x;\ a=\lg\lvert x\rvert;\ b=x+p^3$	*x, p*
15	$y=\lg a/\lg b;\ a=\sqrt{x^2+b^2};\ x=\mathrm{e}^b+n$	*n, b*
16	$y=\ln\lvert x+t\rvert;\ x=t^2+p;\ t=\sqrt{m}$	*m, p*
17	$y=\mathrm{e}^{a+b};\ a=\lg\lvert t+b^2\rvert;\ t=b^2+\sqrt{bx}$	*b, x*
18	$y=\sqrt[3]{x^2+c^2};\ x=\mathrm{e}^{mk};\ c=\cos^2 m+k^2$	*k, m*
19	$y=\mathrm{e}^x+5.8^c;\ c=a^2+\sqrt{b};\ a=b^3+\ln\lvert b\rvert$	*x, b*

Закінчення табл. 8.2

№ вар.	Обчислювані значення	Вхідні параметри
20	$y = x^3 / t^2;\ x = \mathrm{e}^{\sqrt{p+a}};\ t = p^3 + a^3$	*a, p*
21	$y = c^2 + \sqrt{\lvert a\rvert};\ c = \lg\lvert b\rvert;\ a = (b + x)^3$	*b, x*
22	$y = \mathrm{arctg}^2\lvert x\rvert;\ x = t^3 + b^2;\ t = b^3 + \mathrm{e}^{\sqrt{q}}$	*q, b*
23	$y = v^3 + \cos^2 w;\ v = \cos^2 a;\ w = \sqrt{a + \lvert x\rvert}$	*x, a*
24	$y = x^2 + \sqrt[3]{\lvert x\rvert};\ x = \cos^2 b + \sin^2 a;\ a = \sqrt{b + t^2}$	*b, t*
25	$y = \sin^3 x + \cos x^2;\ x = \lg\lvert ct\rvert;\ c = t^2 + \sqrt{a}$	*t, a*
26	$y = \lg^2\lvert x + a\rvert;\ x = \sqrt{a + b};\ a = \mathrm{e}^{t+b}$	*t, b*
27	$y = \mathrm{arctg}^3\lvert p\rvert;\ p = \sqrt{x^2 + a^2};\ x = \sqrt{a} + \sqrt{b}$	*a, b*
28	$y = \sin^4(a^2 + b^2);\ a = \sqrt{b + t};\ t = b^2 + k^3$	*b, k*
29	$y = \cos^3 x + \lvert a\rvert;\ x = \mathrm{e}^b;\ b = \lvert a\rvert + \sqrt{a + p^2}$	*a, p*
30	$y = \sin^4(a^2 + b^2);\ a = \sqrt{b + t};\ t = b^2 + k^3$	*b, k*

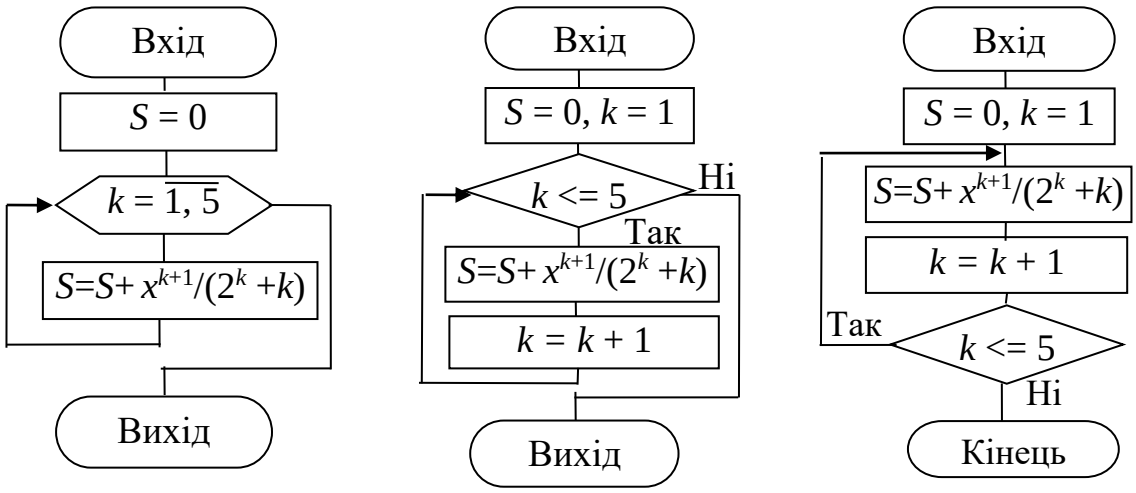
Самостійна робота № 8
Функції

Мета роботи: закріпити практичні навиків розроблення функцій у С++.

Теоретичні відомості

Приклад програм

Приклад 1. Обчислити суму $f(x)=\sum_{k=1}^{5}\frac{x^{k+1}}{2^k+k}$ трьома варіантами, використовуючи різні оператори циклу.



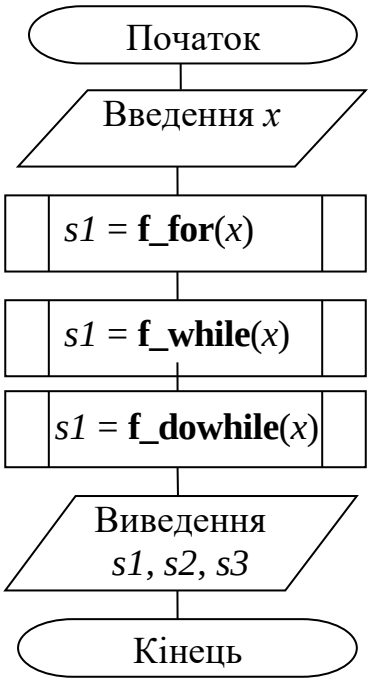
а **б** **в**

Схеми алгоритмів функцій з різними операторами циклу:
а) **for**; б) **while**; в) **do-while**

Програмний код:

```
#include <iostream>
#include <math.h>
using namespace std;

double f_for(double x)
{ double S = 0;
  for (int k=1; k<=5; k++) S += pow(x,k+1)/(pow(2,k)+k);
  return S;
}
double f_while (double x)
{ double S = 0; int k = 1;
  while (k <= 5)
  { S += pow(x,k+1)/(pow(2,k)+k);
    k++;
  }
```

```
  return S;
}
double f_dowhile (double x)
{ double S = 0; int k = 1;
  do
  { S += pow(x,k+1)/(pow(2,k)+k);
    k++;
  }
  while (k <= 5);
  return S;
}
int main ()
{
  setlocale(0,".1251");
  double x, s1, s2, s3;
  s1 = f_for(x);
  s2 = f_while(x);
  s3 = f_dowhile(x);
  cout<<" Ввести x = ";  cin >> x;
  cout<<"for S = "<< s1 << endl;
  cout<<"while S = "<< s2 << endl;
  cout<<"do-while S = "<< s3 << endl;
  system ("pause>>void");
  return 0;
}
```



Результати роботи:

```
Ввести x = 2.7
for S = 28.1871
while S =  28.1871
do-while S = 28.1871
```

Контрольні запитання та завдання для самоконтролю

1) Що таке прототип функції?
2) У чому полягає різниця поміж оголошенням прототипу й визначенням функції?
3) Для яких функцій при їхньому визначенні не використовується ключове слово return?
4) Які параметри функції називають формальними, а які – фактичними?
5) Яку помилку припущено у такій програмі? Як її виправити?

```
#include <iostream>
int main()
{ double x = 5.2;
  cout  << x << " ^ 2 = " << sqr(x);
  return 0;
}
double sqr(double x)
{ return x * x; }
```

Завдання до самостійної роботи

1) Дати відповіді на контрольні запитання.

2) Скласти схеми алгоритмів трьох функцій та основної програми і написати програмний код мовою С++ для обчислення виразу трьома варіантами із застосуванням різних операторів циклу. Вираз вибрати з табл. 8.3.

3) Створити на комп'ютері програмні проєкти мовою С++ для реалізації написаних програм. Занести результати обчислень до протоколу.

Таблиця 8.3

Індивідуальні завдання базового рівня складності

№	Вираз	№	Вираз	№	Вираз
1	$\sum_{k=1}^{7}\frac{2^k\sin(x+k)}{(x+1)^k}$	2	$\sum_{k=1}^{9}\frac{x^{k+1}}{(k+1)^x}$	3	$\sum_{k=1}^{12}\frac{\sin(kx)+k}{\sqrt[k]{x+0.1}+6k}$
4	$\sum_{k=1}^{9}\frac{\ln(x+1)}{(x+k)^k}$	5	$\sum_{k=1}^{9}\frac{\sin(2kx)+0.2}{2k+5}$	6	$\sum_{k=1}^{7}\frac{kx\cos(x+k)}{\ln(2+x)+2k}$
7	$\sum_{k=2}^{6}\frac{\sin(0.17x^k)}{2k+x}$	8	$\sum_{k=1}^{8}\frac{5\ln(2kx)}{\operatorname{arctg}(2x)+k^2}$	9	$\sum_{k=2}^{9}\frac{\operatorname{tg}(x)-x^2/k}{k^2-1}$
10	$\sum_{k=1}^{7}\frac{\sin(x^k-\pi)}{\ln k^2+0.3}$	11	$\sum_{k=1}^{12}\frac{\cos(kx)}{k}$	12	$\sum_{k=1}^{8}\frac{\sin x^k}{4k}$
13	$\sum_{k=1}^{7}\frac{\ln^k(3x)}{(2+x)^k}$	14	$\sum_{k=1}^{8}\sqrt[k]{\ln(x+1)}$	15	$\sum_{k=6}^{1}\frac{x^k}{k^3+x^{k+2}}$
16	$\sum_{k=1}^{11}\frac{\sin(x^k-1)}{4k^2+1}$	17	$\sum_{k=1}^{8}\frac{\ln x^{2k-1}}{2^k(2k-1)}$	18	$\sum_{k=2}^{9}\frac{\operatorname{tg}(e^x)}{3k^2+1}$
19	$\sum_{k=3}^{10}\frac{x^{k-1}\cos x}{12^k-1}$	20	$\sum_{k=3}^{11}\frac{\cos^{2+k}x}{2k-1}$	21	$\sum_{k=1}^{8}\sin^k(x)(k+\cos(x+2))$
22	$\sum_{k=2}^{10}\frac{\operatorname{arctg}^3(2kx)}{1.2\ln(k+x)}$	23	$\sum_{k=1}^{11}\frac{\sin(x)^k+0.3}{(2^k)}$	24	$\sum_{k=1}^{6}\frac{k^2\sin^2(x/k)-kx^2}{e^{kx}}$
25	$\sum_{k=1}^{12}\frac{\cos(x^k)}{(x+5)^k+k}$	26	$\sum_{k=2}^{9}\frac{\sin(x+1)+1.5}{\lg(5kx)+2.1}$	27	$\sum_{k=1}^{7}\frac{2(x+1)^{3-k}}{(k+1)^x+k^3}$
28	$\sum_{k=1}^{10}\cos\left(k^3-\frac{kx}{5}\right)$	29	$\sum_{k=1}^{7}\frac{x\sin(x-k)}{e^{2+x}+k}$	30	$\sum_{k=2}^{6}x\cdot\operatorname{arctg}\frac{x-4.4k}{x+\sin(x+k/5)}$

ТЕМА 4
МАСИВИ

Лабораторна робота № 9
Одновимірні масиви

Мета роботи: набути практичних навиків програмного опрацювання одновимірних масивів.

Теоретичні відомості

Масив у програмуванні – це сукупність однотипних елементів.

При оголошенні масивів у квадратних дужках зазначається кількість елементів, а нумерація елементів завжди розпочинається з нуля.

Одновимірний масив оголошується в програмі таким чином:

<тип_даних> <ім'я_масиву> [*<розмір_масиву>*];

Приклади оголошення масивів:

```
int a[125];   double vector[100];
```

Кожен елемент масиву однозначно можна визначити за ім'ям масиву й індексами. *Індекси* визначають місцезнаходження елемента в масиві і записують після імені масиву в квадратних дужках, тобто використовуючи ім'я масиву та індекс, можна звертатися до елементів масиву:

<ім'я_масиву> [*<значення_ індексу>*]

Значення індексів повинні бути в діапазоні від нуля до величини, на одиницю меншу, ніж розмір масиву, визначений при його оголошенні, оскільки в С++ нумерація індексів розпочинається з нуля.

Наприклад, команда

```
int A[10];
```

оголошує масив з ім'ям `A`, який містить 10 цілих чисел: `A[0]` – перший елемент, `A[1]` – другий, `A[9]` – останній.

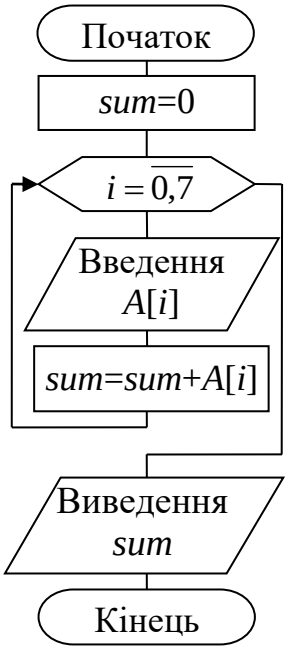
Приклади програм

Приклад 1. Ввести масив з 8 дійсних чисел, обчислити суму елементів масиву.

Текст програми та блок-схема:

```
#include <iostream>
using namespace std;
```

```
int main()
{ setlocale(0,".1251");
  double A[8], sum = 0;
  cout << "Введіть 8 чисел" << endl;
  for (int i=0; i<8; i++)
  { cin >> A[i];     // введення елементів масиву
    sum += A[i];     // обчислення суми елементів
  }
  cout << "Сума елементів масиву дорівнює "
       << sum << endl;
  system ("pause>>void");
  return 0;
}
```



Результати роботи:

Елементи масиву можна вводити через пробіл:

```
Введіть 8 чисел:
1   2  3  4  5  6  7  8
Сума елементів масиву = 36
```

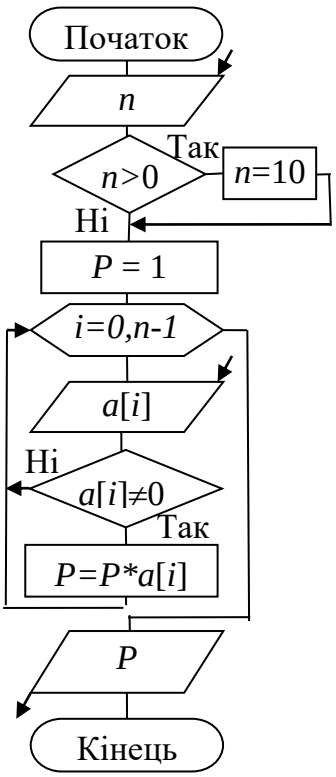
Числа також можна ввести через *Enter*:

```
Введіть 8 чисел:
0.1
-72
3.5
12.7
50
4.6
2
0.2
Сума елементів масиву = 1.1
```

Приклад 2. Ввести масив до 10-ти цілих чисел та обчислити добуток ненульових елементів масиву.

Текст програми та блок-схема:

```
#include <iostream>
using namespace std;
int main()
{ setlocale(0,".1251");
  int A[10], p=1, n;
  cout<<"Введіть кількість елементів масиву (до 10):";
  cin >> n;
  if (n>10) n=10;
  cout<<"Введіть "<< n <<" елементів масиву:\n";
  for (int i=0; i<n; i++)
  { cin >> A[i];
    if (A[i]!=0) p*=A[i];
  }
```



```
  cout<<"p= "<< p <<endl;
  system ("pause>>void");
  return 0;
}
```

Результати роботи:

```
Введіть кількість елементів масиву (до 10): 7
Введіть 7 елементів масиву:
12  9  0  -2  1  0  0
p= -216
```

Приклад 3. Ввести масив із 9-ти дійсних чисел, обчислити найменший елемент масиву та його індекс.

Розв'язок. У вигляді словесного алгоритму пошук мінімального елемента масиву і його індексу можна записати в такий спосіб:

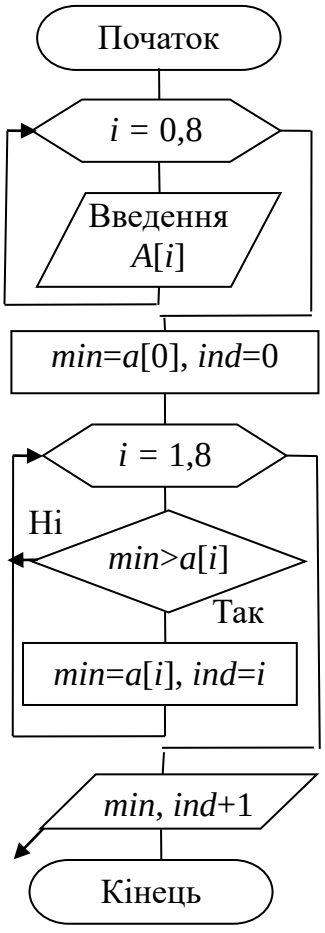
1) вважаємо перше число за найменше. Для цього присвоюємо перший елемент масиву `A[0]` у змінну `min`, а змінній `ind` – значення 0 (тобто індекс першого елемента масиву);

2) у циклі переглядаємо всі елементи. Якщо зустрічаємо число, яке є менше за `min`, запам'ятовуємо це число у `min`, а його індекс – в `ind`.

Коли цикл завершиться, у змінній `min` буде зберігатися найменший елемент масиву, а його індекс – у змінній `ind`. Оскільки нумерація індексів масиву розпочинається з 0, а не з 1, як у повсякденному житті, виведеться значення індексу, збільшене на 1.

Текст програми та блок-схема:

```
#include <iostream>
using namespace std;
int main()
{ setlocale(0,".1251");
  double a[9], min;  int i, ind;
  cout<<"Введіть 9 елементів масиву:\n";
  for (int i=0; i<9; i++)  cin >> A[i];
  min = a[0];  ind = 0;
  for (i=1; i<9; i++)
   if (a[i] < min)
   { min = a[i]; ind = i;  }
  cout<<"min = " << min << "  його індекс = "
      << ind+1 << endl;
  system ("pause>>void");
  return 0;
}
```



Результати роботи:

```
Введіть 9 елементів масиву:
1.2  -2  0  15  -4.9  4  11  102  -3
min = -4.9  його індекс = 5
```

Приклад 4. Ввести масив із 12-ти дійсних чисел і впорядкувати (відсортувати) його за зростанням значень.

Розв'язок. У наведеній нижче програмі запропоновано найбільш поширений метод "бульбашки" (bublesort). Алгоритм сортування полягає у переставлянні двох сусідніх порівнюваних елементів масиву, якщо вони йдуть у неправильному порядку.

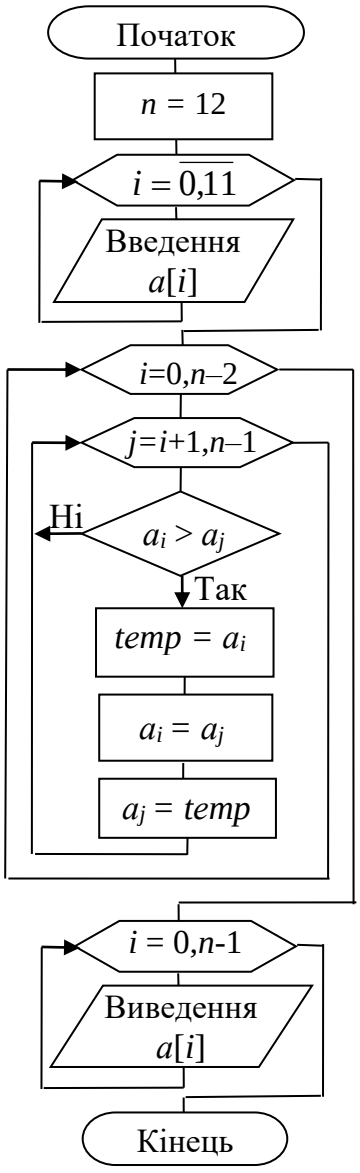
Оскільки одноразового проходження по елементах масиву недостатньо для розв'язання завдання, в програмі організовано два цикли: зовнішній цикл (по `i`) дозволяє перебрати всі елементи масиву для порівняння з іншими елементами, доступ до яких організовано у вкладеному циклі (по `j`). При цьому останнє значення параметра циклу `i` у циклі `for` має бути на один менше за кількість усіх елементів, щоб на останній ітерації можна було порівнювати передостанній елемент з останнім.

Для переставляння двох елементів застосовується тимчасова змінна `temp`:

```
temp = a[i];   // Запам'ятати значення a[i],
a[i] = a[j];   // в a[i] записати a[j],
a[j] = temp;   // а в a[j] записати те, що спочатку було в a[i]
```

Щоб краще зрозуміти необхідність тимчасової змінної, слід уявити собі склянку соку і чашку кави і спробувати перелити сік у чашку, а каву – у склянку. Це неможливо зробити без додаткового посуду, до якого спочатку треба перелити вміст, наприклад, склянки.

Якщо в умові порівняння елементів записати знак ">", то елементи сортуватимуться за зростанням, а якщо знак "<" – за спаданням.



Текст програми та блок-схема:

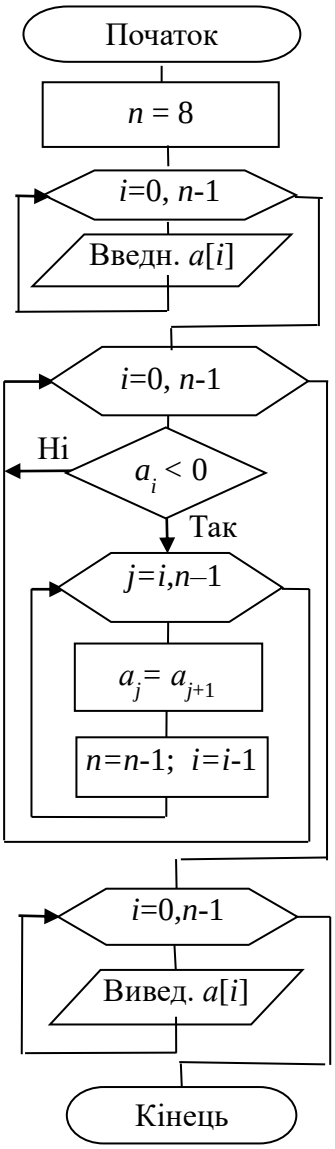
```
#include <iostream>
using namespace std;
int main()
{
 const int n = 12;
 double a[n];  int i;
 cout << "Введіть " << n << " чисел:\n";
 for (i = 0; i < n; i++)
    cin >> a[i];
 for (int i=0; i<n-1; i++)
  for (int j=i+1; j<n; j++)
   if (a[i]>a[j])
   { double temp=a[i];
     a[i]=a[j];
     a[j]=temp;
   }
 cout << "Впорядкований масив:\n";
 for (i = 0; i < n; i++)
   cout << a[i] << " ";
```

```
 cout << endl;
 system("pause");
 return 0;
}
```

Результати роботи:

```
Введіть 12 чисел:
5  0  -2  -6  1  -3  8  9  23  5  -1  3
Впорядкований масив:
-6  -3  -2  -1  0  1  3  5  5  8  9  23
```

Приклад 5. Ввести масив з 8-ми дійсних чисел і видалити з нього всі від'ємні елементи.



Текст програми:

```
#include <iostream>
using namespace std;
int main()
{ setlocale(0,".1251");
 double a[8]; int i,j,n=8;
 cout<<"Введіть масив з 8-ми чисел:\n";
 for (i=0; i<n; i++) cin>>a[i]; //введення масиву
 for (i=0; i<n; i++)
   if (a[i]<0)
   {  // видалення елемента з індексом j
      for (j=i; j<n; j++) a[j]=a[j+1];
      n--;    i--;
   }
  cout<< "\nУ перетвореному масиві " << n
      << " елементи(ів): \n ";
  // виведення масиву
  for(i=0; i<n; i++) cout<<a[i]<<"\t";
  cin.get();  cin.get();
  return 0;
 }
```

Результати роботи:

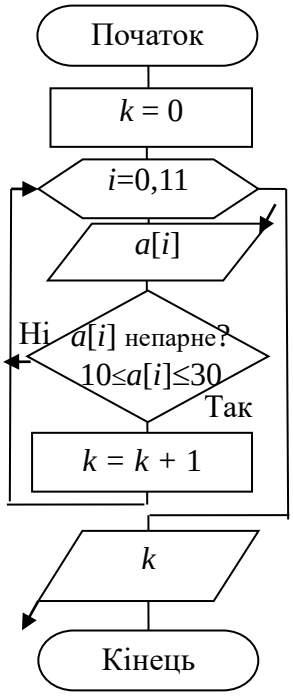
```
Введіть масив з 8-ми чисел:
 -4   3.5  0  -2.3  39  -10  45  9
У перетвореному масиві 5 елементи(ів):
 3.5   0   39   45   9
```

Приклад 6. Ввести масив з 12-ти цілих чисел та обчислити кількість його непарних елементів із діапазону (–10, 30).

Текст програми:

```
#include <iostream>
using namespace std;
```

```
int main()
{
  int a[12], i, k = 0;
  cout << "Введіть 12 цілих чисел:\n";
  for (i = 0; i < 12; i++)
  {
    cin >> a[i];
    if (a[i] % 2 != 0 && a[i] > -10 && a[i] < 30)
      k++;
  }
  cout << "k = " << k << endl;
  system("pause");
  return 0;
}
```



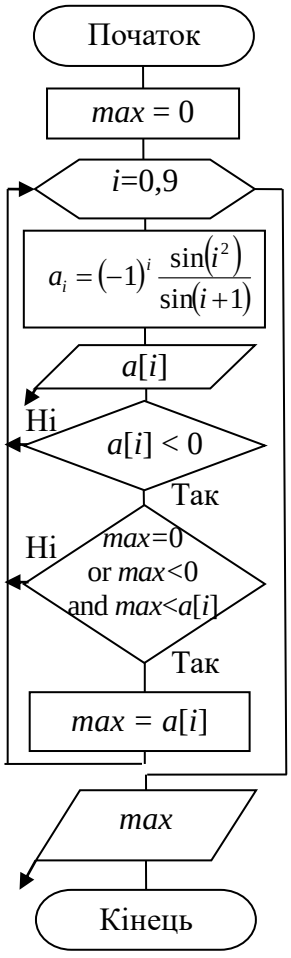
Результати роботи:

```
Enter 12 integer numbers:
29  -817  44  16  -75  1  230  -83  19  -34  8  5
k = 4
```

Приклад 7. Обчислити масив з 10-ти чисел за формулою:

$$a_i = (-1)^i \frac{\sin(i^2)}{\sin(i+1)}$$

та найти найбільший від'ємний елемент масиву.



Текст програми:

```
#include <math.h>
int main()
{
  double a[10], max = 0;  int i;
  printf("Масив: \n");
  for (i = 0; i < 10; i++)
  { a[i] = pow(-1,i) * sin(i*i) / sin(i+1);
    printf("%5.2f ", a[i]);
    if (a[i] < 0)
      if (max == 0 || max < 0 && max < a[i])
        max = a[i];
  }
  printf("\nmax = %5.2f\n", max);
  system("pause");
  return 0;
}
```

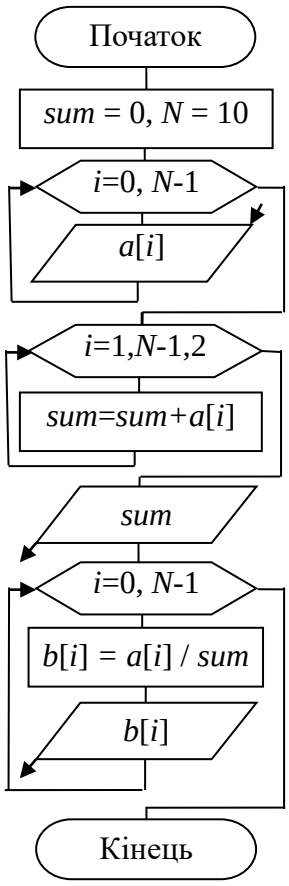
Результати роботи:

```
Масив:
0.00  -0.93  -5.36   0.54  0.30  -0.47  -1.51  0.96  2.23  -1.16
max = -0.47
```

Приклад 8. Ввести масив з 10-ти цілих чисел та створити новий масив з елементами, обчисленими шляхом ділення кожного елемента вихідного масиву на суму його елементів з парними індексами (2, 4, 6, 8, 10).

Текст програми:

```
#include <iostream>
using namespace std;
int main()
{
  const int N = 10;
  int a[N], i, sum = 0;
  double b[N];
  cout << "Ввести 10 цілих чисел:\n";
  for (i = 0; i < N; i++)
    cin >> a[i];
  for (i = 1; i < N; i += 2)
    sum += a[i];
  cout << "Sum = " << sum << endl;
  cout << "Новий масив:\n";
  for (i = 0; i < N; i++)
  { b[i] = 1.*a[i] / sum;
    cout << b[i] << " ";
  }
  cout << endl;
  system("pause");
  return 0;
}
```



Результати роботи:

```
Ввести 10 цілих чисел:
-3  9  12  6  -5  -2  4  7  0  1
Sum = 21
Новий масив:
-0.1428  0.4285  0.5714  0.2857  -0.2380  -0.095  0.1904  0.333  0  0.0476
```

Питання для самоконтролю

1) Дати визначення масиву в програмуванні?
2) Записати оголошення одновимірного масиву з 25-ти дійсних чисел.
3) Навести оператор, який присвоює першому елементу масиву A з 25-ти дійсних чисел нульове значення.
4) Вибрати ім'я змінної, значенням якої є сума елементів масиву, в наведених трьох фрагментах програм:

а)
```
s1=0;
for(i=0;i<10;i++)
  s1 += a[i];
```

б)
```
s2=1;
for(i=0;i<10;i++)
  s2 *= a[i];
```

в)
```
s3=0;
for(i=0;i<10;i++)
  if(a[i]>0) s3++;
```

5) Записати ім'я змінної, значенням якої є кількість додатних елементів масиву, в наведених трьох фрагментах програм:

а)
```
S1=0;
for(i=0;i<10;i++)
 S1 += a[i];
```

б)
```
S2=1;
for(i=0;i<10;i++)
 S2 *= a[i];
```

в)
```
S3=0;
for(i=0;i<10;i++)
 if(a[i]>0) S3++;
```

6) Записати ім'я змінної, значенням якої є максимальний з елементів масиву, в наведених трьох фрагментах програм:

а)
```
S1=0;
for(i=0;i<10;i++)
 S1 += a[i];
```

б)
```
S2=a[0];
for(i=1;i<10;i++)
  if(a[i]>S2)S2=a[i];
```

в)
```
S3=0;
for(i=0;i<10;i++)
  if(a[i]>0)S3=S3+1;
```

Лабораторне завдання

1) У протоколі лабораторної роботи дати відповіді на контрольні запитання.

2) У протоколі лабораторної роботи скласти схему алгоритму і написати програму мовою С++ для розв'язання індивідуальних завдань з опрацювання одновимірних масивів (завдання вибрати з табл. 9.1 … 9.3).

3) Створити на комп'ютері програмні проєкти мовою С++ для реалізації написаних програм. Занести результати обчислень до протоколу.

Таблиця 9.1

Індивідуальні завдання середнього рівня складності

№ вар.	Розмір масиву	Тип даних	Індивідуальне завдання
1	15	цілий	Обчислити кількість та суму парних елементів масиву
2	10	цілий	Обчислити середнє арифметичне додатних елементів
3	8	цілий	Обчислити факторіал значення останнього елемента
4	12	дійсний	Обчислити добуток елементів, значення яких менше 6-ти
5	14	цілий	Обчислити середнє арифметичне непарних елементів
6	18	дійсний	Впорядкувати за спаданням першу половину масиву
7	11	цілий	Розмістити елементи масиву у зворотному порядку
8	14	дійсний	Розмістити елементи масиву у порядку зростання значень їх модулів
9	16	цілий	Обчислити суму елементів масиву, кратних 3
10	14	дійсний	Обчислити суму елементів, абсолютне значення яких не перевищує 10
11	17	цілий	Обчислити середнє арифметичне мінімального та максимального елементів масиву
12	9	дійсний	Вивести кількість елементів, значення яких більше за значення першого елемента масиву
13	15	цілий	Визначити індекси мінімального і максимального елементів масиву
14	10	дійсний	Обчислити суму елементів масиву, значення яких належать проміжку [3, 6]

Закінчення табл. 9.1

№ вар.	Розмір масиву	Тип даних	Індивідуальне завдання
15	8	цілий	Обчислити добуток непарних елементів масиву
16	12	дійсний	Визначити номери мінімального і максимального елементів масиву
17	20	цілий	Впорядкувати за зростанням другу половину масиву
18	18	цілий	Обчислити середнє арифметичне елементів, значення яких більше значення останнього елемента масиву
19	11	дійсний	Обчислити кількість додатних елементів, значення яких менше 20-ти
20	9	дійсний	Обчислити модуль суми всіх від'ємних елементів, суму всіх додатних і різницю між значеннями цих сум
21	16	цілий	Обчислити середнє арифметичне елементів, кратних 5
22	19	дійсний	Визначити мінімальний із додатних елементів
23	17	цілий	Обчислити кількість додатних, від'ємних і нульових елементів
24	8	дійсний	Обчислити добуток одноцифрових елементів масиву
25	7	дійсний	Обчислити суму тільки двоцифрових елементів
26	18	дійсний	Визначити максимальний і мінімальний елементи, наскільки максимальний елемент є більшим за мінімальний
27	11	цілий	Обчислити відсотковий вміст додатних, від'ємних і нульових елементів
28	10	дійсний	Обчислити суму квадратів тих чисел, модуль яких більше значення 2.5
29	12	дійсний	Визначити найбільший з парних додатних елементів
30	9	дійсний	Обчислити суму модулів усіх від'ємних елементів масиву

Таблиця 9.2

Індивідуальні завдання базового рівня складності

№ вар.	Розмір масиву	Тип даних	Індивідуальне завдання
1	15	дійсний	Поміняти місцями мінімальний елемент з передостаннім
2	10	цілий	Обчислити суму додатних непарних елементів і замінити парні елементи масиву на цю суму
3	12	цілий	Перевірити, чи є масив упорядкованим за зростанням
4	8	дійсний	Обчислити факторіал першого елемента масиву, значення якого менше 8-ми
5	14	цілий	Поміняти місцями максимальний елемент з першим
6	18	дійсний	Обчислити новий масив як різницю елементів вихідного масиву та їх середнього арифметичного
7	11	цілий	Замінити всі від'ємні елементи мінімальним
8	14	дійсний	Обчислити кількість елементів, що перевищують середнє арифметичне всіх елементів

Закінчення табл. 9.2

№ вар.	Розмір масиву	Тип даних	Індивідуальне завдання
9	7	цілий	Створити новий масив, розмістити спочатку всі додатні елементи і нулі, після чого – від'ємні, зберігаючи порядок їх слідування
10	19	дійсний	Поміняти місцями максимальний елемент з мінімальним
11	17	цілий	Обчислити факторіал індексу максимального елемента
12	9	дійсний	Замінити мінімальний і максимальний елементи значенням середнього арифметичного всіх елементів
13	15	цілий	Поміняти місцями першу половину масиву з другою
14	10	цілий	Замінити парні за значенням (не за індексом) елементи на 0
15	8	дійсний	Поміняти місцями елементи, які стоять у масиві поряд: 1 і 2, 3 і 4 і т. д.
16	12	дійсний	Видалити з масиву перші три елементи масиву
17	20	цілий	Визначити найменший серед парних додатних елементів
18	18	дійсний	Видалити з масиву всі елементи від початку і до найбільшого елемента
19	11	цілий	Видалити з масиву парні за значенням елементи масиву
20	9	цілий	Створити новий масив із залишків від ділення націло елементів масиву на 3
21	16	цілий	Визначити найменший серед від'ємних елементів масиву
22	19	дійсний	Замінити всі нульові елементи значенням мінімального елемента
23	17	цілий	Обчислити суму від'ємних елементів, розміщених після максимального елемента масиву
24	8	дійсний	Замінити всі від'ємні елементи на нулі, а додатні – на одиниці
25	7	цілий	Замінити всі непарні елементи масиву одиницями
26	18	цілий	Обчислити кількість елементів масиву, розміщених після першого нульового елемента
27	11	цілий	Видалити з масиву нульові елементи масиву
28	10	дійсний	Видалити з масиву найменший елемент масиву
29	16	дійсний	Видалити з масиву перший та останній елементи масиву
30	12	цілий	Обчислити суму парних елементів, розміщених після мінімального елемента масиву

Таблиця 9.3

Індивідуальні завдання високого рівня складності

№ вар.	Розмір масиву	Тип даних	Індивідуальне завдання
1	до 9	цілий	Ввести два масиви й обчислити кількість однакових елементів у них
2	до 10	дійсний	Ввести два масиви і побудувати третій з упорядкованих за зростанням значень елементів обох масивів
3	до 12	цілий	Ввести масив, в якому кожний елемент є 0, 1 або 2, переставити елементи масиву так, щоб спочатку були розміщені всі 1, потім всі 0 і, наприкінці, всі 2
4	до 8	дійсний	Ввести масив і число С. Переставити числа в масиві так, щоб спочатку були розміщені всі елементи менші за значення С, потім – більші С, зберігаючи порядок їх розміщення
5	до 14	цілий	Визначити перше число, яке є присутнім у кожному з 3-х масивів, значення в цих масивах розміщено за зростанням
6	до 7	дійсний	Ввести два масиви та створити третій зі спільних елементів масивів
7	до 11	цілий	Ввести масив, в якому тільки два однакові елементи. Визначити їхні індекси
8	до 14	дійсний	Ввести масив і число *L*. Створити новий масив з елементів, менших за *L*, і впорядкувати новий масив за спаданням
9	до 19	цілий	Визначити пару сусідніх елементів масиву, значення яких є найближчими один до одного, тобто значення $\lvert x_{i+1} - x_i \rvert$ є мінімальним
10	до 12	дійсний	Обчислити добуток *P* від'ємних елементів з парними індексами, які не перевищують введеного числа *L*, і поділити всі додатні елементи на *P*
11	до 17	цілий	Ввести масив і створити на його основі два нових масиви: перший з елементів з непарними індексами, другий – з елементів, кратних 5
12	до 9	дійсний	Ввести масив і число *P*. Визначити елемент масиву, значення якого найближче до *P*, тобто значення $\lvert x_i - P \rvert$ є мінімальним
13	до 15	цілий	Впорядкувати масив так, щоб всі додатні числа були розміщені спочатку за зростанням, а всі від'ємні – наприкінці за спаданням
14	до 10	дійсний	Ввести два масиви і визначити кількість неоднакових елементів у них
15	до 8	цілий	Ввести два масиви та замінити нулями ті елементи першого масиву, яких немає у другому

Закінчення табл. 9.3

№ вар.	Розмір масиву	Тип даних	Індивідуальне завдання
16	до 12	дійсний	Обчислити суму *S* від'ємних елементів, які не перевищують заданого числа *L*, і поділити останній додатний елемент на *S*
17	до 20	цілий	Ввести масив і число *K*. Створити два нових масиви: перший – з елементів, менших за значення *K*, другий – з елементів, більших за *K*, зберігаючи порядок їх розміщення
18	до 18	дійсний	Ввести масив, який містить багато нульових елементів. Замінити всі групи підряд розміщених нулів на значення кількості нулів
19	до 11	цілий	Обчислити суму *S* додатних непарних елементів і замінити всі парні елементи масиву на *S*
20	до 9	дійсний	Ввести два масиви, визначити максимальні елементи кожного з них і поміняти їх місцями
21	до 16	цілий	Ввести масив і на його основі створити новий з від'ємних і нульових елементів з парними індексами
22	до 19	дійсний	Ввести два масиви, визначити мінімальний елемент першого і максимальний елемент другого масиву, і поміняти їх місцями
23	до 17	цілий	Ввести масив і на його основі створити два нових масиви: перший – з парних елементів, другий – з елементів, кратних 3
24	до 8	дійсний	Ввести два масиви і замінити нулями ті елементи другого масиву, які є в першому
25	до 7	цілий	Ввести два масиви і поміняти місцями максимальний елемент першого масиву з першим елементом другого, який більше 5-ти
26	до 18	дійсний	Ввести масив і на його основі створити новий з додатних елементів, більших за середнє арифметичне всіх елементів
27	до 15	цілий	Ввести масив і число *L*. Створити новий масив з елементів, більших за *L*, і впорядкувати створений масив за зростанням
28	до 10	дійсний	Визначити пару сусідніх елементів масиву, значення яких максимально різняться один від одного, тобто значення $\lvert x_{i+1} - x_i \rvert$ є максимальним
29	до 16	цілий	Ввести масив, який містить багато нульових елементів. Замінити всі групи підряд розміщених нулів на один нуль
30	до 12	дійсний	Ввести два масиви, визначити мінімальні елементи кожного з них і поміняти їх місцями

Самостійна робота № 9

Опрацювання одновимірних масивів у функціях

Мета роботи: набути практичних навиків програмного опрацювання одновимірних масивів у функціях засобами С++.

Теоретичні відомості

При опрацюванні масивів у функціях передавання їх у якості аргументів завжди здійснюється за адресою, тобто передається адреса першого елемента (початок масиву), а доступ до кожного з елементів масиву здійснюється як певний зсув (обчислюваний через індекси) від початку масиву.

Передавання одновимірних масивів до функцій у якості аргументів можна організувати одним з трьох способів:

```
double fun (int a[10]);
double fun (int a[]);
double fun (int *a);
```

За першим способом явно зазначено кількість елементів масиву. У другій синтаксичній формі константний вираз у квадратних дужках є відсутній. За третього способу передається вказівник як посилання на масив, явно визначений в основній програмі. Оскільки для другого і третього способів інформація про кількість елементів у прототипі відсутня, то такі форми припустимі, коли кількість елементів масиву є глобальними змінними чи константами. Але доцільніше передавати розмірність одновимірного масиву до функції окремим параметром, оскільки це зробить таку функцію більш універсальною, адже вона зможе коректно опрацьовувати масиви з різною кількістю елементів.

```
double fun (int a[], int n);
double fun (int *a, int n);
```

Хоча наведені записи мають різний синтаксис, насправді вони є рівнозначними, оскільки авторами стандарту С для більшої ясності було вирішено, що масив оголошений як параметр функції є **вказівником**. При цьому навіть явно зазначене число у квадратних дужках ігнорується.

Якщо функція для опрацювання елементів масиву не повинна передавати результат як одне значення, а лише переставляє елементи масиву місцями або змінює їхні значення, то таку функцію оголошують з типом результату `void` (нема величини, що повертається). Оскільки сам масив передається до функції за посиланням, то будь-які змінювання значень елементів масиву у функції буде видно і в основній програмі, яка викликає цю функцію (див. приклад програми 3).

Приклади програм

Приклад 1. Створити функцію обчислення середнього арифметичного значення ненульових елементів цілочислового масиву та за допомогою цієї функції визначити середнє арифметичне ненульових елементів для введеного масиву з 10-ти цілих чисел: спочатку для всіх елементів, а тоді для першої половини масиву.

Схеми алгоритму функції та основної програми:

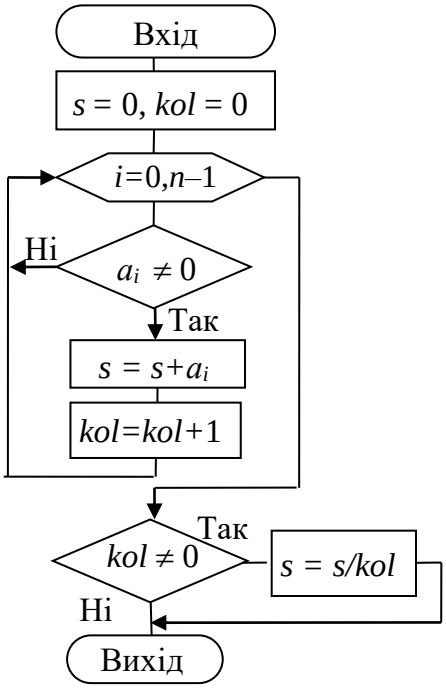


Схема алгоритму функції **sr()**

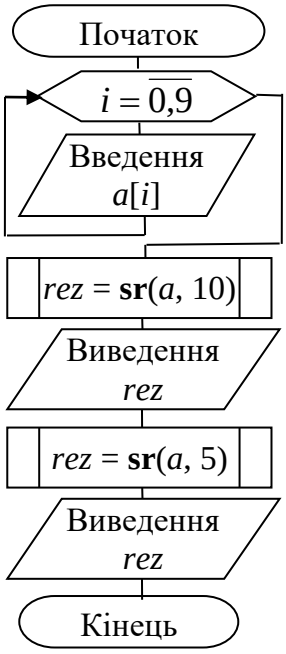


Схема алгоритму до основної програми

Текст програмного коду:

```
#include <iostream>
using namespace std;
double sr(int a[], int n)
{ double s=0; int i, kol=0;
  for (i=0; i<n; i++)
    if (a[i]) { s+=a[i]; kol++; }
  if (kol) s/=kol;
  return s;
}
int main ()
{ setlocale(0,".1251");
  int a[10], i;
  cout<<" Введіть 10 цілих чисел:\n";
  for(i=0; i<10; i++) cin >> a[i];
  cout<<"\n Середнє арифметичне "<<10<<" елементів - " <<sr(a,10) <<endl;
  cout<<" Середнє арифметичне "<< 5 << " елементів - " << sr(a,5) <<endl;
  system ("pause>>void");
  return 0;
}
```

Результати роботи:

```
Введіть 10 цілих чисел:
6   -8   0   3   1   34   -5   1   0   2
Середнє арифметичне 10 елементів –  4.25
Середнє арифметичне 5 елементів –  0.5
```

Приклад 2. Створити функцію визначення максимального і мінімального елементів масиву з 10-ти дійсних чисел та організувати її виклик для введеного масиву.

Схеми алгоритму функції та основної програми:

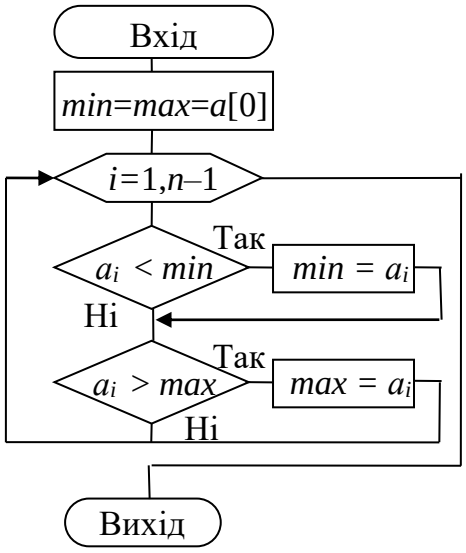


Схема алгоритму функції **MinMax()**

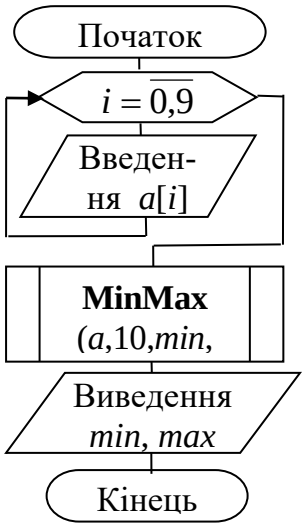


Схема алгоритму до основної програми

Текст функції та основної програми:

```
#include <iostream>
using namespace std;

void MinMax (double a[], int n, double& min, double& max)
{ min=max=a[0];
  for (int i=1; i<n; i++)
  { if (a[i]<min) min=a[i];
    if (a[i]>max) max=a[i];
} }
int main ()
{ setlocale(0,".1251");
  double a[10], min, max;
  cout<<"Введіть 10 чисел:\n";
  for (int i=0; i<10; i++) cin >> a[i];
  MinMax(a, 10, min, max);
  cout<<" Мінімальне значення: "<< min << endl;
  cout<<" Максимальне значення: "<< max << endl;
  system ("pause>>void");
  return 0;
}
```

Результати роботи:

```
Введіть 10 чисел:
1.2   0   -3.8   12   3   34.5   7   0   -54   14

Мінімальне значення:  -54
Максимальне значення:  34.5
```

Приклад 3. Створити функцію для обміну місцями максимального та мінімального елементів масиву і перевірити правильність роботи функції для масиву розмірністю до 12-ти цілих чисел.

Текст функції та основної програми:

```
#include <iostream>
using namespace std;

void change (int a[], int n)
{
  int imin=0, imax=0, tmp;
  for (int i=0; i<n; i++)
  {
    if(a[imin]>a[i]) imin=i;
    if(a[imax]<a[i]) imax=i;
  }
    tmp=a[imin];
    a[imin]=a[imax];
    a[imax]=tmp;
}
int main ()
{
  setlocale(0,".1251");
  int a[12],i,n;
  cout<<"Введіть кількість елементів масиву (до 12): ";
  cin >> n;
  if (n>12) n=12;
  cout<<"Введіть "<< n <<" елементів масиву:\n";
  for (i=0; i<n; i++)  cin >> A[i];
  change (a, n);
  cout<<"Змінений масив:\n";
  for (i=0; i<n; i++) cout << a[i] << "\t";
  cout << endl;
  system ("pause>>void");
  return 0;
}
```

Результати роботи:

```
Введіть кількість елементів масиву (до 12): 5
Введіть 5 елементів масиву:
120  20  0  -60  21
Змінений масив:
-60  20  0  120  21
```

Питання та завдання для самоконтролю

1) Як передаються масиви до функцій у якості їх аргументів?
2) Який тип матиме функція, яка повинна сортувати масив? Обґрунтуйте відповідь.
3) Яку інформацію про функції надають такі їх заголовки:

```
int fun(double x[], int n);
void fun(double x[], int n);
double fun(double x[], int n);
double fun(double x[], int n, double& z);
```

Завдання до самостійної роботи

1) Дати відповіді на контрольні запитання.

2) Скласти схеми алгоритмів функцій та основних програм, а також написати програмний код функцій та основних програм мовою С++ для розв'язання завдань, поданих у табл. 9.4 … 9.5 відповідно до індивідуального варіанта.

3) Створити на комп'ютері програмні проєкти мовою С++ для реалізації написаних програм. Занести результати обчислень до протоколу.

Таблиця 9.4

Варіанти завдань для створення функції з одним або двома результатами

№ вар.	Розмір масиву	Тип даних	Індивідуальне завдання
1	15	цілий	Обчислити факторіал першого елемента масиву, значення якого менше 8-ми
2	10	цілий	Обчислити кількість елементів масиву, розміщених після першого нульового елемента
3	8	цілий	Обчислити факторіал значення першого додатного елемента
4	12	дійсний	Визначити найбільший серед від'ємних елементів масиву
5	14	цілий	Обчислити суму і кількість парних елементів масиву
6	18	дійсний	Обчислити суму елементів, абсолютне значення яких не перевищує 10-ти
7	11	цілий	Обчислити суму та кількість елементів масиву, кратних 3
8	14	дійсний	Обчислити факторіал індексу максимального елемента
9	16	цілий	Обчислити середнє арифметичне мінімального та максимального елементів масиву
10	14	дійсний	Обчислити кількість елементів, значення яких більше за значення першого елемента масиву
11	17	цілий	Обчислити середнє арифметичне елементів, значення яких більше за значення останнього елемента масиву
12	9	дійсний	Визначити індекси мінімального та максимального елементів масиву

Закінчення табл. 9.4

№ вар.	Розмір масиву	Тип даних	Індивідуальне завдання
13	15	цілий	Обчислити модуль суми всіх від'ємних елементів, суму всіх додатних
14	10	дійсний	Обчислити суму від'ємних елементів, розміщених після максимального елемента масиву
15	8	цілий	Визначити мінімальний із додатних елементів
16	12	дійсний	Обчислити різницю між значеннями суми всіх додатних елементів і модулем суми всіх від'ємних елементів
17	20	цілий	Обчислити добуток одноцифрових елементів масиву
18	18	цілий	Обчислити суму елементів масиву, значення яких належать проміжку [3, 6]
19	11	дійсний	Визначити максимальний і мінімальний елементи, наскільки максимальний елемент є більшим за мінімальний?
20	9	дійсний	Обчислити кількість елементів, що перевищують середнє арифметичне всіх елементів
21	16	цілий	Визначити найбільший з парних додатних елементів
22	19	дійсний	Перевірити, чи є масив упорядкованим за зростанням
23	17	цілий	Обчислити суму тільки двоцифрових елементів
24	8	цілий	Обчислити середнє арифметичне елементів, кратних 5
25	7	цілий	Визначити найменший серед парних додатних елементів
26	18	дійсний	Обчислити кількість додатних елементів, значення яких менше 20-ти
27	11	дійсний	Обчислити суму модулів усіх від'ємних елементів масиву
28	10	дійсний	Обчислити суму квадратів тих чисел, модуль яких більше значення 2.5
29	12	цілий	Обчислити суму парних елементів, розміщених після мінімального елемента масиву
30	9	дійсний	Обчислити кількість додатних і від'ємних елементів

Таблиця 9.5

Варіанти завдань створення функції для змінення елементів масиву

№ вар.	Розмір масиву	Тип даних	Індивідуальне завдання
1	15	цілий	Обчислити суму додатних непарних елементів і замінити всі парні елементи масиву на цю суму
2	10	дійсний	Поміняти місцями мінімальний елемент з передостаннім
3	12	цілий	Замінити всі від'ємні елементи на максимальний
4	8	дійсний	Поміняти місцями першу половину масиву з другою
5	14	цілий	Замінити парні за значенням (не за індексом) елементи на 0
6	11	цілий	Обчислити суму додатних непарних елементів і замінити парні елементи масиву на цю суму

Закінчення табл. 9.5

№ вар	Розмір масиву	Тип даних	Індивідуальне завдання
7	18	дійсний	Поміняти місцями елементи, які стоять у масиві поряд: 1 і 2, 3 і 4 і т. д.
8	14	цілий	Поміняти місцями максимальний елемент з першим
9	7	дійсний	Поміняти місцями два найбільші за значенням елементи
10	11	цілий	Впорядкувати за спаданням першу половину масиву
11	12	дійсний	Замінити мінімальний і максимальний елементи значенням середнього арифметичного всіх елементів
12	15	цілий	Замінити всі непарні елементи масиву одиницями
13	9	дійсний	Замінити всі нульові елементи значенням мінімального елемента
14	8	дійсний	Розмістити елементи масиву у зворотному порядку
15	12	цілий	Розмістити елементи масиву у порядку зростання значень їх модулів
16	12	дійсний	Замінити нулями ті елементи масиву, які більші за середнє арифметичне
17	16	дійсний	Впорядкувати за зростанням другу половину масиву
18	14	цілий	Замінити нулями всі елементи від початку і до найбільшого елемента
19	12	цілий	Замінити на максимальне значення масиву всі його нульові елементи
20	16	дійсний	Замінити найменший та найбільший елементи на нулі
21	9	цілий	Поміняти місцями два найменші за значенням елементи
22	16	дійсний	Замінити на значення мінімального елемента ті елементи масиву, які менші за середнє арифметичне
23	10	цілий	Замінити елементи кратні 5 на значення найбільшого непарного елемента
24	18	цілий	Замінити всі елементи кратні двом на значення найбільшого елемента
25	14	дійсний	Поміняти місцями максимальні елементи першої та другої половин масиву
26	12	цілий	Замінити парні за значенням елементи масиву на 0
27	10	цілий	Замінити всі парні елементи масиву на значення останнього елемента масиву
28	11	дійсний	Замінити нульові елементи на середнє арифметичне найменшого і найбільшого елементів
29	12	дійсний	Поміняти місцями мінімальні елементи першої та другої половин масиву
30	16	цілий	Поміняти місцями перший додатний елемент з першим від’ємним. Якщо жодного від’ємного (чи то додатного) елемента в масиві немає, видати про це повідомлення

Лабораторна робота № 10
Двовимірні масиви

Мета роботи: набути практичних навиків програмного опрацювання двовимірних масивів засобами C++.

Теоретичні відомості

1. Організація багатовимірних масивів

Вимірність масиву визначається кількістю індексів. Елементи одновимірного масиву (вектора) мають один індекс, двовимірного масиву (матриці, таблиці) – два індекси: перший з них – номер рядка, другий – номер стовпця. Кількість індексів у масивах є необмежена. При розміщуванні елементів масиву в пам'яті комп'ютера першою чергою змінюється крайній правий індекс, потім решта – справа наліво.

Багатовимірний масив оголошується у програмі в такий спосіб:

<тип> *<ім'я>* [*<розмір1>*] [*<розмір2>*] … [*<розмірN>*];

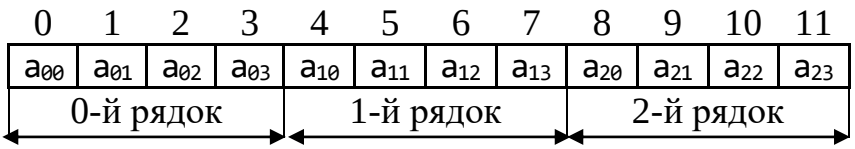
Кількість елементів масиву дорівнює добутку кількості елементів за кожним індексом. Наприклад, матриця з 3-х рядків і 4-х стовпців оголошена як

```
int a[3][4];
```

має 12 елементів цілого типу:

```
a[0][0], a[0][1], a[0][2], a[0][3],
a[1][0], a[1][1], a[1][2], a[1][3],
a[2][0], a[2][1], a[2][2], a[2][3];
```

Під масив надається пам'ять, потрібна для розміщення усіх його елементів. Елементи масиву один за одним, з першого до останнього, запам'ятовуються у послідовно зростаючих адресах пам'яті так само, як і елементи одновимірного масиву. Наприклад, масив `int a[3][4]` зберігається у пам'яті у такий спосіб:



Наведемо ще кілька прикладів оголошення масивів:

```
int x[5][5];    // Матриця 5×5=25 елементів цілого типу
char S[10][3]; // Двовимірний масив з 10×3=30 елементів символьного типу
double z[4][5][4]; //Тривимірний масив з 4×5×4=80 елементів дійсного типу
```

2. Введення-виведення двовимірних масивів

Здійснювати введення-виведення значень елементів масиву можна лише поелементно, для чого слід організовувати цикли, в яких послідовно змінюватимуться значення індексів елементів.

Введення-виведення матриць у консольному режимі

а) **Введення у консолі** матриці a розміром 3×4:

1 спосіб (див. приклад 1):

```
for(i=0; i<3; i++)
for(j=0; j<4; j++)
{ cout<<"Введіть елемент "<<i+1<<"-го рядка "<<j+1<<"-го стовпця: ";
  cin>>a[i][j];
}
```

2 спосіб:

```
for(i=0; i<3; i++)
{ cout<<"Введіть 4 елементи "<<i+1<<"-го рядка "<<endl;
  for(j=0; j<4; j++) cin>>a[i][j];
}
```

3 спосіб (див. приклад 2):

```
cout<<"Введіть матрицю з 3-х рядків і 4-х стовпців:"<<endl;
for(i=0; i<3; i++)
for(j=0; j<4; j++) cin>>a[i][j];
```

б) Для **виведення у консолі** двовимірного масиву a розміром 3×4 у вигляді матриці елементи слід розмістити у рівні стовпці. Це можна зробити за допомогою символу табуляції "\t":

```
for(i=0; i<3; i++)
{
  for(j=0; j<4; j++) cout << a[i][j] << "\t";
  cout << endl;
}
```

При виведенні елементів матриць дійсних чисел командою cout<< доречним є використання маніпуляторів форматування:

– **setprecision**, який обмежує кількість знаків після десяткової крапки;

– **fixed**, який задає формат з фіксованою крапкою;

– **setw**, який дозволяє задавати ширину (мінімальну кількість символьних позицій) кожного виведеного числа. Якщо виведене число має ширину меншу, аніж зазначену у модифікаторі setw, перед ним будуть виведені пробіли:

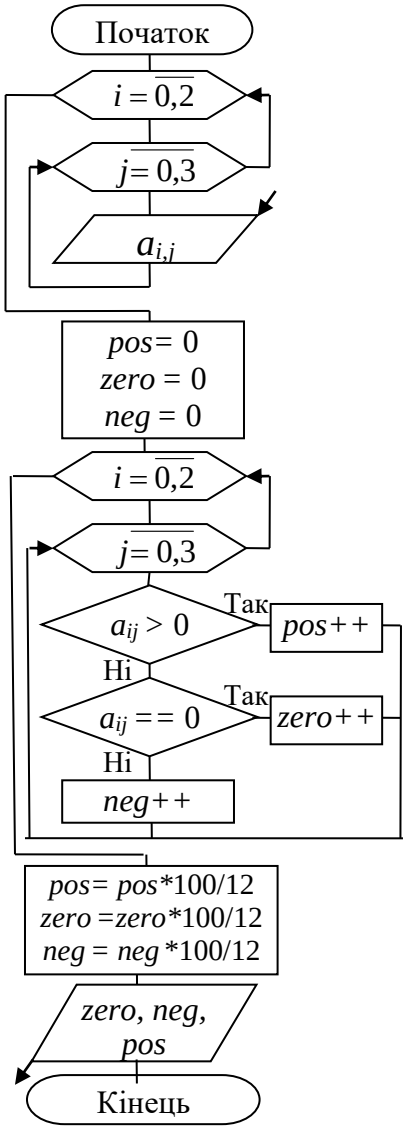
```
for(i=0; i<3; i++)
{ for(j=0; j<6; j++)
    cout<<fixed<<setprecision(2)<<setw(5)<<a[i][j]/3.<<"\t";
  cout << endl;
}
```

Для можливості використання цих маніпуляторів слід долучити заголовний файл iomanip командою #include <iomanip>.

Приклади програм

Приклад 1. Ввести матрицю 3×4 цілих чисел та обчислити відсотковий вміст від'ємних, нульових і додатних елементів.

Текст програмного коду та схема алгоритму:




```
#include <iostream>
//бібліотека для використання маніпуляторів форматування
#include <iomanip>
using namespace std;
int main()
{ setlocale(0,".1251");
  int a[3][4],i,j;
  for(i=0; i<3; i++)
  for(j=0; j<4; j++)
  { cout<<"Введіть елемент "<<i+1<<"-го рядка "
        <<j+1<<"-го стовпця: ";
    cin>>a[i][j];
  }
  double pos,zero,neg;  pos=zero=neg=0;
  for(i=0; i<3; i++)
  for(j=0; j<4; j++)
  { if(a[i][j]>0) pos++;
    else
      if(a[i][j]==0) zero++;
      else neg++;
  }
  pos *= 100.0/12;
  neg *= 100.0/12;
  zero *= 100.0/12;
  cout<<"\nВідсотковий вміст від'ємних елементів - "
      <<setprecision(3) <<neg<<"%"<<endl;
  cout<<"Відсотковий вміст нульових елементів
- "<<zero<<"%"<<endl;
  cout<<"Відсотковий вміст додатних елементів - "<<pos<<"%"<<endl;
  system ("pause>>void");
  return 0;
}
```

Результати роботи:

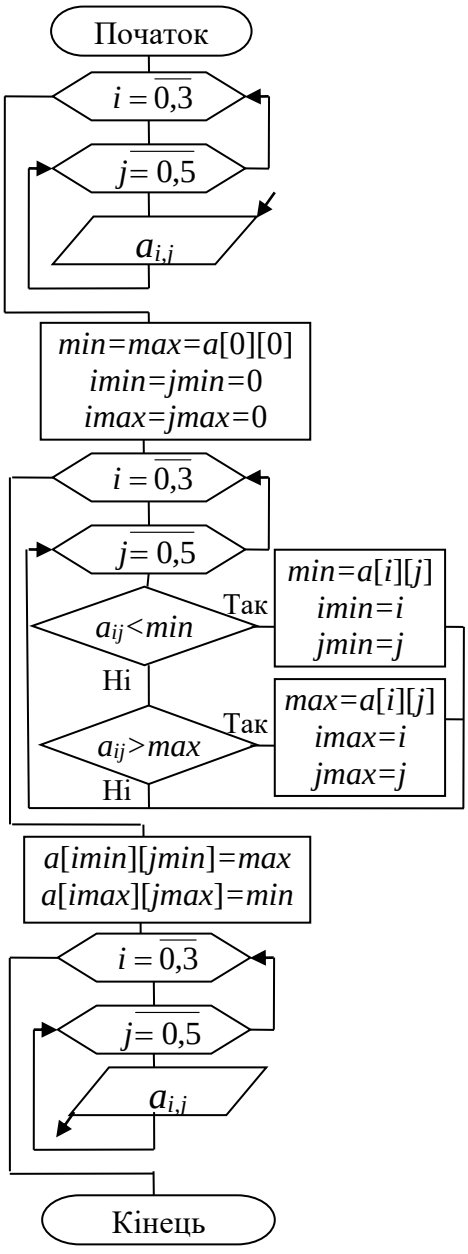
```
Введіть елемент 1-го рядка 1-го стовпця: 23
Введіть елемент 1-го рядка 2-го стовпця: 4
Введіть елемент 1-го рядка 3-го стовпця: 6
Введіть елемент 1-го рядка 4-го стовпця: 0
Введіть елемент 2-го рядка 1-го стовпця: -4
Введіть елемент 2-го рядка 2-го стовпця: -8
Введіть елемент 2-го рядка 3-го стовпця: 23
Введіть елемент 2-го рядка 4-го стовпця: 2
Введіть елемент 3-го рядка 1-го стовпця: 0
Введіть елемент 3-го рядка 2-го стовпця: -12
```

```
Введіть елемент 3-го рядка 3-го стовпця: 1
Введіть елемент 3-го рядка 4-го стовпця: 4

Відсотковий вміст від'ємних елементів – 25%
Відсотковий вміст нульових елементів –  16.7%
Відсотковий вміст додатних елементів -  58.3%
```

Приклад 2. Ввести матрицю 4×6 дійсних чисел і поміняти місцями максимальний та мінімальний елементи.

Програмний код та схема алгоритму:

```
#include <iostream>
using namespace std;
int main()
{ setlocale(0,".1251");
  double a[4][6],min,max;
  int i,j,imin,jmin,imax,jmax;
  cout<<"Введіть матрицю з 4-х рядків і
        6-ти стовпців:"<<endl;
  for(i=0; i<4; i++)
  for(j=0; j<6; j++) cin>>a[i][j];
  min=max=a[0][0]; imin=jmin=imax=jmax=0;
  for(i=0; i<4; i++)
  for(j=0; j<6; j++)
  { if(a[i][j]<min)
     { min=a[i][j]; imin=i; jmin=j; }
    if(a[i][j]>max)
     { max=a[i][j]; imax=i; jmax=j; }
  }
  a[imin][jmin]=max;  a[imax][jmax]=min;
  cout<<"\nМатриця, в якій поміняні місцями
  максимальний і мінімальний елементи:"<<endl;
  for(i=0; i<4; i++)
  { for(j=0; j<6; j++) cout<<a[i][j]<<"\t";
    cout << endl;
  }
  system ("pause>>void");
  return 0;
}
```

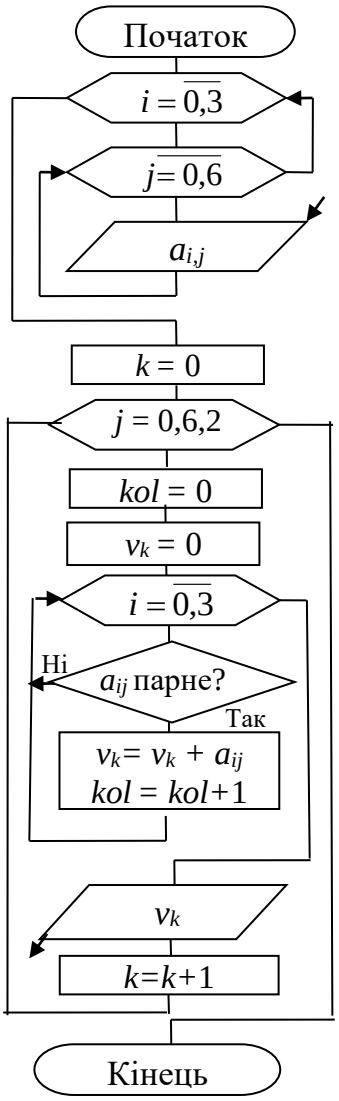


Результати роботи:

```
Введіть матрицю з 4-х рядків і 6-ти стовпців:
23      0       -4      9       2.6     8.1
1       -100    4.6     7       0       9
-3      0       5.7     -0.6    34      10
1       2       3       -4      0       59
Матриця, в якій поміняні місцями максимальний і мінімальний елементи:
23      0       -4      9       2.6     8.1
1       59      4.6     7       0       9
-3      0       5.7     -0.6    34      10
1       2       3       -4      0       -100
```

Приклад 3. З елементів матриці розміром 4×7 цілих чисел сформувати вектор середніх арифметичних парних елементів непарних (1, 3, 5 і 7) стовпців матриці.

Розв'язок. Оскільки при формуванні вектора його індекси не збігаються з індексами непарних стовпців матриці, за якими слід формувати вектор, виникає потреба у використанні додаткової змінної, приміром *k* для індексів вектора.



Текст програмного коду та схема алгоритму:

```
#include <iostream>
using namespace std;
int main()
{ setlocale(0,".1251");
  int i,j;   int a[4][7]; double v[4];
  cout<<"Введіть матрицю з 4-х рядків
        і 7-ми стовпців:" << endl;
  for(i=0; i<4; i++)
  for(j=0; j<7; j++) cin>>a[i][j];
  cout<<"\nВектор середніх арифметичних парних
  елементів непарних (1, 3, 5 та 7) стовпців:"<<endl;
  int kol, k=0;
  for(j=0; j<7; j+=2)
  { kol = 0;
    v[k] = 0;
    for(i=0; i<4; i++)
     if(a[i][j]%2 == 0 && a[i][j])
       { v[k] += a[i][j];
         kol++;
       }
    if(kol) v[k] /= kol;
    cout<< v[k]<<"\t\t";
    k++;
   }
  cout<<endl;
  system ("pause>>void");
  return 0;
}
```

Результати роботи:

```
Введіть матрицю з 4-х рядків і 7-ми стовпців:
1         3       -6      0       9       61      8
2         8       -4      45      3       -1      0
4         -45     13      4       13      8       4
20        6       1       2       7       5       1

Вектор середніх арифметичних парних елементів непарних (1, 3, 5 та 7) стовпців:
8.66667          -5          0           6
```

Приклад 4. З елементів матриці розміром 5×7 цілих чисел сформувати вектор сум від'ємних елементів стовпців матриці.

Текст програмного коду:

```
#include <iostream>
using namespace std;
int main()
{
  int a[5][7], x[7], i, j, s;
  cout << "Input matrix 5x7:"<<endl;
  for (i = 0; i < 5; i++)
    for (j = 0; j < 7; j++)
       cin >> a[i][j];
  cout << "Vector:\n";
  for (j = 0; j < 7; j++)
  {
    s = 0;
    for (i = 0; i < 5; i++)
      if (a[i][j] < 0)
        s += a[i][j];
    x[j] = s;
  }
  for (i = 0; i < 7; i++)
    cout << x[i] << "\t";
  cout << endl;
  system("pause");
  return 0;
}
```

Результати роботи:

```
Input matrix 5x7:
 7   1   0   0   2  -4   1
 4   2  -2   0   0   0   0
-1   2   3   4   5  -1  -1
-2  -4  -5   0  -1  -1   6
 0   0   1   1   2   2   3
Vector:
-3  -4  -7   0  -1  -6  -1
```

Приклад 4. Сформувати із випадкових чисел матрицю цілих чисел розміром 5×10 та обчислити вектор максимальних елементів стовпців матриці.

Текст програмного коду:

```
#include <iostream>
using namespace std;

int main()
{
  int a[5][10], x[10];
```

```
  for (int i = 0; i < 5; i++)
    for (int j = 0; j < 10; j++)
      a[i][j] = rand() % 100 - 50;
  cout << "Matrix: " << endl;
  for (int i = 0; i < 5; i++)
  { for (int j = 0; j < 10; j++)  cout << a[i][j] << "\t";
    cout << endl;
  }
  for (int j = 0; j < 10; j++)
  {
    x[j] = a[0][j];
    for (int i = 0; i < 5; i++)
      if (x[j] < a[i][j])  x[j] = a[i][j];
  }
  cout << "Vector: " << endl;
  for (int j = 0; j < 10; j++)
    cout << x[j] << "\t";
  system("pause");
  return 0;
}
```

Результати роботи:

```
Matrix:
-9     17    -16    -50    19    -26     28    8     12    14
-45    -5     31    -23    11     41     45    -8    -23   -14
 41    -46    -48     3     42     32    -29   -34   -32    45
-3     -24    21     -12    19    -38     17    49    -15    44
-47    -39    -28    -17    23     14     -9    -39    3     18
Vector:
 41    17     31      3     42     41     45    49    12     45
```

Питання та завдання для самоконтролю

1) Які з наведених оголошень двовимірних масивів неправильні й чому?
   а) `int C[1..5, 1..5];` в) `double C[1..5][1..5];`
   б) `double C[5][5];` г) `int C: [5][5];`
2) В який спосіб розміщуються елементи двовимірних масивів в оперативній пам'яті?
3) Записати оператор оголошення матриці цілих чисел `S` розміром 7×3.
4) Під яким номером записано оператор, що обчислює суму елементів головної діагоналі матриці `A` цілих чисел розміром 5×5?
   а) `for(i=0, s=0; i<5; i++) s++;`
   б) `for(i=0, s=0; i<5; i++) s+=A[i][i];`
   в) `for(i=0, s=0; i<5; i++) for(j=0; j<5; j++) s+=A[i][j];`
   г) `for(i=0, s=0; i<5; i++) A[i][i]=0;`
5) Чи можна виконувати опрацювання двовимірного масиву, організувавши зовнішній цикл по стовпцях, а внутрішній – по рядках?

Лабораторне завдання

1) У протоколі лабораторної роботи дати відповіді на контрольні запитання.

2) У протоколі лабораторної роботи скласти схеми та написати програми мовою С++ для розв'язання завдань, поданих в табл. 10.1 … 10.3 відповідно до індивідуального варіанта.

3) Створити на комп'ютері програмні проєкти мовою С++ для реалізації написаних програм. Занести результати обчислень до протоколу.

Таблиця 10.1

Індивідуальні завдання середнього рівня складності

№ вар.	Індивідуальне завдання
1	У матриці дійсних чисел з 5-ти рядків і 4-х стовпців обчислити кількість додатних, від'ємних і нульових елементів
2	У матриці цілих чисел розміром 4×5 визначити найбільший елемент та його індекси
3	Визначити мінімальний елемент головної діагоналі квадратної матриці розміром 5×5 і номер рядка, в якому він міститься
4	Поміняти місцями елементи першого рядка матриці дійсних чисел розміром 4×4 з елементами її неголовної діагоналі
5	У матриці цілих чисел розміром 3×5 замінити від'ємні елементи нулями
6	Визначити максимальний і мінімальний елементи матриці дійсних чисел розміром 6×6
7	Обчислити вектор середньоарифметичних значень елементів рядків матриці дійсних чисел розміром 5×4
8	Обчислити суму елементів неголовної діагоналі матриці 5×5 цілих чисел
9	Для матриці цілих чисел розміром 5×5 обчислити транспоновану матрицю
10	Визначити номер стовпця матриці дійсних чисел розміром 3×6 з найменшим елементом
11	Визначити мінімальний елемент неголовної діагоналі матриці цілих чисел розміром 5×5 і номер стовпця, в якому він розміщений
12	Обчислити вектор сум елементів рядків матриці цілих чисел розміром 7×3
13	Замінити в непарних рядках матриці дійсних чисел розміром 7×4 від'ємні елементи на нулі, а додатні елементи – на одиниці
14	Обчислити суми елементів головної і неголовної діагоналей матриці дійсних чисел розміром 5×5 та різницю між цими сумами
15	У матриці дійсних чисел розміром 7×5 обчислити суму всіх від'ємних елементів перших чотирьох рядків
16	У матриці цілих чисел розміром 4×5 замінити всі від'ємні елементи на нулі
17	Обчислити добуток мінімального елемента матриці цілих чисел розміром 4×5 на значення середнього арифметичного матриці

Закінчення табл. 10.1

№ вар.	Індивідуальне завдання
18	У матриці цілих чисел розміром 6×4 обчислити середнє арифметичне додатних елементів
19	У матриці цілих чисел розміром 5×4 замінити в непарних рядках додатні елементи на 1, а в парних – від'ємні на 0
20	У матриці дійсних чисел розміром 6×3 обчислити добуток всіх від'ємних елементів у парних рядках
21	У матриці цілих чисел розміром 3×5 обчислити кількість елементів, менших за середнє арифметичне
22	У матриці дійсних чисел розміром 5×3 замінити всі елементи, які більші 2.5, на –1
23	Обчислити вектор сум модулів елементів рядків матриці дійсних чисел розміром 4×5
24	У матриці цілих чисел розміром 7×4 визначити найменший елемент з числа додатних і найбільший з числа від'ємних і поміняти їх місцями
25	Обчислити вектор елементів головної діагоналі матриці дійсних чисел розміром 5×5
26	Обчислити вектор сум квадратів елементів стовпців матриці дійсних чисел розміром 3×5
27	У матриці цілих чисел розміром 5×5 поміняти місцями елементи головної та неголовної діагоналей
28	У матриці цілих чисел розміром 5×5 замінити всі парні елементи на нулі
29	Обчислити різницю сум елементів першого рядка й останнього стовпця матриці дійсних чисел розміром 4×6
30	Обчислити вектор сум елементів головної і неголовної діагоналей матриці дійсних чисел розміром 6×6

Таблиця 10.2

Індивідуальні завдання базового рівня складності

№ вар.	Індивідуальне завдання
1	Обчислити вектор сум елементів непарних стовпців матриці 3×7 цілих чисел
2	Обчислити вектор скалярних добутків рядків матриці дійсних чисел розміром 4×4 на її останній стовпець
3	Обчислити вектор добутків непарних елементів парних рядків матриці цілих чисел розміром 6×4
4	Обчислити вектор скалярних добутків елементів першого рядка матриці цілих чисел розміром 4×4 на стовпці цієї матриці
5	Обчислити визначник введеної матриці 3×3 дійсних чисел
6	Обчислити вектор як стовпець матриці дійсних чисел розміром 6×4 з найменшою сумою елементів

Продовження табл. 10.2

№ вар.	Індивідуальне завдання
7	Обчислити вектор добутків парних елементів непарних стовпців матриці цілих чисел розміром 4×5
8	Обчислити вектор як рядок матриці цілих чисел розміром 4×5 з найбільшою сумою елементів
9	Обчислити вектор сум непарних елементів парних рядків матриці цілих чисел розміром 6×6
10	Обчислити вектор скалярних добутків елементів стовпців матриці дійсних чисел розміром 3×3 на її головну діагональ
11	Обчислити вектор з найменших додатних елементів стовпців матриці дійсних чисел розміром 4×6
12	Замінити кутові елементи матриці дійсних чисел розміром 5×6 на значення середнього арифметичного елементів
13	Замінити елементи головної діагоналі матриці цілих чисел розміром 5×5 сумами елементів стовпців
14	Обчислити вектор середньоарифметичних додатних елементів парних рядків матриці цілих чисел розміром 7×8
15	Обчислити вектор з найбільших елементів рядків матриці дійсних чисел розміром 7×6
16	Замінити елементи неголовної діагоналі матриці цілих чисел розміром 4×4 сумами елементів її рядків
17	Обчислити вектор скалярних добутків рядків матриці цілих чисел розміром 5×5 на її неголовну діагональ
18	Обчислити вектор як рядок матриці, який містить найбільший елемент матриці цілих чисел розміром 5×6
19	Замінити елементи головної діагоналі матриці цілих чисел розміром 6×6 сумою максимального і мінімального елементів матриці
20	Замінити елементи неголовної діагоналі матриці дійсних чисел розміром 4×4 значенням мінімального елемента матриці
21	Замінити нульові елементи матриці цілих чисел розміром 5×5 її максимальним елементом
22	У матриці дійсних чисел розміром 7×3 поміняти місцями перший і останній від’ємні елементи
23	Обчислити вектор сум додатних елементів рядків матриці цілих чисел розміром 6×5. Упорядкувати цей вектор за зростанням
24	Серед рядків матриці цілих чисел розміром 5×4 знайти той, для якого сума непарних елементів буде мінімальною, і побудувати з цього рядка вектор
25	У матриці дійсних чисел 6×4 обчислити суму додатних і суму від’ємних елементів, замінити кутові елементи на значення більшої за модулем суми
26	Обчислити вектор скалярних добутків стовпців матриці дійсних чисел розміром 4×4 на її неголовну діагональ

Закінчення табл. 10.2

№ вар.	Індивідуальне завдання
27	Замінити від'ємні елементи матриці дійсних чисел розміром 4×6 на значення середнього арифметичного додатних елементів
28	Обчислити вектор максимальних елементів рядків матриці дійсних чисел розміром 7×5
29	Обчислити вектор як середньоарифметичні значення першого й останнього елементів парних рядків матриці цілих чисел розміром 7×8
30	Обчислити вектор квадратів елементів мінімальних елементів непарних стовпців матриці дійсних чисел розміром 6×5

Таблиця 10.3

Індивідуальні завдання високого рівня складності

№ вар.	Індивідуальне завдання
1	За введеною матрицею дійсних чисел розміром 5×7 обчислити нову матрицю, в якій кожен елемент обчислюється як півсума середньоарифметичних відповідних рядка і стовпця
2	У матриці дійсних чисел розміром 5×5 обчислити визначник
3	Найменший за довжиною рядок матриці дійсних чисел розміром 5×4 замінити на найбільший за довжиною
4	Обчислити добуток матриці 5×5 цілих чисел на її транспоновану матрицю
5	У матриці дійсних чисел розміром 7×3 обчислити номер рядка, довжина якого (як вектора) є максимальною
6	Створити вектор з рядка матриці дійсних чисел розміром 8×3, найменш віддаленого від другого рядка (відстань між рядками обчислюється за формулою $d_i = \sum_{j=0}^{2} \lvert a_{ij} + a_{2j} \rvert$)
7	Ввести матрицю 5×5 цілих чисел від 0 до 9. Якщо кількість повторів елемента матриці збігається з самим елементом, то замінити його на нуль
8	Упорядкувати за зростанням (зліва направо) елементи всіх рядків матриці дійсних чисел розміром 4×4, а тоді за зростанням (зверху вниз) – елементи всіх стовпців
9	Сформувати вектор як рядок матриці дійсних чисел розміром 5×5, найбільш віддалений від першого рядка (відстань від першого рядка матриці до *i*-го обчислюється за формулою $d_i = \sum_{j=0}^{4} \lvert a_{ij} \rvert + \lvert a_{1j} \rvert,\ i \neq 1$)
10	Сформувати вектор як стовпець матриці дійсних чисел розміром 6×8, найменш віддалений від першого (відстань між *j*-м стовпцем і першим обчислюється за формулою $d_j = \sum_{i=0}^{5} \lvert a_{ij} \rvert \cdot \lvert a_{i1} \rvert,\ \ j \neq 1$

Продовження табл. 10.3

№ вар.	Індивідуальне завдання
11	Обчислити вектор як стовпець матриці дійсних чисел розміром 3×7 з найбільшою вагою (вага стовпця матриці обчислюється $W_j = \sum_{i=0}^{2} \lvert a_{ij} \rvert$)
12	Сформувати вектор як стовпець матриці дійсних чисел розміром 8×6 з найбільшою сумою $\sum_{j=0}^{5} \lvert a_{ij} \rvert + a_{ij}$
13	Кожен від'ємний елемент матриці дійсних чисел розміром 4×8 замінити сумою додатних елементів того рядка, в якому розміщений цей елемент
14	Сформувати вектор з рядка матриці дійсних чисел розміром 6×4 з найменшою сумою $\sum_{j=0}^{3} a_{ij}^2$
15	Обчислити вектор як рядок матриці дійсних чисел розміром 5×3 з найменшою вагою (вага рядка матриці обчислюється за формулою $W_i = \sum_{j=0}^{2} \lvert a_{ij} \rvert$)
16	У матриці цілих чисел розміром 6×6 визначити мінімальний елемент у секторі над головної діагоналлю і мінімальний елемент у секторі під головною діагоналлю. Найбільше з цих значень замінити елементи головної діагоналі
17	Ввести матрицю розміром 3×5 з цілих чисел від 0 до 9. Обчислити відсотковий вміст кожного з цих чисел у матриці
18	Обчислити вектор як стовпець матриці дійсних чисел розміром 3×6 з найменшим значенням $W_j = \sqrt{\sum_{i=0}^{5} a_{ij}^2}$
19	Визначити координати елемента матриці дійсних чисел розміром 5×7 з найменшою вагою (вага обчислюється за формулою $W_{ij} = \sum_{i=0}^{4} \sum_{j=0}^{6} \left\lvert \frac{a_{ij}}{i+j} \right\rvert$)
20	У матриці цілих чисел розміром 6×7 визначити рядок з мінімальною сумою і поміняти місцями цей рядок з першим
21	Створити вектор з рядка матриці дійсних чисел розміром 7×5, найбільш віддаленого від третього рядка (відстань між рядками обчислюється за формулою $d_i = \sum_{j=0}^{4} \lvert a_{ij} + a_{3j} \rvert$)
22	Ввести матрицю розміром 6×4 з цілих чисел від 0 до 9. Обчислити вектор з 10-ти елементів як значення кількості повторів цих констант у матриці
23	Якщо в матриці дійсних чисел 4×5 сума додатних чисел більше модуля суми від'ємних, то замінити кутові елементи середнім арифметичним елементів матриці. Інакше видати про це повідомлення
24	Ввести дві матриці дійсних чисел 4×5. Поміняти місцями рядки матриць, які містять максимальні елементи

Закінчення табл. 10.3

№ вар.	Індивідуальне завдання
25	Обчислити вектор як найбільш віддалений стовпець від (n – 1)-го стовпця матриці дійсних чисел розміром 6×5 (відстань обчислюється за формулою $d_j = \sum_{i=0}^{4} \left|a_{ij}\right| + \left|a_{i,n-1}\right|$). Значення n, як і значення елементів матриці, ввести з екрана
26	Ввести матрицю дійсних чисел 5×5 і вектор з п’яти дійсних чисел. Замінити всі рядки матриці, в яких є від’ємні числа, на елементи вектора
27	За матрицею цілих чисел розміром 4×5 обчислити вектор як середні арифметичні значення елементів тих стовпців, які містять максимальний і мінімальний елементи
28	За матрицею дійсних чисел 4×5 сформувати нову матрицю зі значень ваги відповідних елементів (вага обчислюється за формулою $W_{ij} = \sum_{i=0}^{3} \sum_{j=0}^{4} \left\lvert \frac{a_{ij}}{i+j} \right\rvert$)
29	Якщо в матриці цілих чисел 5×6 міститься парна кількість від’ємних елементів, то замінити будь-яку половину з цих чисел на нуль, інакше замінити всі від’ємні елементи на значення їхньої кількості
30	Вивести елемент матриці 6×6 з найбільшою відстанню до діагоналі (відстань обчислюється за формулою $d_{ij} = \sum_{i=0}^{5} \left\lvert \sum_{j=0}^{5} \left|a_{ij} - a_{jj}\right| - a_{ii} \right\rvert$)

Самостійна робота № 10

Опрацювання двовимірних масивів у функціях

Мета роботи: набути практичних навиків програмного опрацювання елементів матриць у функціях засобами С++.

Теоретичні відомості

При передаванні до функції *багатовимірних масивів* усі розмірності, якщо вони є невідомі на етапі компіляції, мають передаватися як параметри. Наприклад, заголовок функції, яка обчислює суму елементів динамічного двовимірного масиву, може мати вигляд:

```
int sum(int **a, int m, int n);
```

Виклик цієї функції у програмі може бути таким:

```
cout << "Сума елементів a: " << sum((int**)a,m,n);
```

Для звичайного статичного двовимірного масиву, коли обидві розмірності є відомі та є константами, заголовок функції матиме вигляд

```
int sum(int a[4][6]);
```

При опрацюванні у функціях як одновимірних, так і двовимірних масивів, які є параметрами цих функцій, насправді до функції передаються не самі масиви, а вказівники на їхні перші елементи.

Розглянемо кілька варіантів передавання матриці цілих чисел `a[3][4]` до функції `print()`, яка організовує виведення цієї матриці на екран у консолі.

1) Якщо розміри обох індексів є відомі, то проблем немає:

```
void print(int a[3][4])
{ for (int i=0; i<3; i++)
  { for (int j=0; j<4; j++) cout << a[i][j] << "\t";
    cout << endl;
} }
```

Матриця і тут передається як вказівник, а розмірності наведено просто для повноти опису. Виклик такої функції може бути таким:

```
int x[3][4];
. . . . . .
print(x);
```

2) Перша розмірність для обчислення адреси елемента є неважлива, тому її можна передавати як параметр:

```
void print(int a[][4], int m)
{ for (int i=0; i<m; i++)
  { for (int j=0; j<4; j++) cout << a[i][j] << "\t";
    cout << '\n';
} }
```

Виклик цієї функції:

```
int x[3][4];
. . . . . .
print(x,3);
```

3) Найскладніший випадок – коли треба передавати обидві розмірності. Наведений нижче код функції є поширеною помилкою:

```
void print(int a[][], int m, int n)  // Помилка!
{ for (int i=0; i<m; i++)
  { for (int j=0; j<n; j++) cout << A[i][j] << "\t";
    cout << '\n';
} }
```

По-перше, опис параметра `a[][]` є неприпустимий, оскільки для обчислення адреси елемента двовимірного масиву слід знати другу розмірність. У такому разі найбільш поширеним і грамотним є застосування динамічних масивів, створювання яких більш докладно розглянуто в подальших лабораторних і практичних заняттях.

Приклади програм

Приклад 1. Ввести матрицю дійсних чисел розмірності 4×3 й обчислити за допомогою функції мінімальний елемент матриці та його індекси.

Розв'язок. Оскільки функція має обчислити і повернути три значення, то один з результатів – мінімальний елемент – буде повернено в головну програму через оператор `return`, а інші два результати – індекси – передаватимуться за посиланням (чи за вказівником).

Схеми алгоритму функції та основної програми:

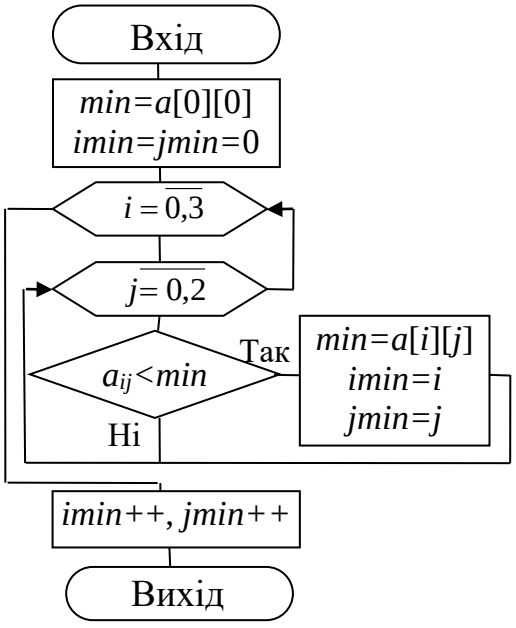


Схема алгоритму функції **MIN ()**

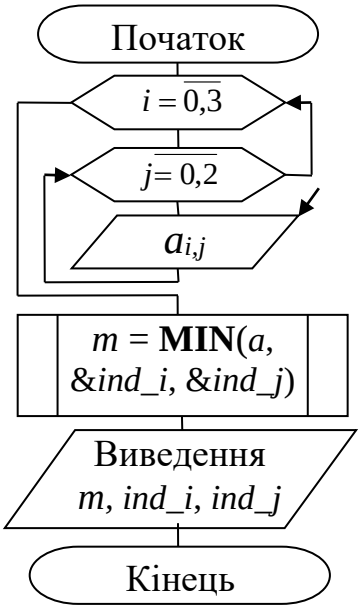


Схема алгоритму до основної програми

Тексти функції та її виклику в основній програмі:

```
#include <iostream>
using namespace std;

double MIN(double a[4][3], int& imin, int& jmin)
{
  double min=a[0][0];
  imin=jmin=0;
  for(int i=0; i<4; i++)
  for(int j=0; j<3; j++)
   if(a[i][j]<min)
   { min=a[i][j];
     imin=i;  jmin=j;
   }
  imin++; jmin++;
  return min;
}
int main ()
{
  setlocale(0,".1251");
  double a[4][3], m;
  int i, j, ind_i, ind_j;
  cout<<" Введіть матрицю з 4-х рядків і 3-х стовпців:"<<endl;
  for(i=0; i<4; i++)
  for(j=0; j<3; j++)
     cin>>a[i][j];
  m = MIN(a, ind_i, ind_j);
  cout<< "\n Мінімальний елемент " << m << "\n розміщено у "
      << ind_i <<"-му рядку і " << ind_j <<"-му стовпці" << endl;
  system ("pause>>void");
  return 0;
}
```

Результати роботи:

```
Введіть матрицю з 4-х рядків і 3-х стовпців:
34      0       4.3
12.7    -7.1    0.3
-5      9       -1
2       -9.4    1.5

Мінімальний елемент -9.4
розміщено у 4-му рядку і 2-му стовпці
```

Приклад 2. Ввести матрицю дійсних чисел розмірності 5×5 і за допомогою функції замінити елементи головної діагоналі на середнє арифметичне відповідного рядка.

Схеми алгоритму функції та основної програми:

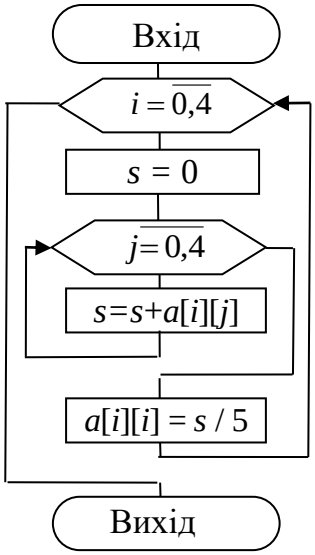


Схема алгоритму функції **zamina ()**

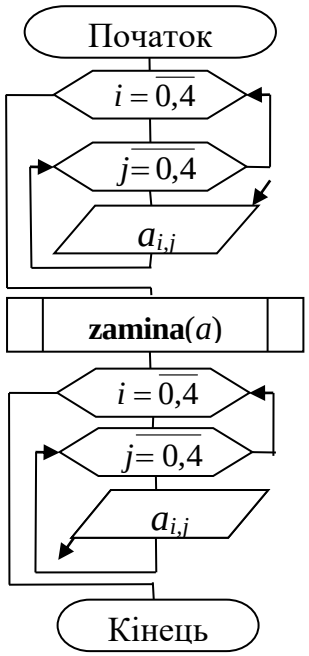


Схема алгоритму до основної програми

Текст функції та основної програми:

```
#include <iostream>
using namespace std;

void zamina(double a[5][5])
{ for(int i=0; i<5; i++)
  { double s=0;
    for(int j=0; j<5; j++) s+=a[i][j]/5;
    a[i][i]=s;
} }
int main ()
{ setlocale(0,".1251");
  double a[5][5];
  int i, j;
  cout<<"Введіть матрицю з 5-ти рядків і 5-ти стовпців:"<<endl;
  for(i=0; i<5; i++)
  for(j=0; j<5; j++) cin >> a[i][j];
  zamina(a);
  cout<< "\nМатриця, в якій елементи головної діагоналі поміняли
          \nна середнє арифметичне відповідного рядка:" << endl;
  for(i=0; i<5; i++)
  { for(j=0; j<5; j++) cout << a[i][j] << "\t";
    cout << endl;
  }
  system ("pause>>void");
  return 0;
}
```

Результати роботи:

```
Введіть матрицю з 5-ти рядків і 5-ти стовпців:
23      6       0       -6.1    45.5
-6      0       1.5     4       -1
6.7     4       99      0       1
9       11      -5      5.8     0.1
3       3.2     7       49      9

Матриця, в якій елементи головної діагоналі поміняли
на середнє арифметичне відповідного рядка:
13.68   6       0       -6.1    45.5
-6      -0.3    1.5     4       -1
6.7     4       22.14   0       1
9       11      -5      4.18    0.1
3       3.2     7       49      14.24
```

Приклад 3. Розробити програму для введення матриці цілих чисел розмірністю 6×3 і створити функцію обчислення елементів вектора як середні арифметичні значення непарних елементів парних рядків.

Розв'язок. У функції `Vektor()` для обчислення елементів вектора спочатку слід організувати цикл для переміщення по парних рядках (2, 4, 6). У кожному з парних рядків, рухаючись по стовпцях, підсумовуємо тільки парні елементи та їхню кількість. Перед обчисленням середнього арифметичного, тобто перед діленням, перевіряємо можливість відсутності в даному парному рядку непарних елементів, тобто виключаємо поділ на нуль. Змінна k є потрібна для послідовного формування індексів вектора, оскільки їхня нумерація не збігається з нумерацією рядків у матриці.

Текст функції та основної програми:

```
#include <iostream>
using namespace std;

void Vektor(int a[6][3], double X[3])
{
  int i, j, kol, k = 0;
  for(i=1; i<6; i+=2)
   { X[k] = kol = 0;
     for(j=0; j<3; j++)
       if(a[i][j] % 2 == 1)
        { X[k] += a[i][j]; kol++;  }
     if(kol) X[k] /= kol ;
     else X[k] = 0;
     k++;
   }
}
int main()
{
  int a[6][3];   double V[3];
  for (int i = 0; i < 6; i++)
    for (int j = 0; j < 3; j++) a[i][j] = rand() % 100 - 50;
```

```
  cout << "Matrix: " << endl;
  for (int i = 0; i < 6; i++)
  { for (int j = 0; j < 3; j++)
       cout << a[i][j] << "\t";
    cout << endl;
  }
  Vektor(a, V);        // Виклик функції Vektor
  cout << "Vector: " << endl;
  for (int j = 0; j < 3; j++)
    cout <<V[j] << "\t";
  system("pause");
  return 0;
}
```

Результати роботи:

```
Matrix:
11   12  5
2    4   9
22  44   3
3    6   7
4   34  56
23  0   -4
Vector:
9   5   23
```

Питання та завдання для самоконтролю

1) Як передаються матриці до функцій у якості аргументів?
2) Який тип матиме функція, яка повинна для матриці цілих чисел:
   а) визначити максимальний елемент?
   б) обчислити середнє арифметичне?
   в) сформувати вектор за певним правилом?
   г) змінити розміщення певних елементів у цій матриці?
3) Які з наведених заголовків функцій є помилковими і чому саме?

```
void fun(int a[][4], int m);
void fun(int a[4][], int m);
void fun(int a[3][4]);
void fun(int a[][], int m, int n);
```

Завдання до самостійної роботи

1) Дати відповіді на контрольні запитання.

2) Скласти схеми алгоритмів функцій та основних програм, а також написати програмний код функцій та основних програм мовою С++ для розв'язання завдань, поданих в табл. 10.4 … 10.6 відповідно до індивідуального варіанта.

3) Створити на комп'ютері програмні проєкти мовою С++ для реалізації написаних програм. Занести результати обчислень до протоколу.

Таблиця 10.4

Варіанти завдань для створення функції з одним або двома результатами

№ вар.	Розмір масиву	Тип даних	Індивідуальне завдання
1	5×5	цілий	Обчислити кількість від'ємних елементів матриці
2	4×4	дійсний	Обчислити суму елементів головної діагоналі матриці
3	6×4	цілий	Визначити найменший елемент матриці
4	3×3	дійсний	Обчислити добуток ненульових елементів матриці
5	4×5	цілий	Обчислити середнє арифметичне мінімального та максимального елементів матриці
6	3×5	дійсний	Обчислити кількість елементів, значення яких більше за значення першого елемента матриці
7	5×3	цілий	Обчислити середнє арифметичне елементів матриці
8	3×4	цілий	Визначити найменший серед парних додатних елементів
9	5×3	цілий	Обчислити суму та кількість парних елементів матриці
10	5×5	дійсний	Обчислити суму та кількість трицифрових елементів
11	4×6	цілий	Обчислити суму та кількість елементів матриці, кратних 3
12	5×4	дійсний	Обчислити модуль суми всіх від'ємних елементів матриці
13	3×5	цілий	Визначити розміщення (індекси) мінімального та максимального елементів матриці
14	4×3	дійсний	Обчислити середнє арифметичне від'ємних елементів
15	6×4	цілий	Обчислити кількість ненульових елементів матриці
16	5×5	дійсний	Визначити максимальний і мінімальний елементи матриці
17	4×5	дійсний	Обчислити середнє арифметичне елементів матриці, значення яких належать проміжку [10, 20]
18	3×5	цілий	Визначити найменший елемент матриці
19	5×3	цілий	Обчислити добуток одноцифрових елементів матриці
20	3×4	дійсний	Обчислити суму модулів всіх від'ємних елементів матриці
21	3×3	цілий	Обчислити суму та кількість двоцифрових елементів
22	5×5	цілий	Обчислити середнє арифметичне елементів, кратних 5
23	4×6	цілий	Визначити найбільший з парних додатних елементів
24	5×4	дійсний	Обчислити суму додатних елементів і кількість від'ємних елементів матриці
25	3×4	цілий	Обчислити суму елементів матриці, значення яких належать проміжку [3, 6]
26	3×3	дійсний	Обчислити визначник матриці
27	6×4	дійсний	Обчислити кількість елементів, що перевищують середнє арифметичне всіх елементів
28	5×5	дійсний	Обчислити середнє арифметичне елементів неголовної діагоналі матриці
29	4×5	цілий	Обчислити суму елементів парних стовпців матриці
30	3×5	дійсний	Визначити мінімальний із додатних елементів матриці

Таблиця 10.5

Варіанти завдань створення функції для змінення елементів матриці

№ вар.	Розмір масиву	Тип даних	Індивідуальне завдання
1	4×3	цілий	Замінити парні за значенням (не за індексом) елементи на 0
2	6×4	дійсний	Поміняти місцями мінімальний і максимальний елементи
3	4×4	цілий	Поміняти місцями елементи головної та неголовної діагоналей матриці
4	4×5	дійсний	Замінити всі від'ємні елементи на значення мінімального
5	3×5	цілий	Обчислити суму додатних непарних елементів і замінити кутові елементи матриці на цю суму
6	5×3	цілий	Замінити всі нульові елементи значенням мінімального елемента
7	5×5	цілий	Транспонувати матрицю
8	5×3	цілий	Замінити всі непарні елементи матриці одиницями
9	3×4	дійсний	Замінити нульові елементи на середнє арифметичне найменшого і найбільшого елементів
10	3×5	цілий	Замінити від'ємні елементи матриці нулями
11	4×6	дійсний	Замінити нулями ті елементи матриці, які більші за середнє арифметичне
12	5×5	цілий	Поміняти місцями елементи першого рядка матриці з елементами її неголовної діагоналі
13	4×4	дійсний	Замінити нулями всі елементи від початку і до найбільшого елемента
14	3×5	дійсний	Замінити найменший та найбільший елементи на нулі
15	4×3	цілий	Обчислити суму додатних непарних елементів і замінити парні елементи масиву на цю суму
16	6×4	дійсний	Замінити мінімальний і максимальний елементи значенням середнього арифметичного всіх елементів
17	5×3	цілий	Замінити парні елементи на значення найменшого елемента
18	4×5	цілий	Замінити парні за значенням елементи матриці на 0
19	3×5	цілий	Поміняти місцями максимальний елемент з першим
20	5×5	дійсний	Розмістити елементи головної діагоналі у зворотному порядку
21	3×4	цілий	Замінити елементи кратні 5-ти на значення найбільшого елемента
22	3×6	дійсний	Поміняти місцями елементи першого й останнього стовпців
23	5×5	цілий	Замінити на максимальне значення матриці всі його нульові елементи
24	4×6	цілий	Замінити всі парні елементи масиву на значення останнього елемента матриці
25	5×4	дійсний	Поміняти місцями елементи першого й останнього рядків
26	3×4	цілий	Поміняти місцями два найбільші за значенням елементи
27	3×3	дійсний	Замінити елементи головної діагоналі матриці на значення середнього арифметичного її елементів

Закінчення табл. 10.5

№ вар.	Розмір масиву	Тип даних	Індивідуальне завдання
28	6×4	цілий	Замінити на значення мінімального елемента ті елементи матриці, які менші за середнє арифметичне
29	5×3	дійсний	Замінити від'ємні елементи в непарних рядках матриці на нулі, а парних рядках – на одиниці
30	4×5	цілий	Поміняти місцями два найменші за значенням елементи

Таблиця 10.6

Варіанти завдань створення функції для формування вектора

№ вар.	Розмір масиву	Тип даних	Індивідуальне завдання
1	3×5	дійсний	Обчислити вектор квадратів елементів мінімальних елементів стовпців матриці
2	5×3	цілий	Обчислити вектор як середньоарифметичні значення кожного рядка матриці
3	3×4	дійсний	Обчислити вектор як суми елементів стовпців матриці, абсолютне значення яких не перевищує 10
4	6×4	цілий	Обчислити вектор як середньоарифметичні значення додатних елементів кожного рядка матриці
5	4×3	дійсний	Обчислити вектор сум додатних елементів рядків матриці
6	5×4	цілий	Обчислити вектор з середньоарифметичних двоцифрових елементів кожного стовпця матриці
7	4×6	дійсний	Обчислити вектор сум елементів непарних стовпців матриці
8	5×5	цілий	Обчислити вектор з елементів головної діагоналі матриці
9	3×4	дійсний	Обчислити вектор квадратів значень останніх елементів стовпців матриці
10	3×5	цілий	Обчислити вектор добутків непарних елементів стовпців
11	4×4	дійсний	Обчислити вектор сум елементів головної і неголовної діагоналей матриці
12	6×4	цілий	Обчислити вектор з найбільших елементів рядків матриці
13	5×3	дійсний	Обчислити вектор сум перших трьох елементів стовпців
14	4×6	цілий	Обчислити вектор сум непарних елементів стовпців матриці
15	3×5	дійсний	Обчислити вектор середньоарифметичних елементів кожного стовпця матриці
16	4×5	цілий	Обчислити вектор добутків одноцифрових (від 0 до 9) елементів стовпців матриці
17	4×4	дійсний	Обчислити вектор елементів неголовної діагоналі матриці
18	3×6	цілий	Обчислити вектор добутків ненульових елементів стовпців матриці
19	5×5	дійсний	Обчислити вектор сум модулів від'ємних елементів рядків
20	4×6	цілий	Обчислити вектор середньоарифметичних першого й останнього елементів кожного рядка матриці

Закінчення табл. 10.6

№ вар.	Розмір масиву	Тип даних	Індивідуальне завдання
21	3×4	дійсний	Обчислити вектор з найменших елементів стовпців матриці
22	5×4	цілий	Обчислити вектор середньоарифметичних парних елементів кожного рядка матриці
23	5×3	дійсний	Обчислити вектор сум модулів елементів рядків матриці
24	6×4	цілий	Обчислити вектор середньоарифметичних першого й останнього елементів кожного стовпця матриці
25	6×6	дійсний	Обчислити вектор кількостей додатних елементів рядків
26	4×5	цілий	Обчислити вектор сум квадратів елементів стовпців матриці
27	5×3	дійсний	Обчислити вектор квадратів значень останніх елементів рядків матриці
28	6×4	цілий	Обчислити вектор добутків парних елементів непарних рядків
29	5×5	дійсний	Обчислити вектор середньоарифметичних елементів головної і неголовної діагоналей матриці
30	6×5	цілий	Обчислити вектор середньоарифметичних елементів максимального і мінімального елементів кожного стовпця

Лабораторна робота № 11
Вказівники і динамічна пам'ять при опрацюванні одновимірних масивів

Мета роботи: набути практичних навиків програмного використання вказівників та динамічної пам’яті при опрацюванні одновимірних масивів.

Теоретичні відомості

1. Вказівники

Вказівник – це змінна, яка містить адресу іншої змінної, при чому вказівник вказує на змінну того типу, адресу якої він містить.

Вказівник оголошується за допомогою *:

*<тип> *<ім’я>;*

Тут *тип* – це базовий тип вказівника, котрим може бути який завгодно тип. *Ім’я* є ідентифікатором змінної-вказівника. Слід звернути увагу на те, що *тип* – це не тип вказівника, а тип даних, адресу яких буде записано у вказівнику.

Наприклад, оголошення вказівників `A` та `B` на ціле число та дійсне числа відповідно:

```
int *A;  double *B;
```

У мові C++ визначено дві операції для роботи з вказівниками:

1) `&` – **адресація**, тобто отримання адреси змінної. Приміром, щоб отримати адресу змінної `x`, оголошеної як `int x=10`, можна скористатись командою `A = &x;`

2) * – **розадресація** чи розіменування вказівника. Щоб взяти значення 10, з адреси у вказівнику `A` слід звернутися `*A`, а щоб змінити це значення, збільшивши його на 1, можна скористатись командою `(*A)++` (при чому команда `*A++` дасть помилкове рішення).

Звернімо увагу на те, що "зірочки" при оголошенні вказівника та операції розадресації – це зовсім різні речі, які лише позначаються однаковим символом.

Ініціалізувати вказівник треба до його використання, тобто присвоїти йому адресу якоїсь змінної або об’єкта. Вказівник, який не вказує на жодне значення, називається порожнім, чи нульовим. Такий вказівник має значення 0 чи NULL.

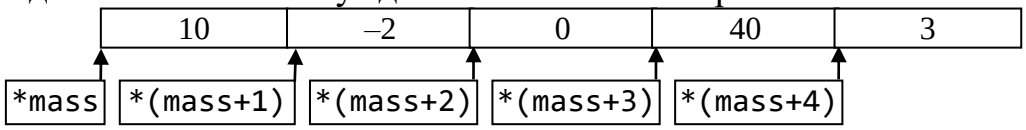
2. Вказівники на одновимірні масиви

Масиви та вказівники у C++ тісно пов’язані і можуть використовуватись майже еквівалентно. Ім’я масиву можна сприймати як константний вказівник на перший елемент масиву. Його відмінність від звичайного вказівника полягає у тому, що його неможна модифікувати.

Здійснимо оголошення масиву `mass` з п’яти цілих чисел з ініціалізацією значень елементів і вказівника на ціле `ptr`:

```
int mass[5] = {10,-2,0,40,3}, *ptr;
```

При такому оголошенні масиву пам'ять виділиться не лише для п'яти елементів масиву, а й для вказівника з ім'ям `mass`, значення якого дорівнюватиме адресі першого елемента масиву `mass[0]`, тобто сам масив залишиться безіменним, а доступ до елементів масиву здійснюватиметься через вказівник з ім'ям `mass`.



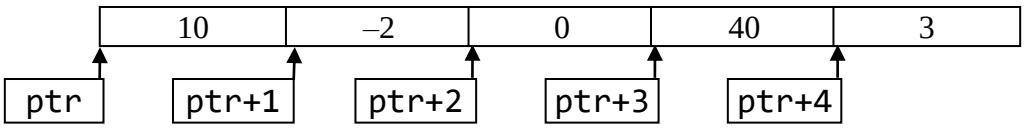
Для того, щоб звернутися до 3-го елемента цього масиву, можна записати чи то `mass[2]` чи `*(mass+2)`. При реалізації на комп'ютері перший з цих способів зводиться до другого, тобто індексний вираз перетвориться до адресного. Оскільки операції над вказівниками виконуються швидше, тому, якщо елементи масиву вибробляються почергово, то доцільніше використовувати другий спосіб. Якщо ж вибір елементів є випадковим, то, щоб уникнути помилок, прийнятнішим є перший спосіб. Крім того, перший спосіб є більш наочним і звичним для сприйняття, що сприяє кращій читабельності програм.

Оскільки ім'я масиву є вказівником на перший елемент масиву, можна надати вказівнику адресу першого елемента масиву за допомогою оператора:

```
ptr = mass;
```

Цей запис є еквівалентним присвоюванню адреси першого елемента масиву (тобто до елемента з нульовим індексом) у вигляді оператора

```
ptr = &mass[0];
```



Після цього звернутися до першого елемента масиву і надати йому значення 2 можна будь-яким з шести операторів:

```
* mass = 2;     mass[0] = 2;       *(mass+0) = 2;
* ptr = 2;      ptr[0] = 2;        *(ptr+0) = 2;
```

Усі ці оператори за дією є тотожними, але швидше за всі виконуватимуться присвоювання `*mass = 2` та `*ptr = 2`, оскільки в них не треба виконувати операції додавання.

Вказівники можна індексувати так само, як і масиви. Наприклад, вираз `ptr[3]` посилається до четвертого елемента масиву `mass[3]`.

Зауважимо, що не слід плутати такі оголошення:

```
int *p1[10];                    // приклад 1
int (*p2)[10];                  // приклад 2
```

У першому прикладі оголошено масив вказівників з ім'ям `p1`. Масив складається з 10-ти елементів, кожний з яких є вказівником на змінну типу `int`.

У другому прикладі оголошено змінну-вказівник з ім'ям `p2`, яка вказує на масив з 10-ти цілих чисел типу `int`.

3. Динамічна пам'ять

Окрім звичайної пам’яті (стека), в якій автоматично розміщуються змінні за їхнього оголошення, існує ще й *динамічна пам’ять* (heap − купа), в якій змінні можуть розміщуватися динамічно. Це означає, що пам’ять виділяється під час виконання програми, й лише тоді, коли у програмі зустрінеться спеціальна інструкція. Основна потреба у динамічному виділенні пам’яті виникає, коли розмір або кількість даних заздалегідь є невідомі, а визначаються в процесі виконання програми.

У C++ існує кілька команд для виділення динамічної пам’яті.

1) Найчастіше для виділення динамічної пам'яті використовується оператор **new**, який у загальному вигляді записується як:

*<тип> *<ім’я_вказівника> =* **new** *<тип>;*

Наприклад, оператор створення *динамічного масиву* з 10-ти елементів:

```
int *a = new int [10];
```

при цьому виділяється пам’ять для 10-ти цілих чисел. Звернутися до кожного з цих чисел можна за його номером: `a[0]`, `a[1]` і т. д. або через вказівник: `*a` − те ж саме, що `a[0]`; `*(a+1)` − те ж саме, що `a[1]` і т. д.

Для звільнення динамічної пам’яті, виділеної за допомогою оператора `new`, використовується оператор **`delete`**:

`delete` *<вказівник>;*

Наприклад:

```
delete []a;
```

Крім операторів `new` та `delete`, існують функції, які перейшли до C++ з C, але вони використовуються на практиці набагато рідше.

2) Виділення динамічної пам'яті функцією `malloc`, яка має такий формат:

```
void *malloc(size_t size);
```

Єдиний аргумент цієї функции `size` − кількість байтів, яку треба виділити. Функція повертає вказівник на початок виділеної пам’яті. Якщо для розміщення заданої кількості байтів є недостатньо пам’яті, функція `malloc()` повертає `NULL`. Вміст ділянки лишається незмінним, тобто там може залишитися “бруд”. Якщо аргумент `size` дорівнює 0, функція повертає `NULL`. Наприклад, команда

```
int *a = (int*) malloc(sizeof(int) * 10);
```

виділяє пам’ять під 10 цілих чисел й адресу початку цієї ділянки пам’яті записує у вказівник `a`.

3) Виділення динамічної пам'яті функцією `calloc`, яка має такий формат:

```
void * calloc(size_t num, size_t size);
```

виділяє блок пам’яті розміром `num×size` (під `num` елементів по `size` байтів кожен) і повертає вказівник на виділений блок. Кожен елемент виділеного блока ініціалізується нульовим значенням (на відміну від функції `malloc`). Функція `calloc()` повертає `NULL`, якщо не вистачає пам’яті для виділення нового блока, або якщо значення `num` чи `size` дорівнюють 0.

Виділення пам’яті під 10 дійсних чисел за допомогою функції `calloc()`:

```
double *b = (double*) calloc(10, sizeof(double));
```

Змінити (зменшити чи то збільшити) розмір виділеного раніш блока динамічної пам’яті можна функцією **realloc**, яка має такий формат:

```
void *realloc(*bl, size);
```

при цьому розмір виділеного раніш блока пам’яті з адресою `*bl` змінюється на новий обсяг, який становитиме `size` байтів. Якщо зміна відбулася успішно, функція `realloc()` повертає вказівник на виділену ділянку пам’яті, а інакше повертається `NULL`. Якщо `*bl` є `NULL`, функція `realloc()` виконує такі самі дії, що й `malloc()`. Якщо `size` є 0, виділений за адресою `*bl` блок пам’яті звільняється – і функція повертає `NULL`. Для ілюстрації роботи функції `realloc()` надамо приклад змінення розміру раніш виділеної пам’яті під одновимірний масив з 10-ти дійсних елементів до 20-ти елементів:

```
a = (int*) realloc(a, 20);
```

Пам’ять, виділену за допомогою функцій `malloc()` і `calloc()`, слід звільнити за допомогою функції `free()`, наприклад:

```
free(a);
```

Якщо не звільняти динамічно виділену пам’ять, коли вона стає більш не потрібною, у системі може виникнути нестача вільної пам’яті. Іноді це називають "витоком пам’яті".

Використання згаданих функцій разом з вказівниками надає можливість керувати розподілом динамічної пам’яті.

4. Доцільність використання вказівників

Правильне розуміння і використання вказівників має велике значення при створенні більшості C++-програм з чотирьох причин:

1) використання вказівників може підвищити ефективність роботи деяких функцій;

2) вказівники надають можливість "бачити" в основній програмі можливі модифікації аргументів у функціях;

3) вказівники використовуються для підтримки системи динамічного виділення пам'яті;

4) вказівники використовуються для підтримки певних структур даних, а саме: зв'язані списки і бінарні дерева.

Крім того, що вказівники – одна з найсильніших сторін С, вони, в той самий час, можуть завдати великої шкоди. Наприклад, неініціалізований чи дикий вказівник може спричинити крах системи, може бути навіть гірше, коли некоректне використання вказівників призводить до невловимих помилок.

Приклади програм

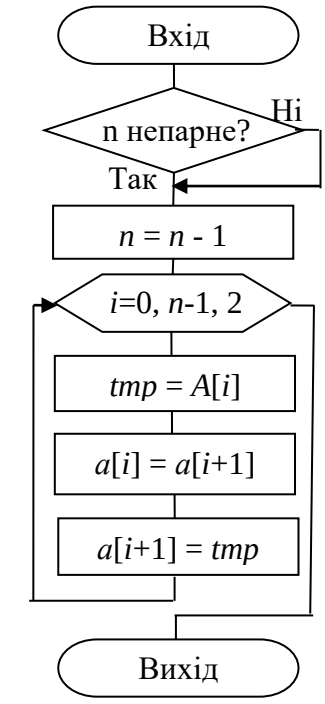
Приклад 1. Ввести деяку послідовність цілих чисел і створити динамічний масив з чисел, розміщених після першого двоцифрового числа (якщо двоцифрових чисел немає, вибрати всі). За допомогою функції поміняти місцями елементи, які стоять поряд: 1 і 2, 3 і 4 тощо.

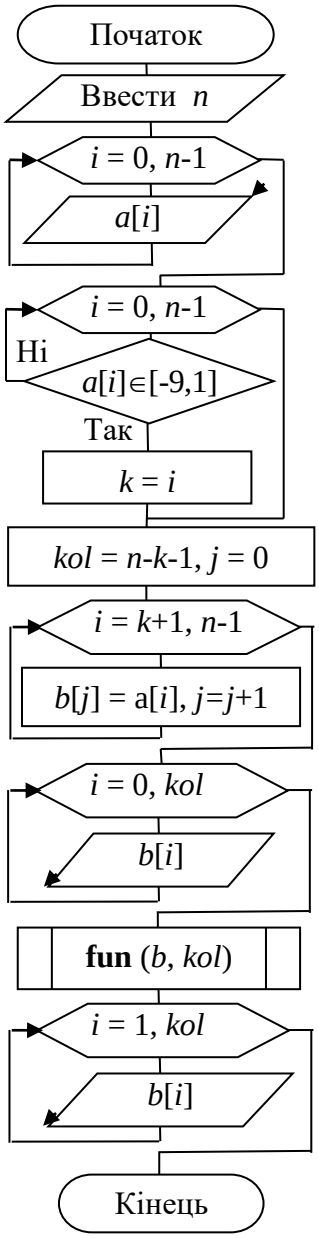
Програмний код:

```
#include <iostream>
using namespace std;

void fun(int a[], int n)
{
  if(n%2) n--; //Якщо кількість елементів непарна,
               //не розглядати останній з них
  for(int i=0; i<n; i+=2)
  { int tmp=a[i];
    a[i]=a[i+1];
    a[i+1]=tmp;
  }
}
int main()
{
  int n = 0, i, j, kol, k = -1;
  cout << "Input n="; cin >> n;
  int * a = new int[n];
  cout << "Input " << n << " integers\n";
  for (i = 0; i < n; i++) cin >> a[i];
  for (i = 0; i < n; i++)
    if (a[i]>-10 && a[i]<0)
    {
      k = i; break;
    }
  kol = n - k - 1;
  int * b = new int[kol];
  for (j = 0, i = k + 1; i < n; i++, j++)
    b[j] = a[i];
  cout << "Dynamic array: \n";
  for (i = 0; i < kol; i++) cout << b[i] << " ";
  fun(b, kol);
  cout<<"\n Dynamic array with swapped elements:\n";
  for (i = 0; i < kol; i++) cout << b[i] << " ";
  delete[]a;  delete[]b;
  system("pause");
  return 0;
}
```

Результати роботи:

```
Input n=10
Input 10 integers
5    -13   7    3    2    10   -9   6    -10   8
Dynamic array:
6 -10 8
Dynamic array with swapped elements:
-10 6 8
```



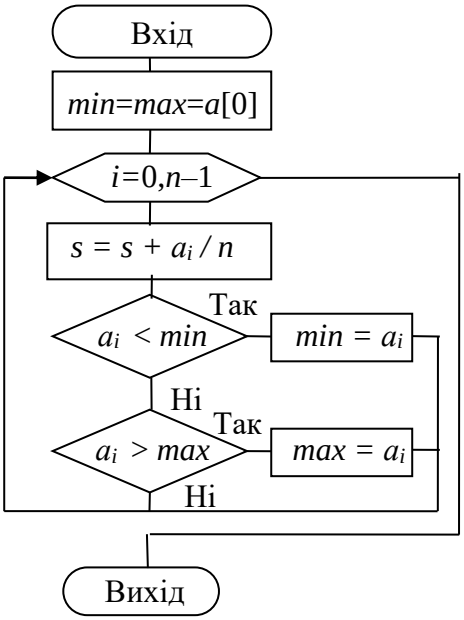


Приклад 2. Ввести деяку послідовність дійсних чисел і створити динамічний масив лише з чисел, значення яких попадає в проміжок [60, 100]. За допомогою функції визначити мінімальний і максимальний елементи та обчислити середнє арифметичне всіх елементів.

Розв'язок. Щодо практичного застосування уявімо, що початкова послідовність чисел є списком екзаменаційних оцінок, з якого слід відібрати лише ті, які задовольняють умові успішної здачі екзамену. А тому доречним є визначення середнього балу успішності та визначення найкращої і найгіршої екзаменаційної оцінки.

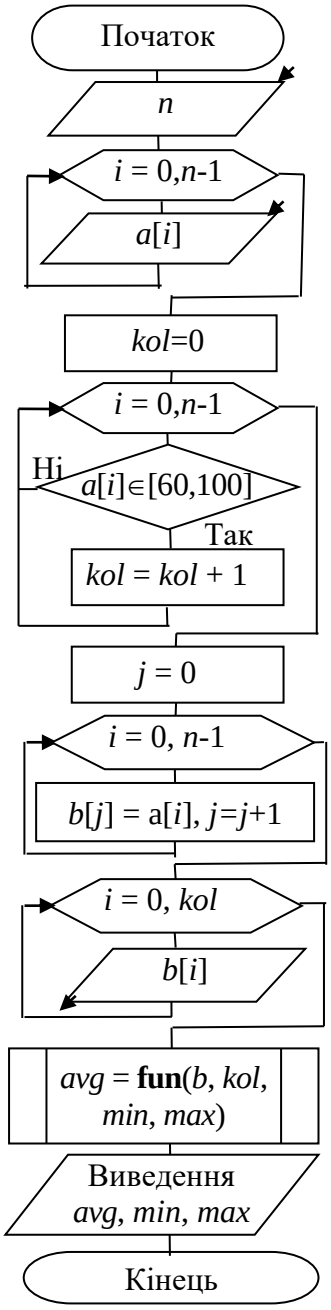
Програмний код:

```
#include <iostream>
using namespace std;

double fun(double a[], int n, double &min, double &max)
{
  double s = 0;
  min = max = a[0];
  for (int i = 0; i < n; i++)
  {
    s += a[i] / n;
    if (min > a[i]) min = a[i];
    if (max < a[i]) max = a[i];
  }
  return s;
}

int main()
{
  int n = 0, i, j, kol = 0;
  double min, max, avg;
  cout<<"Введіть кількість елементів n=";
  cin >> n;
  double * a = new double[n];
  cout<<"Введіть " << n << " чисел\n";
  for (i = 0; i < n; i++)
    cin >> a[i];
  for (i = 0; i < n; i++)
    if (a[i] >= 60 && a[i] <= 100) kol++;
  double * b = new double[kol];
  for (j = 0, i = 0; i < n; i++)
    if (a[i] >= 60 && a[i] <= 100)
    {
      b[j] = a[i];
      j++;
    }
```



Блок-схема функції **fun()**

```
  cout << "Dynamic array: \n";
  for (i = 0; i < kol; i++)
    cout << b[i] << " ";
  cout << endl;
  avg = fun(b, kol, min, max);
  cout << "Average=" << avg << ", min=" << min
      << ", max=" << max << endl;
  delete[]a;
  delete[]b;
  system("pause");
  return 0;
}
```



Блок-схема головної функції

Результати роботи:

```
Input n=9
Input 9 real numbers
90
-123
0
3
216
88
4
56
99
Dynamic array:
90  88  99
Average=92.3333, min=88, max=99
```

Питання для самоконтролю

1) Що таке вказівник? Навести синтаксис оголошення вказівника?
2) Яке тлумачення має значення вказівника NULL?
3) В який спосіб можна отримати адресу змінної? Яке призначення операції &?
4) Пояснити відмінності між змінними a та b, оголошення яких має вигляд:
   а) `int a; double b;` б) `int *a; double *b;`
5) Пояснити відмінності динамічного масиву від звичайного.
6) Вибрати правильні оголошення динамічного масиву з 5-ти цілих чисел:
   a) `int a[5];`
   b) `int *a[5];`
   c) `int *a=malloc(5);`
   d) `int *a=(int*) malloc(20);`
   e) `int *a=(int*) malloc(5*sizeof(int));`
   f) `int *a=new int [5];`
   g) `int *a=new [5];`
   h) `int *a=new int (5);`
7) В який спосіб можна звільнити пам'ять, виділену за допомогою оператора new? Яка функція застосовується для звільнення пам'яті, виділеної функцією calloc?

8) Вибрати правильні команди звільнення пам’яті від динамічного масиву a з 5-ти елементів:

a) `delete a[5];`
b) `delete a[];`
c) `delete []a;`
d) `free (a);`
e) `free a[5];`

9) У чому полягає відмінність роботи функцій `malloc()` та `calloc()`?

10) Яким є призначення функції `realloc()`?

Лабораторне завдання

1) У протоколі лабораторної роботи дати відповіді на контрольні запитання.

2) У протоколі лабораторної роботи скласти схему алгоритму і написати програму мовою C++ для розв’язання індивідуальних завдань з опрацювання одновимірних масивів з виділенням динамічної пам’яті (завдання вибрати з табл. 11.1).

3) Створити на комп’ютері програмні проєкти мовою C++ для реалізації написаних програм. Занести результати обчислень до протоколу.

Таблиця 11.1

№ вар.	Індивідуальне завдання
1	Ввести деяку послідовність цілих чисел і створити динамічний масив лише з тих чисел, значення яких менше 6-ти. За допомогою функції обчислити середнє арифметичне елементів масиву
2	Ввести деяку послідовність дійсних чисел і створити динамічний масив лише з додатних чисел. За допомогою функції визначити мінімальний елемент масиву та його індекс
3	Ввести деяку послідовність дійсних чисел і створити динамічний масив з чисел, розміщених до першого від’ємного числа (якщо від’ємних чисел немає, вибрати всі). За допомогою функції розмістити елементи масиву у порядку зростання значень
4	Ввести деяку послідовність цілих чисел і створити динамічний масив лише з парних ненульових чисел. За допомогою функції визначити мінімальний та максимальний елементи масиву
5	Ввести деяку послідовність чисел і створити динамічний масив з чисел, розміщених до першого числа зі значенням нуль (якщо нулів немає, вибрати всі числа). За допомогою функції обчислити добуток елементів, значення яких за модулем менше 10-ти
6	Ввести деяку послідовність дійсних чисел і створити динамічний масив лише з чисел, модуль яких не перевищує 8. За допомогою функції обчислити середнє арифметичне мінімального і максимального елементів масиву
7	Ввести деяку послідовність цілих чисел і створити динамічний масив з чисел, розміщених до першого трицифрового числа (якщо трицифрових чисел немає, вибрати всі). За допомогою функції обчислити середнє арифметичне непарних елементів

Продовження табл. 11.1

№ вар.	Індивідуальне завдання
8	Ввести деяку послідовність цілих чисел і створити динамічний масив лише з двоцифрових чисел. За допомогою функції обчислити кількість та суму парних елементів масиву
9	Ввести деяку послідовність цілих чисел і створити динамічний масив з чисел, розміщених до першого двоцифрового числа. За допомогою функції обчислити добуток непарних елементів масиву
10	Ввести деяку послідовність цілих чисел і створити динамічний масив з чисел, розміщених після першого трицифрового числа (якщо трицифрових чисел немає, видати про це повідомлення). За допомогою функції обчислити середнє арифметичне елементів масиву, кратних 3
11	Ввести деяку послідовність дійсних чисел і створити динамічний масив з чисел, розміщених після першого числа зі значенням 0 (якщо нулів немає, видати про це повідомлення). За допомогою функції обчислити суму елементів, абсолютне значення яких не перевищує 10
12	Ввести деяку послідовність дійсних чисел і створити динамічний масив з чисел, розміщених після першого від'ємного числа (якщо від'ємних чисел немає, видати про це повідомлення). За допомогою функції поміняти місцями мінімальний і максимальний елементи масиву
13	Ввести деяку послідовність дійсних чисел і створити динамічний масив лише з чисел, значення яких належить проміжку [10, 25]. За допомогою функції визначити кількість елементів, значення яких більше за значення першого елемента масиву
14	Ввести деяку послідовність цілих чисел і створити динамічний масив лише з непарних чисел. За допомогою функції замінити всі від'ємні елементи нулями
15	Ввести деяку послідовність дійсних чисел і створити динамічний масив лише з чисел, які перевищують свій порядковий номер. За допомогою функції розмістити елементи масиву у зворотному порядку
16	Ввести деяку послідовність дійсних чисел і створити динамічний масив лише з чисел, модуль яких не попадає в проміжок (20, 40]. За допомогою функції обчислити кількість елементів, що перевищують середнє арифметичне всіх елементів
17	Ввести деяку послідовність цілих чисел і створити динамічний масив лише з ненульових чисел. За допомогою функції визначити найбільший з парних елементів
18	Ввести деяку послідовність дійсних чисел і створити динамічний масив лише з чисел, модуль яких попадає в проміжок [5, 50). За допомогою функції визначити мінімальний із додатних елементів
19	Ввести деяку послідовність цілих чисел і створити динамічний масив з чисел, розміщених до першого від'ємного двоцифрового числа (якщо від'ємних двоцифрових чисел немає вибрати всі). За допомогою функції замінити мінімальний і максимальний елементи значенням середнього арифметичного всіх елементів

Закінчення табл. 11.1

№ вар.	Індивідуальне завдання
20	Ввести деяку послідовність цілих чисел і створити динамічний масив лише з чисел, кратних 3. За допомогою функції обчислити середнє арифметичне елементів, значення яких більше значення останнього елемента масиву
21	Ввести деяку послідовність цілих чисел і створити динамічний масив лише з чисел, значення яких не перевищують 100. За допомогою функції обчислити суму тільки двоцифрових елементів
22	Ввести деяку послідовність цілих чисел і створити динамічний масив з чисел, розміщених після першого двоцифрового числа. За допомогою функції обчислити середнє арифметичне елементів, кратних 5
23	Ввести деяку послідовність цілих додатних чисел і створити динамічний масив з чисел, які мають серед своїх цифр хоча б одну цифру 2. За допомогою функції обчислити добуток одноцифрових елементів масиву
24	Ввести деяку послідовність цілих чисел і створити динамічний масив з чисел, які завершуються цифрою 5. За допомогою функції обчислити середнє арифметичне двоцифрових елементів
25	Ввести деяку послідовність цілих чисел і створити динамічний масив з чисел, розміщених після першого парного числа (якщо парних чисел немає, видати про це повідомлення). За допомогою функції поміняти місцями мінімальний елемент з першим
26	Ввести деяку послідовність цілих додатних чисел і створити динамічний масив з чисел, які мають серед своїх цифр хоча б одну цифру 1. За допомогою функції поміняти місцями першу половину масиву з другою
27	Ввести деяку послідовність дійсних чисел і створити динамічний масив з чисел, розміщених до першого числа, модуль якого більше 100. За допомогою функції визначити максимальний елемент та його індекс
28	Ввести деяку послідовність дійсних чисел і створити динамічний масив з чисел, які відрізняються від числа 5 не більше ніж на 10. За допомогою функції визначити максимальний з від'ємних елементів масиву та його номер
29	Ввести деяку послідовність цілих чисел і створити динамічний масив з чисел, які завершуються цифрою 1. За допомогою функції замінити елементи, кратні 3, на 0
30	Ввести деяку послідовність дійсних чисел і створити динамічний масив лише з чисел, значення яких належить проміжку (–20, 10]. За допомогою функції обчислити середнє арифметичне від'ємних елементів

Лабораторна робота № 12
Вказівники і динамічна пам'ять при опрацюванні двовимірних масивів

Мета роботи: набути практичних навиків програмного використання вказівників та динамічної пам’яті при опрацюванні двовимірних масивів.

Теоретичні відомості

1. Вказівники на багатовимірні масиви

Вказівники на багатовимірні масиви у мові C++ – це масиви масивів, тобто такі масиви, елементами яких є масиви. При оголошенні таких масивів у пам’яті комп’ютера створюється декілька різних об’єктів. Приміром, при виконанні оголошення двовимірного масиву `int arr[4][3]` у пам’яті виділяється ділянка для зберігання значення змінної `arr`, яка є вказівником на масив з чотирьох вказівників. Для цього масиву з чотирьох вказівників теж виділяється пам’ять. Кожний з цих чотирьох вказівників містить адресу масиву з трьох елементів типу `int`, і, отже, у пам’яті комп’ютера виділяється чотири ділянки для зберігання чотирьох масивів чисел типу `int`, кожна з яких складається з трьох елементів. Такий розподіл пам’яті показано на схемі:

```
   arr
    ↓
arr[0]      →      arr[0][0]    arr[0][1]    arr[0][2]
arr[1]      →      arr[1][0]    arr[1][1]    arr[1][2]
arr[2]      →      arr[2][0]    arr[2][1]    arr[2][2]
arr[3]      →      arr[3][0]    arr[3][1]    arr[3][2]
```

Отже, оголошення `arr[4][3]` породжує в програмі три різних об’єкти: вказівник з ідентифікатором `arr`, безіменний масив з чотирьох вказівників і безіменний масив з дванадцяти чисел типу `int`. Для доступу до безіменних масивів використовуються адресні вирази з вказівником `arr`. Доступ до елементів масиву вказівників здійснюється із зазначенням одного індексного виразу у формі `arr[2]` чи `*(arr+2)`. Для доступу до елементів двовимірного масиву чисел типу `int` має бути використано два індексні вирази у формі `arr[1][2]` або еквівалентних їй `*(*(arr+1)+2)` та `(*(arr+1))[2]`.

Нагадаємо, що елементи двовимірних масивів розміщуються у пам’яті підряд і між його рядками немає ніяких проміжків. Такий порядок дає можливість звертатися до будь-якого елемента багатовимірного масиву, використовуючи адресу його початкового елемента та лише один індексний вираз. Так, для матриці `arr` звертання `*(arr)` є посиланням на елемент `arr[0][0]`, звертання `*(arr+2)` – на елемент `arr[0][2]`, а звертання `*(arr+3)` – на елемент `arr[1][0]` тощо. Тобто, застосовуючи оператори циклу, можна організувати поелементне опрацювання елементів матриці, наприклад введення усієї матриці:

```
for(int i=0; i<4; i++)
for(int j=0; j<3; j++) cin >> *(arr+i*4+j);
```

Тут вираз `*(arr+i*4+j)` є аналогічний до `arr[i][j]`.

2. Динамічна пам'ять для матриць

Основний спосіб роботи з динамічними матрицями базується на використанні подвійних вказівників. Для того, щоб працювати з динамічними матрицями, доцільно представити матрицю як масив векторів (масив, що містить вказівники на рядки матриці). Перед початком роботи з такою матрицею треба спочатку виділити пам'ять під масив вказівників на рядки, а лише тоді виділяти пам'ять для зберігання самих даних. Наприклад, оголошення динамічної матриці 7×8 дійсних чисел може бути таким:

```
double **a = new double*[7];          // сім рядків матриці
for (int i=0; i<7; i++)
  a[i] = new double[8];               // по вісім стовпців
```

Після використання матриці виділена для неї пам'ять звільняється у зворотній послідовності:

```
for(int i=0; i<7; i++) delete [] a[i];
delete []a;
```

Тобто при виділенні пам’яті для цієї матриці спочатку оголошується вказівник другого порядку `double **a`, який посилається на масив вказівників `double*[7]` (де 7 – розмір масиву). Після цього в циклі для кожного рядка матриці виділяється по вісім елементів.

Звертання до елементів динамічної матриці здійснюється так само, як і до елементів статичної матриці – `a[i][j]`.

Наведемо приклад програми з динамічним виділенням пам’яті для матриці, розмірність якої (`m×n`) вводить користувач. Значення елементів цієї матриці заповнюються в програмі випадковими числами з діапазону від 1 до 100 і виводяться на екран.

Для генерації випадкових чисел у мові С існує спеціальна функція `rand()`, яка повертає псевдовипадкове ціле число у діапазоні від 0 до вказаного значення. Приміром, команда для надання цілій змінній псевдовипадкового значення від 0 до 99 буде такою:

```
int x=rand() % 100;
```

Тобто вираз `rand()%n` сформує псевдовипадкове ціле число з проміжку від 0 до `n-1`. Щоб отримати псевдовипадкове число не від 0 до `n`, а з іншого проміжку (приміром `[a; a+n-1]`), треба скористатись виразом `rand()%n+a`. Причому, `a` може бути як додатним, так і від’ємним. Наприклад, вираз `(rand() % 31)+20` генеруватиме числа від 20 до 50, а вираз `(rand() % 101)-50` – числа від –50 до 50.

Перед використанням цієї функції треба ініціалізувати генератор випадкових чисел функцією `srand()` для того, щоб кожного разу формувалися різні, а не одні й ті самі числа. Аргументом функції `srand()` можна задати функцію `time()`, викликану з нульовим вказівником типу `time_t`, яка повертає кількість секунд від 1 січня 1970 року і тим самим задає випадкове зерно для генерації

чисел. Функції rand() та srand() прописані в заголовному файлі <cstdlib>, а time() – у файлі <ctime>. Наприклад, команди

```
time_t t;
srand ((unsigned) time(&t));
for (int i=0; i<10; i++)
  printf ("%d\n", rand() % 100);
```

генеруватимуть послідовність з 10-ти випадкових цілих чисел від 0 до 99. Подібна конструкція для дійсних чисел може бути такою:

```
srand (time(NULL));       // можна і так:  srand(time(0));
for (int i=0; i<10; i++)
  cout<<setw(4)<<setprecision(2)<<(rand()%100)/double((rand()%10))<<endl;
```

Приклад програми з динамічним виділенням пам'яті для матриці випадкових чисел, розмірність якої (m×n) вводить користувач:

```
#include <iostream>
#include <cstdlib>
#include <ctime>
#include <iomanip>
using namespace std;
int main()
{
  setlocale(0,".1251");
  srand (time(NULL));              // генерація випадкових чисел
  int m, n;                        // змінні для розмірності матриці
  cout<<"Введіть розмірність матриці через пробіл:  ";  cin>>m>>n;
  // динамічне створення матриці дійсних чисел розміром m×n
  double **a = new double*[m];  // n рядків матриці
  for (int i=0; i<m; i++)
    a[i] = new double[n]; // по m стовпців
  // заповнення матриці випадковими дійсними числами від 1 до 10
  for (int i=0; i<m; i++)
   for (int j=0; j<n; j++)
     a[i][j] = (rand() % 100 + 1) / double((rand() % 10 + 1));
  // виведення матриці
  for (int i=0; i<m; i++)
  { for (int j=0; j<n; j++)
      cout << setw(4) << setprecision(2) << a[i][j] << "   ";
    cout << endl;
  }
  for (int i=0; i<m; i++) delete []a[i];  // звільнення динамічної пам'яті
  delete []a;
  system("pause>>void");
  return 0;
}
```

```
Введіть розмірність матриці через пробіл:   3  7
34      0       4.3     12.7    -7.1     0.3    -5      9      -1
2       -9.4    1.5     6       123      -68    2.1     11     5.8
12      4.78    30      -64     -0.45    5      -3.9    3      12.1
```

При виведенні цієї матриці було використано модифікатор `setw`, який дозволив задавати ширину (кількість символьних позицій) кожного виведеного числа. Так, модифікатор `setw(4)` для кожного елемента відвів максимум по чотири позиції, що дозволило вирівняти по стовпцях елементи різного розміру. Якщо елемент мав меншу, аніж чотири знаки ширину, перед ним було виведено пробіли. Використаний при виведенні маніпулятор `setprecision(2)` дозволив обмежити кількість знаків після десяткової крапки максимум двома знаками. Саме для цих засобів було долучено файл `iomanip`, а докладно вони були розглянуті в лабораторній роботі № 1.

Приклади програм

Приклад 1. Ввести цілочислову матрицю 6×4 і створити нову матрицю з тих рядків введеної матриці, які не містять нульових елементів.

Розв'язок. На відміну від введеної матриці, створювана матриця буде динамічною, оскільки кількість її рядків заздалегідь є невідома. Для обчислення кількості нульових елементів кожного рядка організуємо тимчасовий допоміжний масив `r`. Кількість рядків створюваної матриці визначатимемо як кількість ненульових елементів масиву `r` й у нову матрицю будемо копіювати лише ті рядки введеної, для яких `r=0`.

Програмний код:

```
#include <iostream>
using namespace std;
int main()
{
  system("chcp 1251 > null");
  system("color F0");
  int a[6][4], i, j, rowc=0, k=0;
  cout<<"Введіть матрицю з 6-ти рядків і 4-х стовпців:"<<endl;
  int r[6]={0};  // r – масив кількості нульових елементів рядків матриці
  for (i=0; i<6; i++)
   for (j=0; j<4; j++)
   { cin>>a[i][j];
     if(a[i][j]==0) r[i]++;
   }
  for(i=0; i<6; i++)  // Обчислення кількості рядків без нульових елементів
    if(r[i]==0) rowc++;  // r[i]=0  означає, що в і-му рядку немає нулів
  int **b=new int *[rowc];  // Виділення пам'яті для динамічної матриці b
  for(i=0; i<rowc; i++) b[i]=new int [4];
  for(i=0; i<6; i++)
   if(r[i]==0)            // Якщо в і-му рядку нулів немає, відбувається
    { for(j=0;j<4;j++) b[k][j]=a[i][j]; // копіювання рядка у нову матрицю
      k++;                // і збільшення індексу рядка нової матриці.
    }
```

```
  cout<<"\nМатриця, в рядках якої немає нульових елементів:"<<endl;
  for (i=0; i<rowc; i++)
  { for (j=0; j<4; j++) cout<<b[i][j]<<"\t";
    cout << endl;
  }
  // Звільнення пам'яті від динамічної матриці
  for (int i=0; i<rowc; i++)
    delete [] b[i];
  delete []b;
  system ("pause>>void");
  return 0;
}
```

Результати роботи:

```
Введіть матрицю з 6-ти рядків і 4-х стовпців:
23      7       90      0
-6      0       88      45
33      78      9       4
12      4       8       90
0       3       0       -7
50      7       9       7

Матриця, в рядках якої немає нульових елементів:
33      78      9       4
12      4       8       90
50      7       9       7
```

Приклад 2. Ввести матрицю 3×7 дійсних чисел та створити нову матрицю з тих стовпців введеної матриці, які містять нульові елементи.

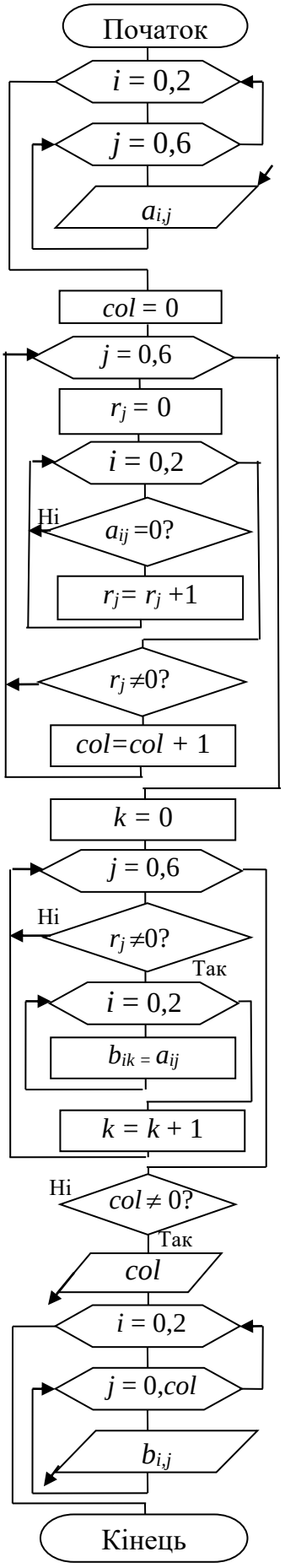
Програмний код:

```
#include <iostream>
using namespace std;
int main()
{
  system("chcp 1251 > null");
  system("color F0");
  double a[3][7]; int i, j;
  cout<<"Введіть матрицю з 3-х рядків і 7-ми стовпців:"<<endl;
  for (i=0; i<3; i++)
   for (j=0; j<7; j++)
     cin>>a[i][j];
  int col=0;      // col – кількість стовпців з нульовими елементами
  int r[7]={0};  // r – масив кількості нульових елементів стовпців матриці
  for (j=0; j<7; j++)
  {
    for (i=0; i<3; i++)         // Обчислення
      if (a[i][j]==0) r[j]++; // кількості нулів у стовпці та
    if (r[j]!=0) col++;       // кількості стовпців з нульовими елементами
  }
```

```
  // Виділення пам’яті для динамічної матриці
  double **b=new double*[3];
  for(i=0; i<3; i++)
    b[i]=new double [col];
 //індекс стовпця нової матриці (з нульовими елементами)
  int k=0;
  for(j=0; j<7; j++)
   if(r[j]!=0) //Якщо в j-му стовпці є нулі, скопіювати
    { for(i=0;i<3;i++)
        b[i][k]=a[i][j]; // стовпець у нову матрицю
      k++; // і збільшити індекс стовпця нової матриці
    }
  if(!col)
   cout << "\nНемає стовпців з нульовими елементами";
  else
   cout << endl << col
        << " стовпці(в) з нульовими елементами";
  cout << "\nМатриця, у стовпцях якої є нульові
          елементи:" << endl;
  for (i=0; i<3; i++)
  { for (j=0; j<col; j++)
      cout<<b[i][j]<<"\t";
    cout << endl;
  }
  // Звільнення пам’яті від динамічної матриці
  for(int i=0; i<3; i++)
    delete [] b[i];
  delete []b;
  system ("pause>>void");
  return 0;
}
```

Результати роботи:

```
Введіть матрицю з 3-х рядків і 7-ми стовпців:
45      7.8     0       -7      2.8     30      -5
0       5.8     5.7     0       12      -7      0
0.1     -5      23      6       0       4.8     41

5 стовпці(в) з нульовими елементами
Матриця, у стопвцях якої є нульові елементи:
45      0       -7      2.8     -5
0       5.7     0       12      0
0.1     23      6       0       41
```



Приклад 3. Обчислити визначник (детермінант) квадратної матриці розмірністю $n \times n$, значення n ввести.

Розв'язок. Обчислення визначника матриці є поширеною задачею вищої математики. Звичайно без значення визначника матриці не обійтися при розв'язанні складних систем рівнянь. За допомогою детермінанта визначають наявність та єдиність існування розв'язку систем рівнянь. Важко переоцінити важливість уміння правильно і точно знаходити визначник матриці в математиці. Методи обчислення визначників є теоретично простими, проте зі збільшенням розміру матриці обчислення стають дуже громіздкими і вимагають величезної уваги і багато часу.

Для матриці першого порядку значення визначника дорівнює єдиному елементу цієї матриці: $\Delta = |a_{11}| = a_{11}$.

Для матриці другого порядку (2×2) значення визначника обчислюється як:

$$\Delta = \begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} = a_{11} \cdot a_{22} - a_{12} \cdot a_{21}.$$

Для матриць більш високих порядків (вище другого) $n \times n$ визначник можна обчислити, застосувавши **рекурсивну формулу**, яку ще називають **розкладанням по рядку**:

$$\Delta = \begin{vmatrix} a_{11} & \dots & a_{1n} \\ \dots & \dots & \dots \\ a_{n1} & \dots & a_{nn} \end{vmatrix} = \sum_{j=1}^{n} (-1)^{1+j} \cdot a_{1j} \cdot M_j^1,$$

де M_j^1 – додатковий мінор до елемента a_{1j} – визначник матриці, отриманий з вихідної викресленням 1-го рядка та j-го стовпця.

Наприклад, формула обчислення визначника матриці 3-го порядку має вигляд:

$$\Delta = \begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix} = a_{11} \cdot \begin{vmatrix} a_{22} & a_{23} \\ a_{32} & a_{33} \end{vmatrix} - a_{21} \cdot \begin{vmatrix} a_{12} & a_{13} \\ a_{32} & a_{33} \end{vmatrix} + a_{31} \cdot \begin{vmatrix} a_{12} & a_{13} \\ a_{22} & a_{23} \end{vmatrix}$$

Тобто, визначник порядку n розкладається на добутки визначників порядку $(n-1)$ і коефіцієнтів, якими є елементи рядка (стовпця), по якому розкладалися мінори. Обчислені визначники $(n-1)$-го порядку обчислюються рекурсивно.

Один з найшвидших методів полягає у простій модифікації методу Гауса. Дотримуючись **методу Гауса**, довільну матрицю можна привести до ступінчастого вигляду (верхньотрикутної матриці), використовуючи лише дві операції над матрицею – перестановку двох рядків і додавання до одного з рядків матриці іншого рядка, помноженого на довільне число. З властивостей визначника випливає, що друга операція не змінює визначника матриці, а перша лише змінює його знак на протилежний. Визначник матриці, приведеної до ступінчастого вигляду, дорівнює добутку елементів її діагоналі, оскільки вона є трикутною. Метод Гауса обчислення визначника добре працює для матриць порядку вище 3 на 3.

Далі у програмному коді наведемо обидва методи обчислення визначника у вигляді окремих функцій: ітеративний метод Гауса і рекурсивний метод розкладання по рядку.

Програмний код:

```
#include <math.h>
#include <iostream>
using namespace std;

double detGauss(double **a, int n)
{ int i,j,t,k; double d,p=0;
  for (i=0; i<n-1; i++)
  { t=1;
    while (a[i][i]==0)
    { for (j=0; j<n; j++)
      { d = a[i][j];
        a[i][j] = a[i+t][j];
        a[i+t][j] = d;
      }
     p++; t++;
   }
   for (k=i+1; k<n; k++)
   { d = a[k][i]/a[i][i];
     for (j=0; j<n; j++) a[k][j] -= a[i][j]*d;
 } }
 d = pow(-1,p);
 for (i=0; i<n; i++)  d*=a[i][i];
 return d;
}

double detR(double** a, int n)
{ int i,j;  double d=0, **matr;
  if (n==1)  d=a[0][0];
  else
    if (n==2)  d=a[0][0]*a[1][1]-a[0][1]*a[1][0];
    else
    { matr = new double*[n-1];
      for (i=0; i<n; ++i)
      { for (j=0; j<n-1; ++j)
        { if (j<i) matr[j]=a[j];
          else     matr[j]=a[j+1];
        }
        d += pow(-1.0, (i+j)) * detR(matr,n-1) * a[i][n-1];
      }
      delete[] matr;
    }
  return d;
}
```

```
int main()
{
 system("chcp 1251 > null");
 system("color F0");
 int n,i,j;
 cout << "Введіть ціле число - розмір квадратної матриці -> ";
 cin >> n; // Розмір квадратної матриці
 if (n <= 0)
 { cout << "Некоректне значення! Введіть новий розмір квадратної матриці";
   system ("pause>>void"); return 0;
 }
 else
 {
  // Виділення пам'яті для динамічної матриці a
  double **a=new double *[n];
  for (i=0; i<n; i++) a[i]=new double [n];
  cout<<"Введіть матрицю "<<n<<"x"<<n<<":"<<endl;
  for (i = 0; i < n; i++)
  for (j = 0; j < n; j++)    cin>>a[i][j];
  double d1 = detGauss(a, n);
  double d2 = detR(a, n);
  cout << "Визначник за методом Гауса - "<< d1<<endl;
  cout << "Визначник за рекурсивним методом - "<< d2<<endl;
  // Звільнення пам'яті від динамічної матриці
  for(int i=0; i<n; i++)
    delete [] a[i];
  delete []a;
 }
 system ("pause>>void");
 return 0;
}
```

```
Введіть ціле число - розмір квадратної матриці -> 3
Введіть матрицю 3x3:
9       9       3
9       -5      -2
-10     1       0
Визначник за методом Гауса - 75
Визначник за рекурсивним методом - 75
```

```
Введіть ціле число - розмір квадратної матриці -> 6
Введіть матрицю 6x6:
3.2     0       -7      1       -0.2    3
0.2     -3      10      0.4     2.6     0
-7      1       -1      0       -0.3    5
0       -0.25   10      0       1       -7
1       3.9     1.7     -4.9    8       -1
-3      1       -2      11.8    -8      0
Визначник за методом Гауса - 13828.7
Визначник за рекурсивним методом - 13828.7
```

Питання для самоконтролю

1) Вибрати правильне оголошення динамічної матриці 5×4 цілих чисел:

a) `int a[5][4];`

b) `int *a[5][4];`

c)
```
int *a=new int [5][4];
  for(int i=0; i<5; i++) a[i]=new int [4];
```

d) `int **a=new int*[5];`

e)
```
int **a=new int *[4];
  for(int i=0; i<4; i++) a[i]=new int [5];
```

2) Вибрати правильний варіант звільнення пам’яті від динамічної матриці `a` з 5×4 цілих елементів:

a) `delete a[5][5];`

b) `delete a[][];`

c) `delete [][]a;`

d)
```
for(int i=0; i<5; i++) delete [] a[i];
  delete []a;
```

e)
```
for(int i=0; i<4; i++) delete [] a[i];
  delete []a;
```

Лабораторне завдання

1) У протоколі лабораторної роботи дати відповіді на контрольні запитання.

2) У протоколі лабораторної роботи скласти схему алгоритму і написати програму мовою С++ для розв’язання індивідуальних завдань (табл. 12.1) з опрацювання двовимірних масивів з виділенням динамічної пам’яті.

3) Створити на комп’ютері програмні проєкти мовою С++ для реалізації написаних програм. Занести результати обчислень до протоколу.

Таблиця 12.1

№ вар.	Індивідуальне завдання
1	Ввести матрицю 10×3 дійсних чисел і створити нову матрицю з тих рядків введеної матриці, які не містять від’ємних елементів
2	Ввести цілочислову матрицю 3×9 і створити нову матрицю з тих стовпців введеної матриці, які містять парні елементи
3	Ввести матрицю 9×3 дійсних чисел і створити нову матрицю з тих рядків введеної матриці, які містять елементи лише в діапазоні [50, 100]
4	Ввести цілочислову матрицю 3×8 і створити нову матрицю з тих стовпців введеної матриці, які містять елементи, кратні 3
5	Ввести цілочислову матрицю 10×3 і створити нову матрицю з тих рядків введеної матриці, які містять хоча б один елемент – двійку
6	Ввести матрицю 3×9 дійсних чисел і створити нову матрицю з тих стовпців введеної матриці, які не мають елементи менші 10-ти
7	Ввести цілочислову матрицю 8×3 і створити нову матрицю з тих рядків введеної матриці, які містять хоча б один елемент, кратний 100
8	Ввести цілочислову матрицю 5×9 і створити нову матрицю з тих стовпців введеної матриці, які не містять жодного елемента 10
9	Ввести цілочислову матрицю 9×5 і створити нову матрицю з тих рядків введеної матриці, які не містять парні елементи

Закінчення табл. 12.1

№ вар.	Індивідуальне завдання
10	Ввести матрицю 3×6 дійсних чисел і створити нову матрицю з тих стовпців введеної матриці, які містять від'ємні елементи
11	Ввести цілочислову матрицю 10×4 і створити нову матрицю з тих рядків введеної матриці, які містять елементи, кратні 10
12	Ввести матрицю 4×9 дійсних чисел і створити нову матрицю з тих стовпців введеної матриці, які мають елементи більші за 100
13	Ввести цілочислову матрицю 8×4 і створити нову матрицю з тих рядків введеної матриці, які не містять жодного елемента 5 чи 25
14	Ввести матрицю 3×10 дійсних чисел і створити нову матрицю з тих стовпців введеної матриці, які мають елементи за модулем менші 7-ми
15	Ввести цілочислову матрицю 7×3 і створити нову матрицю з тих рядків введеної матриці, які містять хоча б один елемент – 7 або 13
16	Ввести цілочислову матрицю 4×10 і створити нову матрицю з тих стовпців введеної матриці, які містять хоча б один елемент – одиницю
17	Ввести матрицю 8×3 дійсних чисел і створити нову матрицю з тих рядків введеної матриці, які не містять елементів більших за 100
18	Ввести цілочислову матрицю 4×9 і створити нову матрицю з тих стовпців введеної матриці, які не містять елементи, кратні 10
19	Ввести матрицю 9×5 дійсних чисел і створити нову матрицю з тих рядків введеної матриці, які мають елементи менші 5-ти
20	Ввести цілочислову матрицю 4×9 і створити нову матрицю з тих стовпців введеної матриці, які містять хоча б один елемент 2 або –1
21	Ввести цілочислову матрицю 7×3 і створити нову матрицю з тих рядків введеної матриці, які не містять жодного елемента 7
22	Ввести цілочислову матрицю 4×8 і створити нову матрицю з тих стовпців введеної матриці, які містять непарні елементи
23	Ввести матрицю 10×3 дійсних чисел і створити нову матрицю з тих рядків введеної матриці, які не мають елементи за модулем менші за 9
24	Ввести матрицю 3×8 дійсних чисел і створити нову матрицю з тих стовпців введеної матриці, які не містять елементи з діапазону (10, 20]
25	Ввести цілочислову матрицю 8×3 і створити нову матрицю з тих рядків введеної матриці, які не містять елементи, кратні 3
26	Ввести матрицю 5×9 дійсних чисел і створити нову матрицю з тих стовпців введеної матриці, які містять елементи, дійсна частина яких дорівнює 0
27	Ввести цілочислову матрицю 7×4 і створити нову матрицю з тих рядків введеної матриці, які не містять непарні елементи
28	Ввести цілочислову матрицю 3×9 і створити нову матрицю з тих стовпців введеної матриці, які містять хоча б один елемент, кратний 20
29	Ввести матрицю 7×5 дійсних чисел і створити нову матрицю з тих рядків введеної матриці, які не містять елементи, дійсна частина яких дорівнює 0
30	Ввести цілочислову матрицю 5×9 і створити нову матрицю з тих стовпців введеної матриці, які не містять жодного елемента 11 або 22

Самостійна робота № 11

Створення бібліотеки функцій

Мета роботи: набути практичних навиків зі створення статичної бібліотеки функцій мовою C++ та долучення її заголовного файлу до програмних проєктів.

Теоретичні відомості

Статичні та динамічні бібліотеки функцій

Бібліотеками називають набори функцій (підпрограм) і/або об'єктів, організовані у вигляді окремих модулів (файлів), звичайно орієнтованих на розв’язання близьких за тематикою завдань. Бібліотеки є хорошим способом повторного використання коду. Замість того щоб щоразу реалізовувати одні і ті самі функції для забезпечення тієї чи іншої функціональності в кожному створюваному проєкті, їх можна створити один раз і потім викликати з проєктів. Для використання бібліотеки необхідно вказати компілятору, що її потрібно долучити і викликати функцію з бібліотеки (скориставшись відповідним заголовним файлом), при цьому вихідний текст функції не потрібен. З точки зору організації та використання бібліотеки бувають статичними і динамічними.

Статичні бібліотеки (static library) – це набір функцій, але не у формі коду, а у вже скомпільованому вигляді. Файли статичних бібліотек C++ мають розширення `lib`. У результаті зв'язування зі статичною бібліотекою, програма долучає всі використовувані функції, що збільшує її розмір, але робить більш автономною.

Динамічні бібліотеки (dynamic link library – dll) завантажуються операційною системою "на вимогу" запущеної програми вже в ході її виконання. Якщо необхідна бібліотека вже була завантажена в пам'ять, то повторне завантаження не виконується. При цьому один і той самий набір функцій чи об'єктів бібліотеки може бути використаний одночасно кількома працюючими програмами, що дозволяє ефективно використовувати ресурси оперативної пам'яті. Динамічні бібліотеки здебільшого мають розширення `dll`.

Заголовні файли

Заголовні файли мають розширення `h` і містять заголовки (прототипи) функцій, а також оголошення типів, констант і змінних з ключовим словом `extern`. Перевага заголовних файлів полягає у тому, що, коли створено заголовний файл для одного модуля (одного або декількох `cpp`-файлів) програми, можна переносити всі зроблені оголошення до іншої програми C++. Для цього слід просто залучити заголовний файл за допомогою директиви `#include`. Зазвичай для кожного `cpp`-файлу створюється власний заголовний файл з таким самим ім’ям і розширенням `h`. І навпаки, здебільшого для кожного заголовного файлу створюється `cpp`-файл з реалізацією функцій, оголошених у `h`-файлі.

Заголовні файли можна розділити на стандартні й створювані програмістом. Заголовні файли, створені програмістом, звичайно розміщують у теці проєкту. Імена цих файлів у директиві `#include` пишуться у подвійних лапках і завжди з розширенням `h`, наприклад: `#include "MyBibl.h"`.

Директиви препроцесора

Директиви препроцесора є інструкціями, записаними в тексті С-програми, які виконуються до трансляції програми. Директиви препроцесора дозволяють вставляти програмний код з іншого файлу, забороняти трансляцію частини тексту тощо. Всі директиви препроцесора розпочинаються зі знаку `#`. Після директив препроцесора крапка з комою не ставиться.

Найбільш поширені препроцесорні директиви:

`#include` – долучення до програми вмісту зазначеного файлу;

`#define` – визначення макросу або препроцесорного ідентифікатора;

`#pragma` – дії, передбачені реалізацією;

`#ifdef` – перевірка визначеності ідентифікатора;

`#ifndef` – перевірка невизначеності ідентифікатора;

`#error` – формування тексту повідомлення про помилку при трансляції;

Наприклад, щоб задати константу `n` зі значенням 10 слід написати директиву

```
#define n 10
```

Ця директива є аналогом такої інструкції `const int n = 10;`

Відмінності полягають у тому, що:

1) при використанні константи пам’ять під константну змінну виділяється, а при використанні директиви – ні;

2) при використанні директиви не виконується перевірка типу значення, що може спричинити помилки у програмі;

3) тип значення `n` при використанні директиви визначається автоматично, тобто, якщо у програмі є суттєвим, щоб `n` мала тип `unsigned short`, зробити це за допомогою директиви неможливо.

Отже, використання директиви `#define` у C++ без особливої на те потреби є небажаним.

Директива `#include` долучає до тексту програми вміст зазначеного файлу. Ця директива широко використовується для долучення до програм заголовних файлів бібліотек. При чому імена заголовних файлів стандартних бібліотек зазвичай записують у кутових дужках, а власних бібліотек – у лапках, наприклад:

```
#include "MyBibl.h"
#include <math.h>
```

Дія директиви `#pragma` залежить від конкретної реалізації компілятора. Ця директива дозволяє видавати компілятору різні інструкції.

Директива `#if` та її модифікації `#ifdef`, `#ifndef` разом з директивами `#else`, `#endif`, `#elif` дозволяють організувати умовне опрацювання тексту програми. При використанні цих засобів компілюється не весь текст, а лише ті його частини, які вибираються за допомогою вищенаведених директив. Головна ідея полягає у тому, що, якщо вираз, який розміщено після директив `#if`, `#ifdef`, `#ifndef` виявиться істинним, то буде скомпільовано код, розміщений між одні-

єю з цих трьох директив та директивою `#endif`, інакше цей код буде опущений. Директива `#endif` використовується для позначення закінчення блока `#if`. Директиву `#else` можна використовувати з кожною із наведених вище директив для надання альтернативного варіанта компіляції.

Директива `#error` дозволяє задавати текст діагностичного повідомлення, яке виводиться при виявленні помилок. За наявності цієї директиви відображується задане повідомлення і номер рядка.

Приклади створення бібліотеки функцій і використання цих функцій в основній програмі

Розглянемо на прикладах процеси створення статичної і динамічної бібліотек у Visual Studio 2017.

Приклад 1. Створити статичну бібліотеку з функціями, які реалізовуватимуть вилучення та вставлення певної частини рядка. Також створити консольний проєкт, який використовуватиме функції бібліотеки для введеного рядку.

Розв'язок. У бібліотеці буде дві функції, кожна з яких матиме по три аргументи. Функція вилучення певної частини рядка `DelChars()` матиме такі аргументи:

- рядок, з якого треба вилучити частину символів;
- позиція (індекс) символу, з якого треба починати вилучення;
- кількість символів, що мають бути вилучені.

Аргументи функції вставлення `Insert()` будуть такі:

- рядок, в який треба вставити підрядок (слово);
- підрядок (слово), яке вставляється;
- позиція, на яку вставляється підрядок (слово).

Алгоритм створення статичної бібліотеки складатиметься з таких етапів:

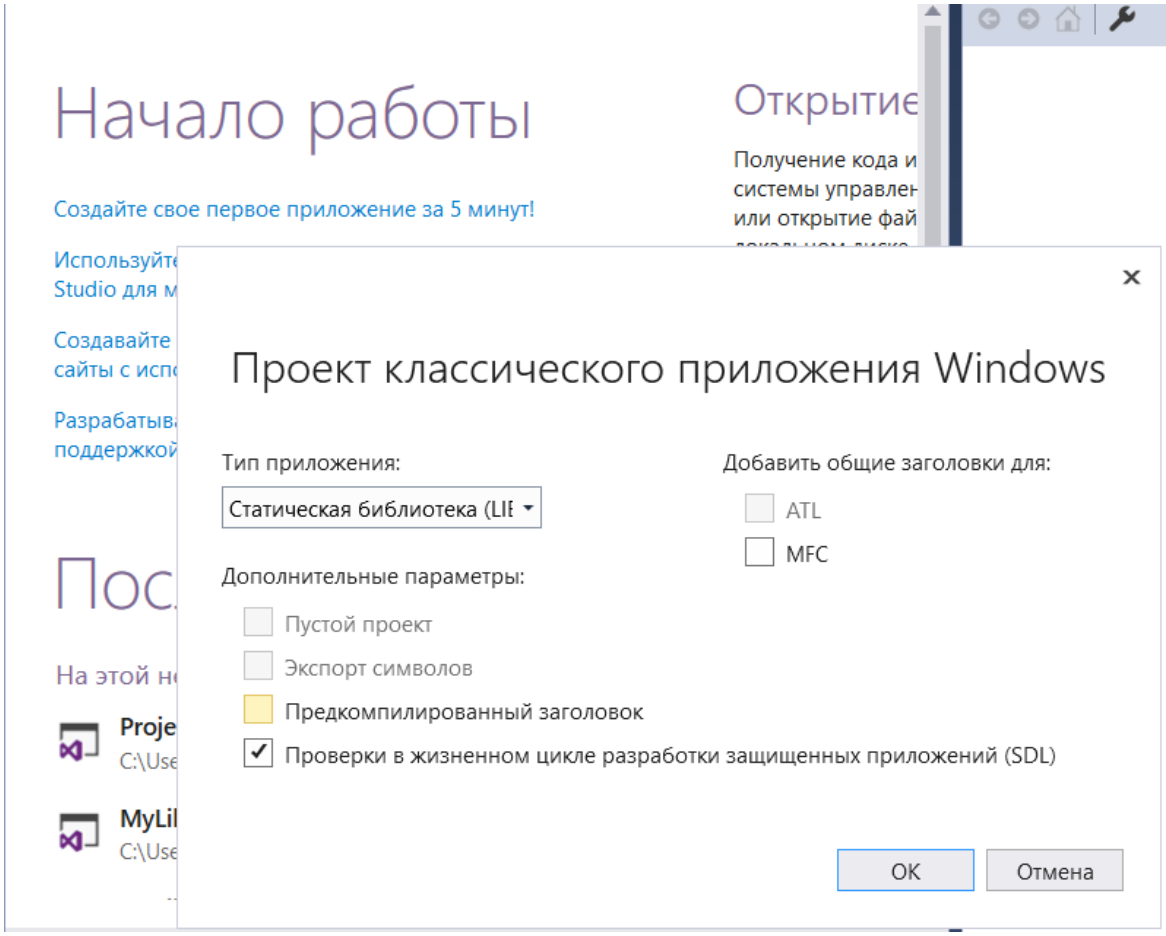
1. створення статичної бібліотеки у Visual Studio;
2. додавання класу до статичної бібліотеки;
3. створення та налаштування консольного проєкту в Visual Studio, який використовуватиме створену бібліотеку.

Розглянемо докладно кожен з цих етапів.

1. Створення статичної бібліотеки у Visual Studio

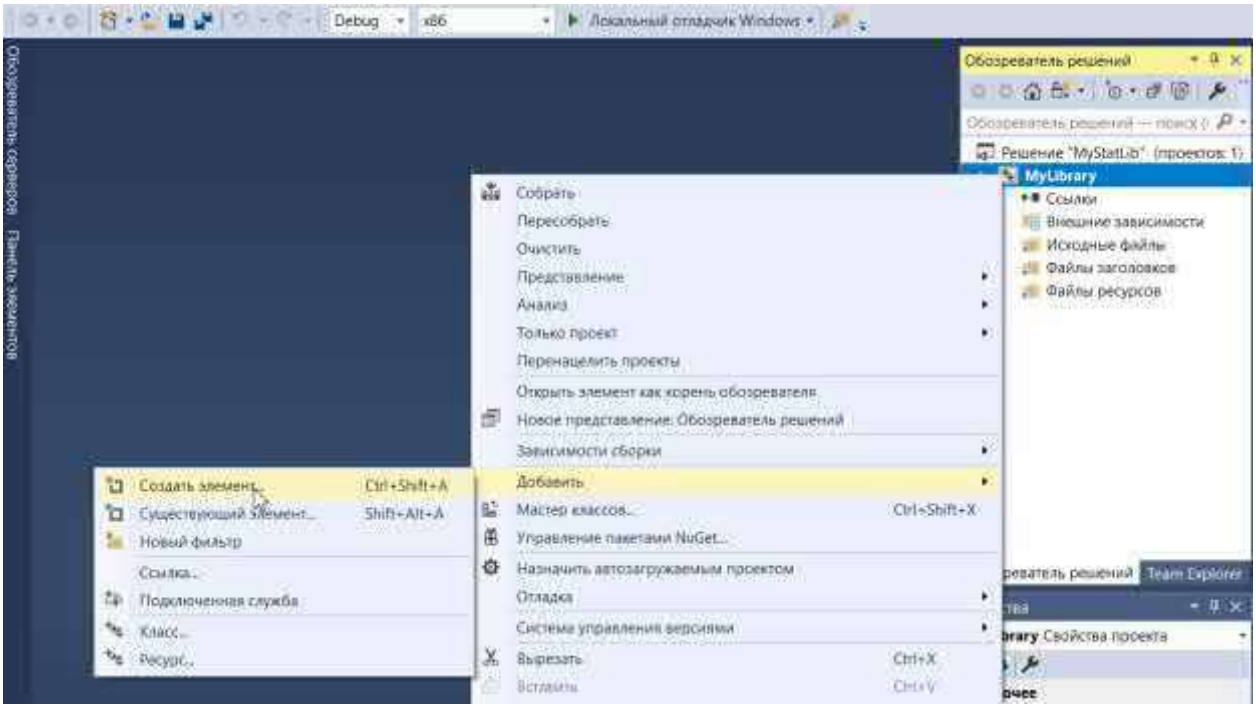
1.1. Спочатку треба при створенні нового проєкту вибрати *Мастер создания классических приложений Windows* (*Windows Desktop Wizard*) і ввести назву проєкту, наприклад **`MyLibrary`**, і назву рішення, наприклад **`MyStatLib`**.

1.2. У вікні майстра вибрати тип проєкту *Статическая библиотека (.lib)*, вимкнути прапорець *Предкомпилированный заголовок* і натиснути *ОК*.



2. Додавання класу в статичну бібліотеку

2.1. Для створення файлу заголовка для нового класу треба в оглядачі розв'язків відкрити контекстне меню проєкту `MyLibrary`, а потім вибрати команду *Добавить – Создать элемент*.



2.2. У діалоговому вікні додавання нового елемента ліворуч у розділі *Visual C++* треба вибрати *Код*. У центральній області вибрати *Заголовочный файл (.h)*, вказати ім'я для файлу заголовка, наприклад **MyLibrary.h**, і натиснути кнопку *Добавить*. З'явиться порожній файл заголовка.

2.3. Створити простір імен **StringFuncs** і клас **MyStrFuncs**:

```
namespace StringFuncs
{
  class MyStrFuncs
  {
  public:
       static void  DelChars(char* s, int p, int l);
       static char* Insert(char* str, const char* spos, int pos);

  };
}
```

Модифікатор `static` використовується тут для оголошення статичного члена класу, що належить власне типу, а не конкретному об'єкту. Доступ до цих методів буде здійснюватися із використанням імені класу та оператора доступу (::).

2.4. При створенні вихідного файлу для нового класу треба в оглядачі розв'язків відкрити контекстне меню для проєкту `MyLibrary` і вибрати *Добавить – Создать элемент*. У діалоговому вікні додавання нового елемента у лівій області розділу Visual C++ вибрати *Код*, після чого у центральній області вибрати *Файл ++ (.cpp)*. Вказати ім'я для файлу заголовка, наприклад `MyLibrary.cpp`, і натиснути кнопку *Додавить*. Після цього з'явиться вікно порожнього вихідного файлу.

2.5. У цьому файлі треба ввести код реалізації функцій `MyStrFuncs`:

```
#include "MyLibrary.h"
using namespace std;

namespace StringFuncs
{
     void MyStrFuncs::DelChars(char* s, int p, int l)
     {
          char* dst;
          for (dst = s + p, s = dst + l; *dst++ = *s++; );
     }
     char*  MyStrFuncs::Insert(char* str, const char* spos, int pos)
     {
          char* tmp = str;
          const char*  ps;
          int  len1, len2, cnt, i;
          for (len1 = 0, ps = spos; *ps; *ps++, len1++);
```

```
        for (len2 = 0, ps = str; *ps; *ps++, len2++);
        cnt = len1;
        while (cnt--)
        {    for (i = len2 + len1; i >= pos; i--)
                  *((str)+i + 1) = *((str)+i);
        }
        str += pos;
        while (*spos)
             *str++ = *spos++;
        return tmp;
    }
}
```

2.6. Скомпілювати бібліотеку командою *Сборка – Собрать решение*.

3. Створення та налаштування консольного проєкту в Visual Studio, який використовуватиме створену бібліотеку

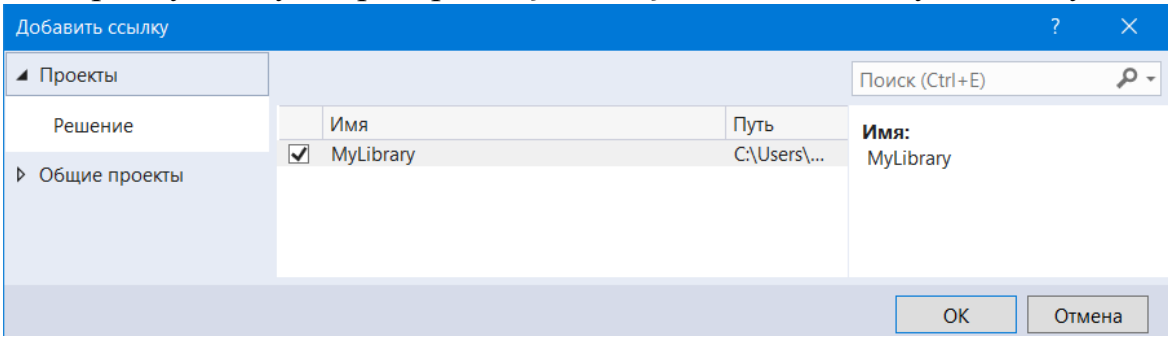
3.1. При створенні нового проєкту вибрати *Мастер классических приложений Windows*.

3.2. Вказати ім'я проєкту, наприклад `Project1`, у полі *Имя*. Зі списку поруч з полем *Решение* вибрати *Добавить в решение*. В результаті новий проєкт буде додано до рішення, що містить статичну бібліотеку. Натиснути кнопку *ОК*.

3.3. У вікні майстра вибрати тип проєкту *Консольное приложение*. Вимкнути прапорець *Предкомпилированный заголовок* і натиснути *ОК*.

3.4. Для використання функцій зі статичної бібліотеки у проєкті треба послатися на цю бібліотеку. Для цього треба в оглядачі розв'язків для проєкту `Project1` натиснути правою кнопкою миші і вибрати *Ссылки – Добавить ссылку*.

3.5. У діалоговому вікні додавання посилання перераховані бібліотеки, на які можна сформувати посилання. На вкладці *Проєкты* буде наведено усі проєкти поточного рішення і долучені до них бібліотеки, якщо вони є. Треба увімкнути прапорець `MyLibrary`, а потім натиснути кнопку *ОК*.



3.6. Для створення посилання на файл заголовка `MyLibrary.h` треба змінити шлях до папки з цим файлом. Для цього у діалоговому вікні властивостей проєкту `Project1` (*Проєкт – Свойства: Project1*) треба розгорнути вузли *Свойства конфигурации – C / C++ – Общие*. У полі *До-*

полнительные каталоги включаемых файлов треба вказати шлях до папки `MyLibrary` або відшукати цю папку. Для цього треба відкрити список значень властивості і вибрати *Изменить*. У текстовому полі діалогового вікна *Дополнительные каталоги включаемых файлов* клацнути на порожній рядок і натиснути кнопку з трьома крапками (...) наприкінці рядка. У діалоговому вікні *Вибор каталога* вибрати папку `MyLibrary` і натиснути кнопку *Вибор папки*. Натиснути двічі *ОК*.

3.7. Тепер клас `MyStrFuncs` можна використовувати в проєкті. Для цього треба замінити вміст файлу `Project1.cpp` на такий код:

```
#include "MyLibrary.h"
#include <iostream>

using namespace std;
int main()
{     char s[100], str[100]; int pos;
      setlocale(LC_ALL, ".1251");   puts("Введіть рядок: ");
      setlocale(LC_ALL, ".OCP");    gets_s(s, 100);
      StringFuncs::MyStrFuncs::DelChars(s, 4, 7);
      puts(s);
      setlocale(LC_ALL, ".1251");   puts("Введіть рядок для вставлення: ");
      setlocale(LC_ALL, ".OCP");    gets_s(str, 100);
      setlocale(LC_ALL, ".1251");
      puts("Введіть позицію для вставлення (ціле число): ");
      cin >> pos;
      setlocale(LC_ALL, ".OCP");
      puts(StringFuncs::MyStrFuncs::Insert(s, str, pos));
      system("pause");
      return 0;
}
```

Звертаємо увагу на те, що для виклику створених функцій `DelChars` і `Insert` треба спочатку вказати назву простору імен (`StringFuncs`), а потім назву класу (`MyStrFuncs`) та використати оператор доступу (::).

3.8. Щоб призначити консольний проєкт `Project1` таким, що завантажується першим, треба натиснути на проєкті правою кнопкою миші і вибрати *Назначить автозагружаемым проєктом*.

3.9. Скомпілювати проєкт командою *Сборка – Собрать решение* і запустити на виконання.

Результати виконання програми:

```
Введіть рядок:
Один два три чотири
Один  чотири
Введіть рядок для вставлення:
шість
Введіть позицію для вставлення (ціле число):
5
Один шість чотири
```

Приклад 2. Створити динамічну бібліотеку з функціями, які реалізовуватимуть такі завдання з одно- та двовимірними масивами:

1) обчислення елементів матриці *a* розмірністю 8×8 за формулою

$$a_{i,j} = \begin{cases} \sqrt[3]{\left(\dfrac{i+j+2}{7i+1}+j\right)^4} - 3.7*(i+j), & \text{за} \quad \sin(i+j) > 0 \\ \pi \sin^3(i-j), & \text{за} \quad \sin(i+j) \le 0 \end{cases}$$

2) обчислення вектора (одновимірного масиву), елементи якого є сумами елементів відповідних рядків матриці *a*;

3) пошук елемента матриці *a*, найбільш наближеного до значення середнього арифметичного елементів матриці *a*.

Після створення DLL-бібліотеки створити консольний проєкт, який використовуватиме функції цієї бібліотеки.

Розв'язок. У бібліотеці буде три функції. Алгоритм створення DLL-бібліотеки з цими функціями у Visual Studio 2017 складатиметься з таких етапів:

1. створення проєкту бібліотеки DLL;
2. створення заголовного файлу (.h) з оголошенням констант і заголовків функцій;
3. реалізація функцій (оголошених у заголовному файлі) у .cpp файлі;
4. створення консольного проєкту, який використовуватиме створену DLL-бібліотеку.

Розглянемо докладно кожен з цих етапів.

1. Створення проєкту бібліотеки DLL у Visual Studio.

1.1. При створенні нового проєкту вибрати *Мастер создания классических приложений Windows* (*Windows Desktop Wizard*) і ввести назву, наприклад, **`MyLib`**.

1.2. У вікні майстра вибрати тип проєкту *Библиотека динамической компоновки (.dll)*.

1.3. У властивостях проєкту (меню *Проєкт – Свойства*) у розділі *C/C++ – Препроцессор* перевірити визначення препроцесора: *MYLIB_EXPORTS* має бути написано великими літерами. Якщо це не так (написано *MyLib_EXPORTS*), треба виконати такі дії:

- вгорі вікна властивостей вибрати конфігурацію *Все конфигурации*;
- у розділі *C/C++ – Препроцессор – Определения препроцессора* клацнути на стрілку праворуч і натиснути *Изменить*;
- у верхній панелі додати *MYLIB_EXPORTS*;
- двічі клацнути *ОК*.

2. Створення заголовного файлу (.h) з оголошенням констант та заголовків функцій.

2.1. Для створення заголовного файлу треба в меню виконати команду *Проєкт – Добавить новый элемент* і вибрати *Заголовочный файл (.h)*. Задати `MathLib.h` як ім'я файлу.

2.2. Ввести такий програмний код з оголошеннями у заголовний файл `MathLib.h`:

```
#pragma once
#define k 8   // Оголошення константи
// Оголошення типів matr (матриця) і vekt (вектор)
typedef double matr[k][k], vekt[k];
// Оголошення прототипів функцій
__declspec(dllexport) void elem_Matr(matr c);
__declspec(dllexport) void elem_Vect(matr c, vekt v);
__declspec(dllexport) double G(matr c);
```

Тут кожна з трьох функцій, які будуть експортуватися, оголошується з ключовим словом: `__declspec(dllexport).`

3. Реалізація функцій у .cpp файлі.

3.1. Перейти на вкладку `MyLib.cpp`, якщо вона вже відкрита. Якщо ні, то відкрити її в оглядачі розв'язків у папці *Исходные файлы*.

3.2. Ввести такий програмний код у файл реалізації `MyLib.cpp`:

```
...
#define _USE_MATH_DEFINES
#include <math.h>  // Долучення математичної бібліотеки
#include "MyLib.h"
// Визначення функції розрахунку елементів матриці
void elem_Matr(matr c)
{
      for (int i = 0; i < k; i++)
        for (int j = 0; j < k; j++)
          if (sin(i + j) > 0)
            c[i][j] = pow((i+j+2.0)/(7*i+1)+j, 4.0/3) - 3.7*(i+j);
          else
            c[i][j] = M_PI * pow(sin(i - j), 3);
}
// Визначення функції  розрахунку елементів вектора
void elem_Vect(matr c, vekt v)
{
      for (int i = 0; i < k; i++)
      {
         v[i] = 0;
         for (int j = 0; j < k; j++) v[i] += c[i][j];
      }
}
// Пошук найближчого елемента до середнього арифметичного
double G(matr c)
{
      int ni = 0, nj = 0, i, j;
      double sr = 0;
      for (i = 0; i < k; i++)
        for (j = 0; j < k; j++)
```

```
            sr += c[i][j] / pow(k, 2);
      for (i = 0; i < k; i++)
        for (j = 0; j < k; j++)
            if (fabs(sr - c[i][j]) < fabs(sr - c[ni][nj]))
            {
                  ni = i;     nj = j;
            }
      return c[ni][nj];
}
```

3.3. Скомпілювати створену динамічну бібліотеку командою *Сборка – Построить решение*. Якщо помилок не буде виявлено, це означатиме, що бібліотеку успішно створено. Інакше треба буде виправити помилки і знову виконати компіляцію.

4. Створення консольного проєкту, який використовуватиме створену DLL-бібліотеку.

4.1. Створити новий консольний проєкт.

4.2. Відкрити властивості проєкту, вибрати конфігурацію *Все конфигурации*, перейти у розділ *C/C++ – Общие*. Клацнути праворуч від пункту *Дополнительные каталоги включаемых файлов*, вибрати *Изменить* і вказати шлях до папки, в якій зберігається заголовний файл `MyLib.h` (…\MyLib\MyLib). Клацнути *ОК*.

4.3. Для використання функцій бібліотеки `MyLib.h` у створеному консольному проєкті треба ввести такий програмний код:

```
#include "pch.h"
#include <iostream>
#include <iomanip>
#include "MyLibrary.h"      // Долучення власної бібліотеки
using namespace std;

int main()
{
      matr a;   vekt x;
      elem_Matr(a); // Виклик функції обчислення елементів матриці
      cout << "Matrix:\n";
      cout << fixed;
      for (int i = 0; i < k; i++)
      {
            for (int j = 0; j < k; j++)
                  cout << setprecision(2) << a[i][j] << setw(8);
            cout << endl;
      }
      elem_Vect(a, x); // Виклик функції обчислення елементів вектора
      cout << "\nVector:\n";
      for (int i = 0;i < k;i++)
            cout << setprecision(2)<<x[i] << setw(8);
      cout << endl;
```

```
    // Виведення значення скаляра
    cout << "\nG=" << G(a)<<endl;
    system("pause");
    return 0;
}
```

Цей код можна скомпілювати, але він не може бути приєднаним, позаяк ще не вказано шлях до файлу `MyLib.lib`.

4.4. Відкрити вікно властивостей проєкту, вибрати розділ *Компоновщик – Ввод – Дополнительные зависимости*. Як і на попередньому кроці вибрати *Изменить* і ввести у верхній панелі назву `MyLib`.

4.5. У вікні властивостей проєкту перейти до розділу *Компоновщик – Общие – Дополнительные каталоги библиотек*. Як і на попередньому кроці вибрати *Изменить* і вказати у верхній панелі шлях до папки, яка містить файл `MyLib.lib` (…`\MyLib\Debug`).

Консольний застосунок відтепер успішно компілюється та з'єднується, однак все ще не може бути виконаним.

4.6. У вікні властивостей проєкту перейти до розділу *События сборки – Событие после сборки*, ввести у командний рядок такий текст:
`xcopy /y /d "..\..\MyLib\$(IntDir)MyLib.dll" "$(OutDir)".`

4.7. Виконати в меню команду *Сборка – Собрать решение*.

Результати роботи:

```
Matrix:
0.00    2.65    3.50    4.90    1.36    2.77    0.07   14.42
   -3.43   -5.68   -7.48   -2.36   -0.01    1.36  -12.19  -12.93
   -7.23   -9.63    0.00   -1.87   -2.36  -15.96  -17.05  -18.01
  -10.96    2.36    1.87    0.00  -18.67  -20.00  -21.17    1.36
   -1.36    0.01    2.36  -20.97  -22.51  -23.87   -2.36   -0.01
   -2.77   -1.36  -22.95  -24.73  -26.30    0.00   -1.87   -2.36
   -0.07  -24.61  -26.68  -28.47    2.36    1.87    0.00  -33.81
  -25.80  -28.32  -30.40   -1.36    0.01    2.36  -36.46  -37.59

Vector:
29.67  -42.72  -72.11  -65.20  -68.71  -82.34 -109.42 -157.57

G=-9.63
```

Питання та завдання для самоконтролю

1) Що являють собою бібліотеки функцій C++?
2) Яке призначення заголовних файлів?
3) Що записують у файлах реалізації?
4) Для чого у мові C++ використовують директиви препроцесора? Назвіть відомі Вам директиви.
5) За допомогою якої директиви долучають заголовні файли?

Завдання до самостійної роботи

1) Дати відповіді на контрольні запитання.

2) Написати мовою С++ програмний код заголовного файлу і файлу реалізації бібліотеки з трьома функціями для розв'язання завдань, поданих в табл. 11.2 відповідно до індивідуального варіанта. Написати програмний код основної програми з долученням написаної бібліотеки, викликом її функцій і виведенням результатів обчислень.

3) Створити на комп'ютері мовою С++ власну статичну бібліотеку з розробленими у п. 2 трьома функціями та програмний проєкт з її долученням. Занести результати обчислень до протоколу.

Таблиця 11.2

Варіанти завдань для створення бібліотеки

№ вар.	Індивідуальне завдання
1	1) Обчислити елементи матриці розміром 8×3 за формулою: $a_{ij} = \left(\sin^2 i + \cos^2 j\right)^{\frac{i-5}{j+1}} + 7{,}45\,\mathrm{tg}\left(\frac{i-5}{j+1.5}\right)$. 2) Обчислити елементи вектора як суми елементів стовпців матриці. 3) Значення різниці між абсолютними значеннями максимального та мінімального елементів матриці
2	1) Обчислити елементи матриці розміром 4×6 за формулою: $a_{ij} = \ln^3(7.3i - j + 8) - \frac{\mathrm{e}^j}{j+1}$. 2) Обчислити елементи вектора як середні арифметичні значення кожного рядка матриці. 3) Значення суми додатних елементів матриці
3	1) Обчислити елементи матриці розміром 6×6 за формулою: $a_{ij} = \left(-\frac{2+i}{3+j}\right)^i - \mathrm{e}^{\cos j}$. 2) Обчислити елементи вектора як максимальні елементів рядків матриці. 3) Значення середнього арифметичного елементів матриці з парними індексами
4	1) Обчислити елементи матриці розміром 7×7 за формулою: $a_{ij} = \mathrm{e}^{\frac{i+j}{12.5}} - \sqrt{\left\lvert\mathrm{ctg}(i + j - 50.5)\right\rvert}$. 2) Обчислити елементи вектора як суми елементів головної та неголовної діагоналей матриці. 3) Значення добутку елементів матриці, більших за 1

Продовження табл. 11.2

№ вар.	Індивідуальне завдання
5	1) Обчислити елементи матриці розміром 5×5 за формулою: $a_{ij} = \left(\frac{2}{3j+1.5}\right)^{i} - \lg\frac{e^{i}}{3j+9}$. 2) Обчислити елементи вектора як стовпець матриці з максимальним елементом. 3) Значення добутку елементів неголовної діагоналі матриці
6	1) Обчислити елементи матриці розміром 7×6 за формулою: $a_{ij} = \left(\frac{2+i}{3i+j+1}\right)^{i+j} + \cos\left(e^{\cos(ij-1.5)}\right)$. 2) Обчислити елементи вектора як середні арифметичні значення кожного стовпця матриці. 3) Значення суми модулів від'ємних елементів матриці
7	1) Обчислити елементи матриці розміром 8×7 за формулою: $a_{ij} = \left(\frac{\cos^2(i+1.5)}{2.5+j}\right)^{\sin(i+j+1.5)} + e^{\frac{i}{j+5.5}}$. 2) Обчислити елементи вектора як рядок матриці з мінімальним елементом. 3) Значення кількості додатних елементів матриці, менших 5-ти
8	1) Обчислити елементи матриці розміром 7×7 за формулою: $a_{ij} = \lg^4(i+j+1.2) - 0.5^{i}$. 2) Обчислити елементи вектора як суми елементів рядків матриці з парними індексами стовпців. 3) Значення середньоарифметичного елементів головної діагоналі матриці
9	1) Обчислити елементи матриці розміром 8×5 за формулою: $a_{ij} = (\sin i \cdot \ln(j+1.2))^3 - \frac{(-1.3)^{i+j}}{\lg(i+j+0.2)}$. 2) Обчислити елементи вектора як середні арифметичні значення додатних елементів кожного рядка матриці. 3) Значення мінімального додатного елемента матриці
10	1) Обчислити елементи матриці розміром 8×5 за формулою: $a_{ij} = \cos(2i+j)\lg(i+5j+0.3)$. 2) Обчислити вектор як суми від'ємних елементів стовпців матриці. 3) Значення кількості елементів матриці, більших за 1
11	1) Обчислити елементи матриці розміром 5×5 за формулою: $a_{ij} = \operatorname{ctg}(3i+2j+5) + \ln(i+2j+2)$. 2) Обчислити елементи вектора як квадрати елементів неголовної діагоналі матриці. 3) Сума найменшого та найбільшого елементів матриці

Продовження табл. 11.2

№ вар.	Індивідуальне завдання
12	1) Обчислити елементи матриці розміром 4×7 за формулою: $a_{ij} = \ln\frac{i+2j+2}{\mathrm{e}^{2i-j}} + (0.5i)^{\frac{1}{j+1}}$. 2) Обчислити елементи вектора як рядок матриці з максимальним елементом. 3) Значення середнього арифметичного елементів другого рядка матриці
13	1) Обчислити елементи матриці розміром 6×5 за формулою: $a_{ij} = \sqrt[3]{\lg\left(i^2 + j^2 + 2.5\right)}(i-2.5)^j$. 2) Обчислити елементи вектора як середні арифметичні значення елементів другого та останнього стовпців матриці. 3) Значення добутку елементів матриці зі значенням меншим за 10
14	1) Обчислити елементи матриці розміром 8×4 за формулою: $a_{ij} = \begin{cases} \operatorname{ctg}(i-j+7) + \lg\left(\dfrac{2i-j+7}{i^2-j^2+11.2}\right) & \text{за } j \neq i, \\ \ln^4(i+j+2.3) & \text{за } j = i. \end{cases}$ 2) Обчислити елементи вектора як різниці елементів першого та третього стовпців матриці. 3) Значення кількості елементів зі значенням більшим за середнє арифметичне елементів матриці
15	1) Обчислити елементи матриці розміром 6×6 за формулою: $a_{ij} = \sin(i+j-7.2)^3 \ln(9.2i-j+7.2)$. 2) Обчислити елементи вектора як скалярний добуток елементів рядків матриці на останній стовпець. 3) Значення максимального елемента головної діагоналі матриці
16	1) Обчислити елементи матриці розміром 8×6 за формулою: $a_{ij} = (-1.3)^{i+j} \ln\left(3^{i-j+6.2}\right)$. 2) Обчислити елементи вектора як середні арифметичні знчення елементів першого та останнього рядків матриці. 3) Значення максимального від'ємного елемента матриці
17	1) Обчислити елементи матриці розміром 5×7 за формулою: $a_{ij} = \tan(j-4i+0.5) + \lg(3i+2j+2)$. 2) Обчислити вектор як рядок з мінімальною сумою елементів матриці. 3) Значення кількості елементів матриці, абсолютне значення яких менше 1
18	1) Обчислити елементи матриці розміром 5×8 за формулою: $a_{ij} = \ln^3(9i+j+1) - \mathrm{e}^i$. 2) Обчислити елементи вектора як стовпець матриці з мінімальною сумою елементів. 3) Сума елементів матриці

Продовження табл. 11.2

№ вар.	Індивідуальне завдання
9	1) Обчислити елементи матриці розміром 8×7 за формулою: $a_{ij} = (\sin(7+i))^{j-7} + \mathrm{e}^{\cos(i+j)}$. 2) Обчислити елементи вектора як суми елементів стовпців матриці, значення яких більше за 10. 3) Значення середньоарифметичного елементів останнього рядка матриці
20	1) Обчислити елементи матриці розміром 8×8 за формулою: $a_{ij} = \lg\left(\mathrm{e}^{\frac{7+i}{j+2}}\right) - \sqrt{\lvert\mathrm{tg}(j+1)\rvert}$. 2) Обчислити елементи вектора як скалярний добуток рядків матриці на другий стовпець. 3) Значення добутку елементів останнього стовпця матриці
21	1) Обчислити елементи матриці розміром 8×5 за формулою: $a_{ij} = \begin{cases} \lg(i-j) + \cos(7i-j) & \text{за } i > j; \\ \ln(j-i+7) - \mathrm{e}^{\frac{i}{j-i+1}} & \text{за } i \le j. \end{cases}$ 2) Обчислити елементи вектора як найбільші за модулем елементи рядків. 3) Значення суми елементів матриці зі значенням більшим за середнє арифметичне елементів
22	1) Обчислити елементи матриці розміром 7×7 за формулою: $a_{ij} = \left(\frac{3}{9j-i-1}\right)^{i} - \sin\left(\mathrm{e}^{ij}\right)$. 2) Обчислити елементи вектора як скалярний добуток елементів стовпців матриці на перший рядок. 3) Значення кількості від'ємних елементів у перших трьох рядках матриці
23	1) Обчислити елементи матриці розміром 8×5 за формулою: $a_{ij} = \ln^{2.5}(4i + j + 1.8)\cos(ij)$. 2) Обчислити елементи вектора як стовпець з мінімальною сумою елементів. 3) Значення середнього арифметичного елементів четвертого рядка матриці
24	1) Обчислити елементи матриці розміром 8×7 за формулою: $a_{ij} = \sqrt{\lvert i - 0.3j\rvert}(\cos i + \sin j)^2 - \ln 2$. 2) Обчислити вектор як мінімальні за модулем елементи рядків матриці. 3) Значення кількості додатних елементів матриці
25	1) Обчислити елементи матриці розміром 8×4 за формулою: $a_{ij} = \sqrt{\lg\left(\frac{i+7}{j+4}\right)} - (-2)^{i+j-1}$. 2) Обчислити вектор як модуль суми елементів рядків матриці. 3) Значення кількості від'ємних елементів матриці

Закінчення табл. 11.2

№ вар.	Індивідуальне завдання
26	1) Обчислити елементи матриці розміром 8×3 за формулою: $a_{ij} = \mathrm{tg}\left(\mathrm{e}^{\frac{i+4}{j+2}} \right) - \frac{-6^{i+j-1}}{\mathrm{e}^{3i-2j}}$. 2) Обчислити елементи вектора як кількості додатних елементів рядків. 3) Середнє арифметичне елементів матриці, значення яких належать діапазону [10, 20]
27	1) Обчислити елементи матриці розміром 3×6 за формулою: $a_{ij} = (\sin(i-1) - \cos j)^3 + \ln\frac{i+7}{j+4}$. 2) Обчислити елементи вектора як квадрати значень останніх елементів стовпців матриці. 3) Модуль суми всіх від'ємних елементів матриці
28	1) Обчислити елементи матриці розміром 8×8 за формулою: $a_{ij} = \frac{\sqrt{\lvert\ln(i+2j+2.4)\rvert}}{\sin(i+j+0.5)}$. 2) Обчислити елементи вектора як скалярний добуток елементів рядків матриці на головну діагональ. 3) Значення середніх арифметичних елементів неголовної діагоналі матриці
29	1) Обчислити елементи матриці розміром 6×6 за формулою: $a_{ij} = \frac{2.1^{i+j}}{\ln(i+2)} - (-2)^{i+j}$. 2) Обчислити елементи вектора як середньоарифметичні значення максимального і мінімального елементів кожного стовпця матриці. 3) Значення суми елементів головної діагоналі матриці
30	1) Обчислити елементи матриці розміром 8×6 за формулою: $a_{ij} = \begin{cases} \mathrm{e}^{\frac{i}{j+2}} \sin(i-j) & \text{за } i \neq j; \\ \arcsin((i+j-2)/15) & \text{за } i = j. \end{cases}$ 2) Обчислити елементи вектора як суми максимального і мінімального елементів рядків матриці. 3) Значення найменшого елемента матриці

ТЕМА 5
ПРОГРАМНЕ ОПРАЦЮВАННЯ РІЗНОРІДНИХ ДАНИХ

Лабораторна робота № 13
Робота з символьними даними

Мета роботи: набути практичних навиків програмного опрацювання символьних даних.

Теоретичні відомості

1. Символьний тип даних у C++

Символами вважаються: великі й малі латинські літери, великі й малі літери кирилиці, цифри, пробіл, розділові знаки (`'.'`, `','`, `';'`, `':'`, `'!'`, `'?'`, `'-'`), знаки арифметичних дій (`'+'`, `'-'`, `'*'`, `'/'`, `'='`), службові символи, що відповідають клавішам [Enter], [Esc], [Tab] тощо. У C++ значення символьних констант записуються в одинарних лапках: `'3'`, `'f'`, `'+'`, `'%'`.

Існує єдиний міжнародний стандарт – так звана таблиця ASCII-кодів (American Standard Code for Information Interchange – американський стандартний код для обміну інформацією). Символи ASCII мають коди від 0 до 127, тобто значення першої половини можливих значень байта, хоча часто кодами ASCII називають всю таблицю з 256 символів. Перші 128 ASCII-кодів є єдині для всіх країн, а коди від 128 до 255 називають розширеною частиною таблиці ASCII, де залежно від країни розміщується національний алфавіт і символи псевдографіки. У різних версіях C++ коди символів національного алфавіту можуть мати різні значення.

Символьний тип у C++ зветься **`char`**. Наприклад, при оголошенні змінних

```
char c, s, g;
```

в оперативній пам'яті для кожної з цих трьох змінних буде відведено по одному байту. Коли символьні змінні набудуть певних значень, то комп'ютер запише в пам'ять не самі символи, а їхні коди (у вигляді цілих чисел). Наприклад, замість літери `'A'` зберігатиметься її код 65. Тому, якщо присвоїти символьній змінній певне ціле число, то C++ сприйме його як код символу з таблиці ASCII-кодів.

Символи можна порівнювати. Більшим вважається той символ, в якого код є більший, тобто символ, розміщений у таблиці ASCII-кодів пізніше. Наприклад: `'a' < 'h'`, `'A' < 'a'`.

Специфіка мови C++ при опрацюванні символьних даних полягає в тому, що за замовчуванням тип є знаковим, тобто зберігає значення кодів символів від –128 до 127. При чому від'ємні значення мають коди розширеної частини таблиці ASCII. Наведений фрагмент програмного коду дозволить побачити, як

C++ інтерпретує символи кирилиці:

```
setlocale(LC_ALL,".1251");
for (int i='А'; i<='я'; i++)
{  if (i%8==0) puts("\n");
   printf("%c = %3i  ",(char)i, i);
}
```

```
А = -64  Б = -63  В = -62  Г = -61  Д = -60  Е = -59  Ж = -58  З = -57
И = -56  Й = -55  К = -54  Л = -53  М = -52  Н = -51  О = -50  П = -49
Р = -48  С = -47  Т = -46  У = -45  Ф = -44  Х = -43  Ц = -42  Ч = -41
Ш = -40  Щ = -39  Ъ = -38  Ы = -37  Ь = -36  Э = -35  Ю = -34  Я = -33
а = -32  б = -31  в = -30  г = -29  д = -28  е = -27  ж = -26  з = -25
и = -24  й = -23  к = -22  л = -21  м = -20  н = -19  о = -18  п = -17
р = -16  с = -15  т = -14  у = -13  ф = -12  х = -11  ц = -10  ч =  -9
ш =  -8  щ =  -7  ъ =  -6  ы =  -5  ь =  -4  э =  -3  ю =  -2  я =  -1
```

2. Функції для роботи з символами

У C++ існують спеціальні функції для роботи з символьними даними.

Функція	Призначення
`tolower()`	повертає символ у нижньому регістрі
`toupper()`	повертає символ у верхньому регістрі
Подальші функції перевіряють належність символу до множини:	
`isalnum()`	латинських літер і цифр (`'A' – 'Z'`, `'a' – 'z'`, `'0' – '9'`)
`isalpha()`	латинських літер (`'A' – 'Z'`, `'a' – 'z'`)
`iscntrl()`	керувальних символів (з кодами 0..31 й 127)
`isdigit()`	цифр (`'0' – '9'`)
`isgraph()`	видимих символів, тобто не є відповідними клавішам [Esc], [Tab] тощо
`islower()`	латинських літер нижнього регістру (`'a' – 'z'`)
`isprint()`	друкованих символів (`isgraph()` + пробіл)
`isupper()`	літер верхнього регістру (`'A' – 'Z'`)
`ispunct()`	знаків пунктуації
`isspace()`	символів-розділювачів

Примітка. Вищезазначені функції перевірки й перетворення регістру працюють лише з латинськими літерами. Спроба використати ці функції для символів кирилиці спричинить виведення повідомлення про помилку. Тому для роботи з літерами кирилиці слід використовувати перевірку належності символу до відповідного символьного проміжку, а для перетворення регістру – зменшення чи то збільшення коду символу на різницю кодів між великими та малими літерами (для більшості літер вона становить 32).

У C++ рядки розглядають як масиви, елементи якого мають тип `char`. Рядок може містити символи літер, цифр і спеціальних символів, які записують у подвійних лапках. Ім'я рядка є константним вказівником на перший символ. Нумерація символів у рядку починається з 0. Останній символ рядка – службо-

вий символ, так званий, завершальний нуль-символ `'\0'`, який є ознакою кінця рядка. При оголошенні рядка слід вказати максимальну кількість символів, які будуть зберігатися у рядку, при чому слід враховувати наявність наприкінці рядка нуль-термінатора і відводити додатковий байт для нього:

```
char s[10]; // Рядок s може містити до 9-ти символів
```

Рядок при оголошенні можна ініціалізувати початковим значенням, наприклад так:

```
char s[10] = "abcdef";
```

При цьому перші шість символів рядка набудуть значень кодів зазначених символів, а решта чотири символи – значення '\0'. До речі, якщо рядок при оголошенні ініціалізується, то зовсім не обов'язково вказувати його розмір у квадратних дужках:

```
char s[] = "abcdef";
```

При цьому розмір рядка визначається автоматично, а в кінець рядка дописується символ '\0'.

3. Введення-виведення символьних даних

Для введення одного символу в консолі існує функція `getch()` з бібліотеки `conio.h`.

```
char c = getch();
getch();
```

Оскільки ця функція призупиняє виконання програми й очікує на введення символу, тобто на натиснення будь-якої клавіші, її доволі часто використовують наприкінці консольних програм перед `return 0`, щоб призупинити програму і надати користувачу можливість прочитати результати.

Для введення й виведення рядків у консолі, крім вже знайомих Вам потокових команд C++ `cin>>` та `cout<<` (з бібліотеки `iostream.h`), використовують С-функції `gets/puts` та `scanf/printf` (з бібліотеки `stdio.h`).

1) Специфіка потокової команди введення `cin>>` полягає у тому, що вона вводить послідовність символів до першого пробілу або іншого розділювача, тобто цю команду недоречно використовувати для введення речень з пробілами.

2) Функція `puts(s)` виводить рядок `s` на екран, замінюючи нуль-символ на *Enter*.

3) Функція `gets(s)` зчитує символи з клавіатури до появи символу переведення рядка *Enter* і записує їх у рядок `s` (власне символ *Enter* до рядка не долучається, а замість нього записується нуль-символ), наприклад:

```
const int n=80;
char s[n];
gets(s);
puts(s);
```

Замість функції `gets()` вважається більш безпечним застосовувати функцію `gets_s()`, оскільки вона дозволяє контролювати кількість зчитуваних символів:

```
gets_s(s);
gets_s(s,80);
```

4) Функції форматного введення-виведення **`scanf`** та **`printf`** мають схожий

формат і дозволяють вводити не лише рядки, а й числові дані, залежно від вказаного формату:

scanf (<*формат*>, < *список_змінних*>);

printf (<*формат*>, <*список_змінних*>);

де: *формат* – рядок специфікаторів формату у подвійних лапках. Найбільш поширеними специфікаторами є: `%s` – для рядка символів; `%i` – для цілих чисел; `%f` – для дійсних чисел у фіксованому форматі;

список_змінних – послідовність розділених комами змінних, значення яких вводиться або виводяться.

Так, функція `printf("Результат: ", str)` виводить рядок `str`, а функція `scanf("%s %i", str, &kol)` вводить значення рядка `str` і цілої змінної `kol`.

Для роботи з рядками існують спеціальні функції, серед яких:

`strlen(s)` – обчислення довжини рядка `s` (нуль-символ не враховується);

`strcat(s1, s2)` – приєднання рядка `s2` до кінця рядка `s1`.

Інші функції для роботи з рядками будуть розглянуті у подальшій лабораторній роботі.

4. Встановлення мовних стандартів

Ще однією особливістю C++ є те, що в консольному режимі по-різному інтерпретуються коди розширеної частини таблиці ASCII для введених символьних даних і набраного у програмі тексту. І саме тому текст кирилицею (українською або російською мовою) в консолі виводиться у вигляді абракадабри. Для визначення мовного стандарту (країна або регіон і мова) при роботі з розширеними символами у Visual C++ існує спеціальна функція `setlocale()` з бібліотеки `locale.h`, формат якої має вигляд:

```
char *setlocale(int type, const char *locale);
```

де `type` – аргумент, який вказує на те, які саме дані мовного стандарту слід змінити: `LC_ALL` – усі; `LC_MONETARY` – формат грошових значень; `LC_NUMERIC` – формат числових значень з десятковою точкою або комою; `LC_TIME` – формат дати і часу тощо;

`locale` – аргумент, який задає мовний стандарт у вигляді рядка із зазначенням мови, коду країни і/або кодової сторінки у форматі: `"<мова>_<країна>.<кодова сторінка>"`, при чому всі три складові зазначати не обов'язково. Наприклад, команди `setlocale(LC_ALL,".1251")`; `setlocale(0,"Ukrainian.1251")`; `setlocale(0, "Ukrainian_Ukraine.1251")` і `setlocale(0,"uk-UA.1251")` для України є тотожними, а для США тотожними є використання команд `setlocale(0,"English")`; `setlocale(LC_ALL,"English_United States.1252")` і `setlocale(LC_ALL, "en-US")`, при чому перша з форм є найбільш рекомендованою.

Набір доступних імен мовних стандартів, назв мов, кодів країн і кодових сторінок можна побачити за Інтернет-посиланням: http://msdn.microsoft.com/ru-RU/goglobal/bb896001.aspx. Приміром, функція `setlocale(0,"French_Canada.1252")` встановлює мовний стандарт "французький (Канада)" з кодовою сторінкою 1252.

Якщо в якості значення кодової сторінки аргументу `locale` вказати `".OCP"`, то встановлюватиметься кодова сторінка за замовчуванням.

Позаяк мова C++ опрацьовує символ як один байт, то спроба встановлення кодової сторінки UTF-7 чи UTF-8, які потребують два байти для кожного символу, завершиться помилкою.

Функція `setlocale()` повертає вказівник на рядок, асоційований з параметром `type`. При виникненні помилки функція поверне нульовий вказівник.

Розглянемо на прикладі доцільність почергового використання різних кодувань (команди `setlocale(0,".1251")` і `setlocale(0,".OCP")`) у програмному коді для коректного виведення літер кирилиці, оскільки без використання цих команд кирилиця виводитиметься у вигляді абракадабри. Далі наведено три варіанти програмного коду, а праворуч від них – вигляд виведеного тексту:

1)
```
char s[100];
puts("Введіть рядок s =");
gets_s(s);
puts("Введений Вами рядок s =");
puts(s);
```

```
тТхфіЄ№ Ё фюъ s =
Мова програмування C++
тТхфхэшщ тръш Ё фюъ s =
Мова програмування C++
```

2)
```
char s[100];
setlocale(LC_ALL,".1251");
puts("Введіть рядок s =");
gets_s(s);
puts("Введений Вами рядок s =");
puts(s);
```

```
Введіть рядок s =
Мова програмування C++
Введений Вами рядок s =
?Rў  ЇaR?a ┐гў --п C++
```

3)
```
char s[100];
setlocale(LC_ALL,".1251");
puts("Введіть рядок s =");
setlocale(LC_ALL,"Ukrainian_Ukraine.OCP");
gets_s(s);
setlocale(0,".1251");
puts("Введений Вами рядок s =");
setlocale(0,".OCP");
puts(s);
```

```
Введіть рядок s =
Мова програмування C++
Введений Вами рядок s =
Мова програмування C++
```

Отже, для виведення на екран текстових повідомлень з кирилицею слід використовувати кодування Windows 1251 (команда `setlocale(LC_ALL,".1251")` або `setlocale(0,".1251")`), а для виведення раніш введених символьних даних слід повернутися до початкових налаштувань командою `setlocale(0,".OCP")`.

Приклади програм

Приклад 1. Вивести всі символи та коди таблиці ASCII.

Текст програми:

```
#include <iostream>
using namespace std;
int main()
{
   system("chcp 1251 > null");
   system("color F0");
```

```
    for (int i=32; i<256; i++)
    { if (i%4==0) printf("\n");
      printf("%3i = %c \t",i,(char)i);
    }
    system("pause");
    return 0;
}
```

Результати роботи:

```
 32 =          33 = !          34 = "          35 = #
 36 = $          37 = %          38 = &          39 = '
 40 = (          41 = )          42 = *          43 = +
 44 = ,          45 = -          46 = .          47 = /
 48 = 0          49 = 1          50 = 2          51 = 3
 52 = 4          53 = 5          54 = 6          55 = 7
 56 = 8          57 = 9          58 = :          59 = ;
 60 = <          61 = =          62 = >          63 = ?
 64 = @          65 = A          66 = B          67 = C
 68 = D          69 = E          70 = F          71 = G
 72 = H          73 = I          74 = J          75 = K
 76 = L          77 = M          78 = N          79 = O
 80 = P          81 = Q          82 = R          83 = S
 84 = T          85 = U          86 = V          87 = W
 88 = X          89 = Y          90 = Z          91 = [
 92 = \          93 = ]          94 = ^          95 = _
 96 = `          97 = a          98 = b          99 = c
100 = d         101 = e         102 = f         103 = g
104 = h         105 = i         106 = j         107 = k
108 = l         109 = m         110 = n         111 = o
112 = p         113 = q         114 = r         115 = s
116 = t         117 = u         118 = v         119 = w
120 = x         121 = y         122 = z         123 = {
124 = |         125 = }         126 = ~         127 = ⌂
128 = Ђ         129 = Ѓ         130 = ‚         131 = ѓ
132 = „         133 = …         134 = †         135 = ‡
136 = €         137 = ‰         138 = Љ         139 = ‹
140 = Њ         141 = Ќ         142 = Ћ         143 = Џ
144 = ђ         145 = ‘         146 = ’         147 = “
148 = ”         149 = •         150 = –         151 = —
152 = ⌂         153 = ™         154 = љ         155 = ›
156 = њ         157 = ќ         158 = ћ         159 = џ
160 =           161 = Ў         162 = ў         163 = Ј
164 = ¤         165 = Ґ         166 = ¦         167 = §
168 = Ё         169 = ©         170 = Є         171 = «
172 = ¬         173 = -         174 = ®         175 = Ї
176 = °         177 = ±         178 = І         179 = і
180 = ґ         181 = µ         182 = ¶         183 = ·
184 = ё         185 = №         186 = є         187 = »
188 = ј         189 = Ѕ         190 = ѕ         191 = ї
192 = А         193 = Б         194 = В         195 = Г
196 = Д         197 = Е         198 = Ж         199 = З
200 = И         201 = Й         202 = К         203 = Л
204 = М         205 = Н         206 = О         207 = П
208 = Р         209 = С         210 = Т         211 = У
212 = Ф         213 = Х         214 = Ц         215 = Ч
216 = Ш         217 = Щ         218 = Ъ         219 = Ы
220 = Ь         221 = Э         222 = Ю         223 = Я
224 = а         225 = б         226 = в         227 = г
228 = д         229 = е         230 = ж         231 = з
232 = и         233 = й         234 = к         235 = л
236 = м         237 = н         238 = о         239 = п
240 = р         241 = с         242 = т         243 = у
244 = ф         245 = х         246 = ц         247 = ч
248 = ш         249 = щ         250 = ъ         251 = ы
252 = ь         253 = э         254 = ю         255 = я
```

Приклад 2. Ввести рядок й вивести ASCII-коди його символів.

Текст програми:

```
include <iostream>
using namespace std;

int main()
{
   char s[100];
   setlocale(LC_ALL,".1251");
   puts("Введіть рядок: ");
   setlocale(LC_ALL,".OCP");
   gets_s(s);
   for (int i=0; i<strlen(s); i++)
     printf("%c = %3i  ",s[i],(int)s[i]);
   system("pause");
   return 0;
}
```

Результати роботи:

```
Введіть рядок:
0123ZzVvЯяАаБб!
0 =  48  1 =  49  2 =  50  3 =  51  Z =  90  z = 122  V =  86  v = 118  Я = -97
 я = -17  А = -128  а = -96  Б = -127  б = -95  ! =  33
```

Приклад 3. Ввести рядок і вивести замість цифр їх ASCII-коди.

Текст програми:

```
include <iostream>
using namespace std;

int main()
{
  char s[100];
  puts("Input one string: ");
  gets_s(s);
  for (int i=0; i<strlen(s); i++)
    if (isdigit(s[i])) printf(" %i ",int(s[i]));
    else printf("%c",s[i]);
  system("pause");
  return 0;
}
```

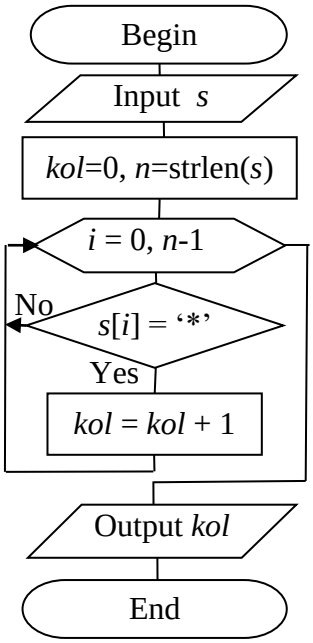
Результати роботи:

```
Input one string:
digits: 0123456789
digits:  48  49  50  51  52  53  54  55  56  57
```

Приклад 4. Ввести рядок й обчислити кількість символів '*' у цьому рядку.

Текст програми та схема алгоритму:

```
#include <iostream>
using namespace std;
int main()
{ setlocale(LC_ALL,".1251");
  char s[100];
  puts("Введіть рядок: ");
  gets_s(s);
  int i, n, kol = 0;
  n = strlen(s);     // Довжина рядка
  for (i = 0; i < n; i++)
    if (s[i] == '*') kol++;
  printf("Кількість символів * у рядку =  %i", kol);
  system("pause");
  return 0;
}
```



Результати роботи:

```
Введіть рядок:
q***н****f**
Кількість символів * у рядку = 9
```

Приклад 5. Ввести рядок і визначити, чи є у ньому знаки пунктуації.

Текст програми:

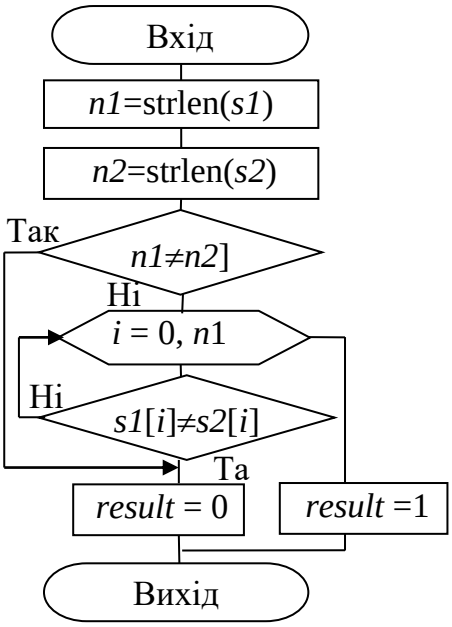
```
#include <iostream>
using namespace std;

int znaki(char* s)
{
  for (int i=0; s[i]!='\0'; i++)
   if(s[i]=='.' || s[i]==',' || s[i]==';' || s[i]=='!' || s[i]=='?' || s[i]=='-')
   // або if(ispunct(s[i])) для латиниці
      return 1; //Якщо зустрічається знак пунктуації, функція поверне значення 1,
  return 0;     //а якщо не зустрілося жодного, поверне значення 0
}
int main()
{
   setlocale(LC_ALL,".1251");
   char s[100];   int res;
   puts("Введіть рядок: ");   gets_s(s);
   res = znaki(s);
   if (res) puts("Розділові знаки присутні");
    else puts("Розділових знаків немає");
   system("pause");
   return 0;
}
```

Результати роботи:

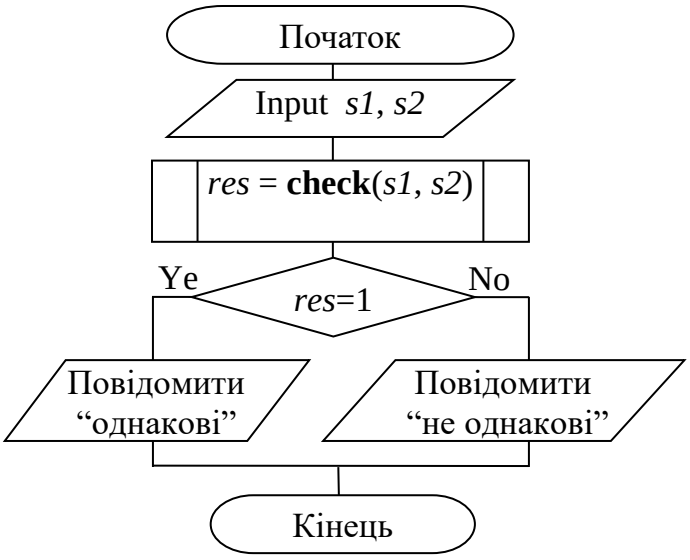
```
Введіть рядок:
Hello, world!
Роздільові знаки присутні
```

```
Введіть рядок:
Несе Галя воду
Роздільових знаків немає
```

Приклад 6. Ввести два рядки й визначити, чи є вони однаковими.



Блок-схема функції **check()**

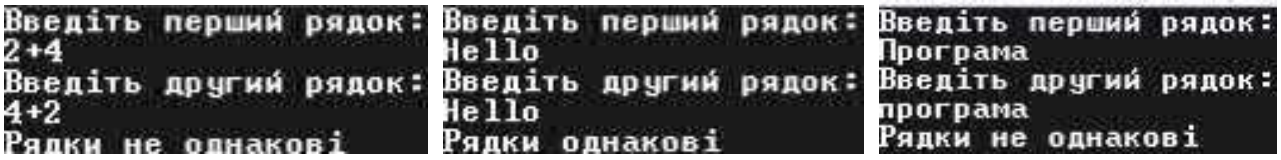


Блок-схема функції **main()**

Текст програми:

```
#include <iostream>
using namespace std;
int check (char* s1, char* s2)
{ int i, n1, n2;
  n1=strlen(s1); n2=strlen(s2);
  if (n1 != n2) return 0; // Якщо довжини рядків не збігаються, повернути 0
  for (i=0; i<n1; i++)           // Якщо знайшлись неоднакові символи,
    if(s1[i] != s2[i]) return 0;  // повернути 0
  return 1;      // Якщо неоднакових символів не знайдено, повернути 1
}
int main()
{  char s1[100], s2[100];
   setlocale(LC_ALL,".1251");
   puts("Введіть перший рядок: ");  gets_s(s1);
   puts("Введіть другий рядок: ");  gets_s(s2);
   int res = check (s1, s2);
   if(res) puts("Рядки однакові");
    else puts("Рядки не однакові ");
   system("pause");
   return 0;
}
```

Результати роботи:



Приклад 7. Перевести до верхнього регістру (у великі літери) всі голосні літери латиниці рядка та обчислити їхню кількість.

Текст програми:

```
include <iostream>
using namespace std;

int lowreg(char* s)
{  int k=0;
   for (int i=0; i<strlen(s); i++)
     if (s[i]=='a'||s[i]=='o'||s[i]=='e'||s[i]=='i'||s[i]=='u')
     { s[i]-=32; k++; }
   return k;
}
int main()
{  char s[100];
   puts("Input one string: "); gets_s(s);
   int k=lowreg(s);
   puts(s);
   cout<<"The number of vowels - "<<k<<endl;
    system("pause");
   return 0;
}
```

```
Input one string:
Open the windows
OpEn thE wIndOws
The number of vowels - 4
```

Приклад 8. Видалити останнє слово з рядка.

Текст програми:

```
include <iostream>
using namespace std;

void  DeleteLastWord(char* s)
{
  int i=0, ind=0;
  while (s[strlen(s)-1]==' ')  s[strlen(s)-1]='\0'; // видалити кінцеві пробіли
  while (s[i]!='\0')
    if(s[i++] ==' ') ind=i; // ind – номер  останнього пробілу в рядку
  s[ind]='\0';              // записати замість пробілу символ кінця рядка
}

int main()
{
  setlocale(LC_ALL,".1251");
  char s[100];
  puts ("Введіть рядок: ");
```

```
  gets_s(s);
  DeleteLastWord(s);
  puts("\nРядок з видаленим останнім словом:");
  setlocale(LC_ALL,".OCP");
  puts(s);
  system("pause");
  return 0;
}
```

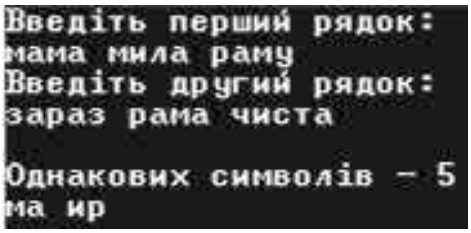
```
Введіть рядок:
Малеча зростає швидко

Рядок з видаленим останнім словом:
Малеча зростає
```

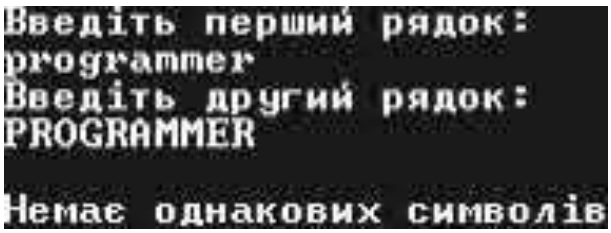
Приклад 9. Ввести два рядки, визначити і вивести всі спільні для обох рядків символи та обчислити їхню кількість.

Текст програми:

```
include <iostream>
using namespace std;

int  SameLetters(char* s1, char* s2, char* rez)
{
  rez[0]='\0';
  int i, j, k, m, kol=0;
  for (i=0; s1[i]!='\0'; i++)
    for (j=0; s2[j]!='\0'; j++)
      if (s1[i] == s2[j])
       { for (k=m=0; rez[k]!='\0'; k++)
           if (s1[i]==rez[k]) m=1;
         if (!m)
          { rez[kol]=s1[i];
            rez[kol+1]='\0';
            kol++;
          }
       }
 return kol;
}
int main()
{
  char s1[100], s2[100],s3[100];
  setlocale(LC_ALL,".1251");    puts("Введіть перший рядок: ");
  setlocale(LC_ALL,".OCP");     gets_s(s1);
  setlocale(LC_ALL,".1251");    puts("Введіть другий рядок: ");
  setlocale(LC_ALL,".OCP");     gets_s(s2);
  int k = SameLetters(s1, s2, s3);
  setlocale(LC_ALL,".1251");
  if (!k) puts("\nНемає однакових символів");
   else printf("\nОднакових символів - %i\n", k);
  setlocale(LC_ALL,".OCP");
  puts(s3);
  system("pause");
  return 0;
}
```

Результати роботи:



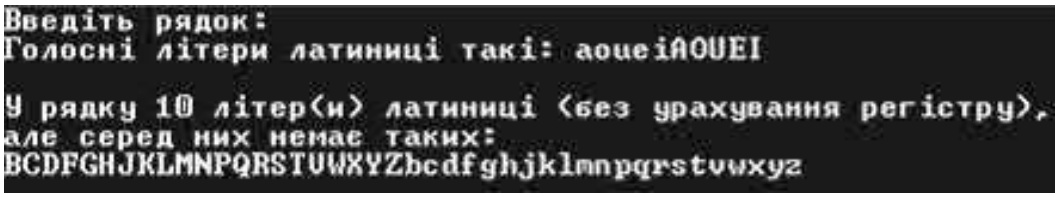


Приклад 10. Вивести ті літери латиниці (і малі, і великі), яких немає у рядку.

Текст програми:

```
#include <iostream>
using namespace std;

int  NotLetters(char* s, char *rez)
{ char c;   int i, j, k, kol=0;
  for (c='A',i=0; c<='Z'; c++) rez[i++]=c;
  for (c='a'; c<='z'; c++) rez[i++]=c;
  rez[i]='\0';
  for (i=0; s[i]!='\0'; i++)
    if (s[i]>='A' && s[i]<='Z' || s[i]>='a' && s[i]<='z' )
      for (j=0; rez[j]!='\0'; j++)
        if (s[i]==rez[j])
          { kol++;
            for (k=j; rez[k]!='\0'; k++)  rez[k]=rez[k+1];
          }
  return kol;
}

int main()
{
  char s1[100], s2[55];
  setlocale(LC_ALL,".1251");
  puts(" Введіть рядок: ");     gets_s(s1);
  int k = NotLetters(s1,s2);
  if (!k) puts("\n У рядку немає літер латиниці:");
  else printf("\n У рядку %i літер(и) латиниці (без урахування регістру),
              \n але серед них немає таких:\n", k);
  puts(s2);
   system("pause");
  return 0;
}
```

Результати роботи:



```
Введіть рядок:
Ходить котик тихо, буде мишкам лихо...

У рядку немає літер латиниці:
ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz
```

Приклад 11. Ввести рядок і вивести окремо у стовпець усі слова цього рядка.

Текст програми:

```
#include <iostream>
using namespace std;
void words(char* st)
{  char sl[100];      int k=0, i, n;
   strcat(st," ");
   n=strlen(st);
   if (n<2) return;
   sl[0]='\0';
   for (i=0; i<n; i++)
     if (st[i] != ' ')
     {
       sl[k]=st[i];
       sl[k+1]='\0';
       k++;
     }
     else
     {
     if (strlen(sl)>0) puts(sl);
     sl[0]='\0';
     k=0;
}  }
int main()
{
   char st[100];
   setlocale(LC_ALL,".1251");
   puts("Введіть рядок: ");
   setlocale(LC_ALL,".OCP");
   gets_s(st);
   words(st);
    system("pause");
   return 0;
}
```

Результати роботи:

```
Введіть рядок:
A NULL pointer indicates an error
A
NULL
pointer
indicates
an
error
```

```
Введіть рядок:
Мама мила раму. Тепер рама чиста.
Мама
мила
раму.
Тепер
рама
чиста.
```

Питання для самоконтролю

1) Дати означення поняття ASCII. Яка є кількість ASCII-кодів?
2) Скільки байтів виділяється для змінної типу `char`?
3) Які операції можна виконувати над символами?
4) Назвати вирази, які матимуть значення 1 (`true` – істина – "так"):
   а) `'0' < 'y'`
   б) `'5' > 'f'`
   в) `'F' > 'f'`
   г) `'w' > 'W'`
   д) `' ' > '0'`
   е) `'я' > 'ю'`
5) Як у програмі змінити регістр символів латиниці?

Лабораторне завдання

1) У протоколі лабораторної роботи дати відповіді на контрольні запитання.

2) У протоколі лабораторної роботи написати мовою С++ програмний код функції та основної програми для розв'язання індивідуальних завдань з опрацювання символьних даних (завдання вибрати з табл. 13.1 … 13.2).

3) Створити на комп'ютері програмні проєкти мовою С++ для реалізації написаних програм. Занести результати обчислень до протоколу.

Таблиця 13.1

Індивідуальні завдання середнього рівня складності

№ вар.	Завдання
1	Визначити кількість малих літер у введеному рядку
2	Замінити великі літери у введеному рядку на пробіли
3	Визначити кількість цифр у введеному рядку
4	Замінити латинські літери у введеному рядку на знаки оклику
5	Замінити розділові знаки у введеному рядку на символи '#'
6	Визначити кількість голосних літер латиниці у введеному рядку
7	Вивести замість літер введеного рядка їхні ASCII-коди
8	Змінити регістр малих латинських літер у введеному рядку
9	Обчислити кількість великих літер у введеному рядку
10	Замінити у введеному рядку цифри на символи '+'
11	Обчислити кількість латинських літер у введеному рядку
12	Визначити індекс першої цифри у введеному рядку
13	Обчислити кількість розділових знаків у введеному рядку
14	Вивести літери латиниці з парними ASCII-кодами та обчислити їх кількість
15	Вивести замість малих літер введеного рядка їхні ASCII-коди
16	Визначити індекс останнього пробілу в рядку
17	Обчислити кількість великих латинських літер у введеному рядку
18	Змінити регістр великих латинських літер у введеному рядку
19	Вивести замість пробілів введеного рядка його ASCII-код
20	Замінити малі латинські літери на крапки у введеному рядку

Закінчення табл. 13.1

№ вар.	Завдання
21	Вивести літери латиниці з непарними ASCII-кодами, обчислити їх кількість
22	Замінити великі латинські літери у введеному рядку на символ коми
23	Вивести замість розділових знаків введеного рядка їхні ASCII-коди
24	Замінити всі пробіли у введеному рядку на знаки підкреслення
25	Визначити, чого більше – ком чи крапок – у рядку, а може їхня кількість однакова?
26	Вивести замість великих літер введеного рядка їхні ASCII-коди
27	У введеному рядку замінити літери латиниці на символ '*'
28	Визначити індекс першого пробілу у введеному рядку
29	Замінити у введеному рядку голосні літери латиниці на символ '%'
30	Визначити, чого у рядку більше – голосних літер латиниці чи цифр, а може їх кількість однакова?

Таблиця 13.2

Індивідуальні завдання підвищеного рівня складності

№ вар.	Завдання
1	Вивести ті слова рядка, довжина яких більше 7-ми
2	Вивести ті слова введеного рядка, які починаються з літери К
3	Знайти найдовшу послідовність літер А, які йдуть у введеному рядку підряд, і визначити її довжину
4	Ввести рядок, який містить круглі дужки. Вивести на екран усі символи, які розміщені усередині цих дужок
5	Визначити, скільки разів у рядку зустрічається комбінація символів "С++"
6	Вивести усі слова, які починаються і закінчуються на одну й ту саму літеру без урахування регістру
7	Визначити, чи є у рядку комбінація символів "С++", і якщо так, вивести індекс її першого входження у рядок
8	Переставити символи рядка у зворотному порядку й обчислити кількість символів, від перестановки яких рядок не змінився
9	Визначити, чи читається рядок однаково зліва направо і справа наліво з урахуванням регістру. Якщо ні, вивести рядок у зворотному порядку
10	Визначити, чи містить рядок лише малі латинські літери. Якщо ні, вивести усі символи, які не є малими латинськими літерами, та їхні індекси
11	Визначити найбільшу кількість пробілів, які йдуть підряд
12	Вивести найдовшу з послідовностей цифр, що є у рядку
13	Змінити регістр малих літер латиниці та обчислити їхню кількість
14	Введіть рядок, всі слова якого зашифровані (записані у зворотному порядку). Розшифрувати текст
15	Видалити з рядка перше слово

Закінчення табл. 13.2

№ вар.	Завдання
16	Видалити з рядка всі пробіли
17	Обчислити кількість слів, які починаються й закінчуються на один і той самий символ з урахуванням регістру
18	Вивести символи першого рядка, яких немає у другому рядку
19	Обчислити кількість слів, які починаються з великої літери
20	Вивести ті великі латинські літери, яких немає у рядку
21	Вивести ті слова рядка, які містять дефіс
22	Визначити і вивести найдовше слово у рядку
23	Визначити середню довжину слів у рядку
24	Вивести слова рядка, довжина яких менше 6, та їх порядкові номери
25	Вивести знаки пунктуації, яких немає у рядку
26	Перевірити, чи є введений рядок послідовністю двійкових цифр (0 і 1). Якщо так, перевести його у десяткову систему. Якщо ні, вивести індекс першого "недвійкового" символу
27	Вивести цифри, яких немає у рядку
28	Вивести маленькі латинські символи, яких немає у рядку
29	Видалити всі зайві (подвійні, початкові та кінцеві) пробіли у рядку
30	Визначити і вивести найкоротше слово у рядку

Самостійна робота № 12
Рядки char*

Мета роботи: набути практичних навиків програмного опрацювання рядків типу char*.

Теоретичні відомості

1. Оголошення С-рядків char*

С-рядок може бути оголошеним в один з нижче наведених **способів**:

```
1) char* s;            // Оголошення вказівника на перший символ рядка;
                       // пам'ять під сам рядок не виділяється.
2) char ss[15];        // Оголошення рядка  ss з 14 символів;
                       // пам'ять виділяється компілятором.
3) const int n = 10;
   char st[n];         // Оголошення рядка st з n-1 (тобто 9) символів;
                       // пам'ять виділяється компілятором.
4) int n = 10;
   char* str = new char[n];  // Оголошення рядка str з n-1 (тобто 9)
                             // символів; пам'ять виділяється динамічно.
```

При зазначенні довжини рядка слід враховувати завершальний нуль-символ. Наприклад, у вищенаведеному рядку str варто зберігати не 10, а лише 9 символів.

Зауважимо, що при оголошенні рядка першим способом пам'ять під рядок не виділяється і це може бути дуже небезпечним, оскільки у ту ж саму пам'ять можуть бути розміщені інші змінні і рядок буде втрачено.

При оголошенні рядок можна ініціювати рядковою константою, при цьому нуль-символ формується автоматично після останнього символу:

```
char str[10] = "Vasia";
```

Крім функцій, зазначених у попередній роботі, для введення рядків можна використовувати також cin.getline(). Наприклад:

```
char s[40];
cout<<"Введіть рядок: "<<endl;
cin.getline(s,40);
```

2. Функції для роботи з рядками char*

C++ має багату колекцію функцій опрацювання рядків char*, найпоширеніші з яких наведено далі в таблиці. Всі ці функції зі стандартної бібліотеки <string.h>. Деякі з цих функцій, наприклад: strlen(), strcpy(), опрацьовують рядки, оголошені в будь-який з чотирьох раніш розглянутих способів. Але більшість функцій потребує, щоб рядок був оголошений як вказівник на тип char*.

Функція	Призначення	Формат
strlen()	повертає довжину рядка (без урахування символу закінчення рядка)	size_t strlen(char *s);
strcat()	приєднує s2 до s1 і, як результат, повертає s1	char *strcat(char *s1,char *s2);

Функ-ція	Призначення	Формат
strncat()	приєднує до рядка s1 перші n символів з рядка s2	char *strncat(char *s1, char *s2, size_t n);
strcmp()	порівнює рядки і повертає нульове значення, (якщо рядки однакові s1=s2), від'ємне (s1<s2) чи додатне (s1>s2). Порівняння відбувається посимвольно і в якості результату повертається різниця кодів перших неоднакових символів	int *strcmp(char *s1, char *s2);
strncmp()	порівнює лише перші n символів рядка s1 з n символами рядка s2	int *strncmp(char *s1, char *s2, size_t n);
stricmp()	порівнює два рядки, не розрізняючи прописні й малі літери латиниці	int stricmp(const char *s1, const char *s2);
strnicmp()	порівнює лише перші n символів двох рядків, не розрізняючи прописні й малі літери латиниці	int strnicmp(const char *s1, const char *s2, size_t maxlen)
strcpy()	копіює в s1 рядок s2, повертає s1, при цьому старе значення s1 втрачається	char *strcpy(char *s1, char *s2);
strncpy()	замінюються перші n символів рядка s1 першими n символами рядка s2	char *strncpy(char *s1, char *s2, size_t n);
strchr()	відшукує в рядку перше входження символу ch і повертає вказівник на цей символ, тобто частину рядка s, починаючи з символу ch і до кінця рядка. Якщо символу в рядку немає, результат – NULL	char *strchr(char *s, char ch);
strrchr()	шукає в рядку s останнє входження символу ch і повертає частину рядка s, починаючи з останнього входження символу ch і до кінця рядка	char *strrchr(char *s, int ch);
strstr()	шукає підрядок s2 в рядку s1, повертає частину рядка s1, починаючи з першого спільного символу для обох рядків і до кінця s1	char *strstr(char *s1, char *s2);
strspn()	повертає довжину початкового сегмента рядка s1, символи якого є в рядку s2	size_t strcspn(char *s1, char *s2);
strcspn()	повертає індекс першого символу в рядку s1, який є спільним для обох рядків (нумерація індексів символів з нуля)	size_t strcspn(char *s1, char *s2);
strlwr()	перетворює латинські літери до нижнього регістру	char *strlwr(char *s);
strupr()	перетворює латинські літери до верхнього регістру	char *strupr(char *s);
strset()	замінює усі символи рядка s символом ch	char *strset(char *s, char ch);
strnset()	замінює лише перші n символів рядка s символом ch	char *strnset(char *s, char ch, size_t n);
strpbrk()	знаходить місце першого входження в рядок s1 будь-якого символу рядка s2 і повертає частину рядка s1, починаючи з цього символу і до кінця	char *strpbrk(char *s1, const char *s2);
strrev()	реверс рядка s	char *strrev(char *s);
strtok()	повертає частину (лексему) рядка s1 від поточної позиції вказівника до розділового символу, зазначеного у рядку s2; звичайно використовується для перетворення рядка у послідовність лексем	char *strtok(char *s1, const char *s2);

Приклади програм

Приклад 1. Ввести рядок й визначити довжину першого й останнього слів у рядку (розділювачем слів вважати пробіл).

Розв'язок. Для визначення довжини першого слова слід знайти індекс першого пробілу. Для визначення довжини останнього слова треба знайти індекс останнього пробілу, обчислити різницю довжини рядка та цього індексу і зменшити її на 1.

Текст програми:

```
#include <iostream>
using namespace std;
int main()
{
 setlocale(LC_ALL,".1251");
 char *s=new char [100];
 puts(" Введіть рядок:");
 gets_s(s, 100);
 int pp=0, op=0, n=strlen(s), i;
 for (i=0; i<n; i++)
   if (s[i]==' ')   // Зупинитись, якщо зустріли перший пробіл
     { pp=i;  cout << "\n Довжина першого слова: " << pp << endl;  break;}
 if(!pp) {cout<<"\nПробілів немає"<<endl; cin.get();  return 0;}
 for (i=n-1; i>=0; i--)
   if (s[i]==' ') { op=i;  break;} // Зупинитись, якщо зустріли пробіл
 cout<< "\n Довжина останнього слова:  " << n-op-1 << endl;
 delete []s;
 system("pause");
 return 0;
}
```

Така само програма зі створенням функцій:

```
#include <iostream>
#include <string.h>
using namespace std;
int first_space(char *s) // Пошук першого пробілу
{ int p=-1;
  for (int i=0; s[i]!='\0'; i++)
    if (s[i]==' ') {p=i; break;}
  return p;
}
int last_space(char *s) // Пошук останнього пробілу
{ int p=-1;
  for (int i=0; s[i]!='\0'; i++)
    if (s[i]==' ') p=i;
  return p;
}
```

```
int main()
{
 setlocale(LC_ALL,".1251");
 int pp, op, n;
 char *s=new char [100];
 puts(" Введіть рядок:");
 gets_s(s, 100);
 n=strlen(s);
 pp=first_space(s);
 if (pp > -1)
   cout<< "\n Довжина першого слова:  " << pp << endl;
 else
   {  cout<<"\nПробілів немає" << endl;  cin.get();  return 0;}
 op=last_space(s);
 cout<< "\n Довжина останнього слова:  " << n-op-1 << endl;
 delete [] s;
 system("pause");
 return 0;
}
```

Результат виконання програми:

```
 Введіть рядок:
програма

Пробілів немає
```

```
 Введіть рядок:
Одеська національна академія зв'язку - super

 Довжина першого слова:  7
```

Приклад 2. На основі введеного рядка створити новий рядок з маленьких голосних літер латиниці, які містяться у введеному рядку.

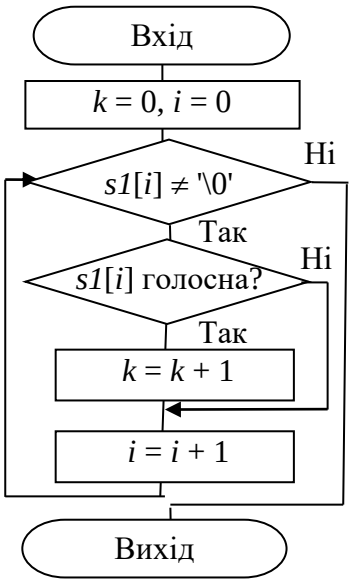
Розв'язок. Спочатку треба обчислити кількість маленьких голосних літер латиниці у введеному рядку `s`, щоб виділити необхідну кількість пам'яті для символів створюваного рядка `s1`. Після цього можна копіювати у новий рядок `s1` голосні маленькі літери латиниці з рядка `s`, а наприкінці долучити завершальний нуль до нового рядка.

Текст програми:

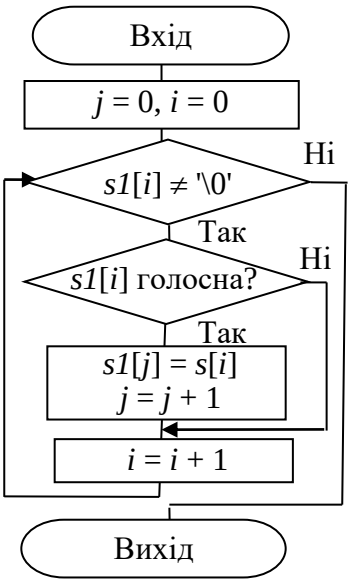
```
#include <iostream>
#include <string.h>
using namespace std;
int main()
{
 setlocale(LC_ALL,".1251");
 int i, n, kol=0, j=0;
 char *s=new char [100];
 puts(" Введіть рядок:");
 gets_s(s, 100);
 n=strlen(s);
```

```
 for (i=0; i<n; i++)
   if (s[i]=='a' || s[i]=='e' || s[i]=='i' || s[i]=='o' || s[i]=='u') kol++;
 char *s1 = new char [kol+1];
 for (i=0; i<n; i++)
   if (s[i]=='a' || s[i]=='e' || s[i]=='i' || s[i]=='o' || s[i]=='u')
     { s1[j]=s[i]; j++; }
 s1[kol]='\0';
 puts("\n Новий рядок з голосних літер  ");
 setlocale(0, ".OCP");
 puts (s1);
 delete [] s;
 system("pause");
 return 0;
}
```

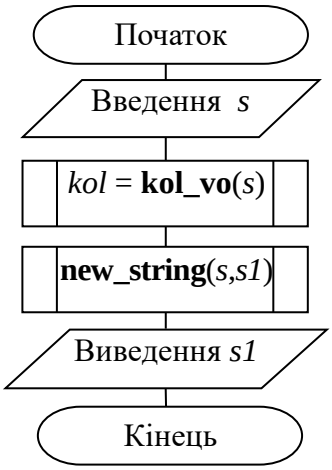
Така ж сама програма зі створенням функцій:



Блок-схема функції
kol_vo()



Блок-схема функції
new_string()



Блок-схема функції
main()

```
#include <iostream>
using namespace std;
int kol_vo(char *s)  // Визначення кількості голосних маленьких літер
{ int k=0, i;
  for (i=0; s[i]!='\0'; i++)
   if (s[i]=='a' || s[i]=='e' || s[i]=='i' || s[i]=='o' || s[i]=='u') k++;
  return k;
}
void new_string(char *s, char *s1) // Заповнення нового рядка
{ int i, j=0;
  for (i=0; s[i]!='\0'; i++)
      if (s[i]=='a' || s[i]=='e' || s[i]=='i' || s[i]=='o' || s[i]=='u')
        { s1[j]=s[i]; j++; }
   s1[j]='\0';
}
```

```
int main()
{
 setlocale(LC_ALL,".1251");
 char *s = new char [100];
 puts(" Введіть рядок:");
 gets_s(s, 100);
 int kol = kol_vo(s);
 char *s1 = new char [kol+1];
 new_string(s, s1);
 puts("\n Новий рядок з голосних літер  ");
 setlocale(0, ".OCP");
 puts(s1);
 delete []s;
 system("pause");
 return 0;
}
```

Результат виконання програми:

```
 Введіть рядок:
Visual Studio Integrated Development Environment

 Новий рядок з голосних літер
iuauioeaeeeoeioe
```

Приклад 3. Ввести рядок і вивести всі заголовні літери, які стоять на початку слів. Слова розділяються одним чи кількома пробілами.

Текст програми:

```
#include <iostream>
using namespace std;
int main()
{ setlocale(LC_ALL,".1251");
 int i, n;
 char *s=new char [100];
 puts(" Введіть рядок:");
 gets_s(s, 100);
 n=strlen(s);
 puts(" Результат:");
 setlocale(0, ".OCP");
 if (s[0]!=' ' && isupper(s[0])) // Перше слово перевіряємо окремо
      cout<<s[0]<<endl;
 for (i=0; i<n-1; i++)
   if (s[i]==' ' && s[i+1]!=' ' && isupper(s[i+1]))
      cout<<s[i+1]<<endl;
 delete [] s;
 system("pause");
 return 0;
}
```

```
 Введіть рядок:
Development Tools and Language
 Результат:
D
T
L
```

Приклад 4. Ввести рядок і вивести його слова окремо без розділових знаків.

Розв'язок. Для розбивання рядка на слова (лексеми) можна застосувати функцію `strtok()`, яка має два аргументи: 1) рядок, який треба розбити на лексеми, і 2) рядок, який містить символи-розділювачі.

Перший виклик функції `strtok()` у нижченаведеній програмі

```
t = strtok(s, " .,;?!-");
```

присвоює `t` вказівник на першу лексему (слово) в рядку `s`. Другий аргумент містить можливі розділові символи. Функція `strtok()` шукає перший символ у рядку `s`, який не є розділювачем. Це початок першого слова. Потім функція знаходить наступний розділювач у рядку і замінює його нуль-символом (`'\0'`). Цим закінчується поточне слово. Функція `strtok()` зберігає вказівник на наступний символ рядка `s` за даною лексемою (словом) і повертає вказівник на поточну лексему (слово).

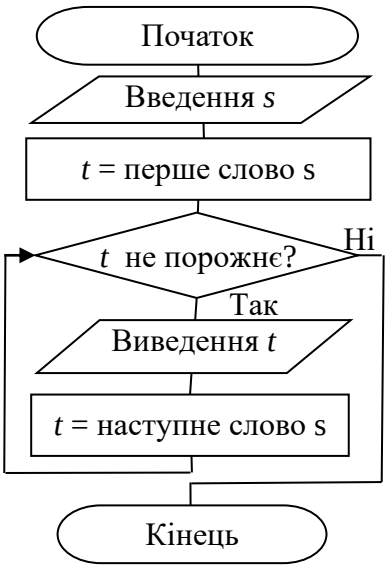
У подальших викликах `strtok()` у програмі для почергового розбивання рядка `s` на слова першим аргументом функції є `NULL` для того, щоб виокремлення наступного слова рядка `s`, відбувалося з позиції, яку було збережено попереднім викликом `strtok()`. Якщо лексем при виклику `strtok()` більше немає, `strtok()` повертає `NULL` і відбувається вихід із циклу.

Зверніть увагу, що `strtok()` модифікує вхідний рядок; тому, якщо рядок після виклику `strtok()` буде знову використовуватись у програмі, слід завчасно зробити копію рядка.

Цей алгоритм можна застосовувати для виокремлення чисел, але тоді у якості розділових знаків не слід застосовувати крапку, яка може інтерпретуватися як десяткова крапка.

Текст програми у консолі:

```
#include <iostream>

int main()
{
 setlocale(LC_ALL,".1251");
 char *s = new char [80], *t;
 puts("Введіть рядок, який треба розбити на лексеми:");
 gets_s(s, 80);
 puts("\nЛексеми:  ");
 setlocale(0, ".OCP");
 t = strtok(s, " .,;?!-");
 while (t != NULL)
 { puts(t);
   t = strtok(NULL, " .,;?!-");
 }
 delete []s;
 system("pause>>void");
 return 0;
}
```



Результат виконання програми:

```
Введіть рядок, який треба розбити на лексеми:
Hi, Tony! How are you? – I’m OK, thanks.

Лексеми:
Hi
Tony
How
are
you
I’m
OK
thanks
```

Приклад 5. Ввести рядок і видалили з нього друге слово.

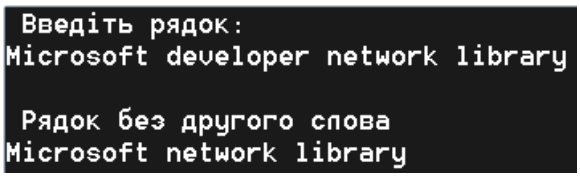
Розв’язок. У нижченаведеному коді розв’язання організовано в окремій функції `del_word()`, яка видаляє з рядка `s`, починаючи з позиції `k`, `n` символів. У цій функції спочатку копіюється з рядка `s` перше слово з пробілом у допоміжний рядок `s1`. Після чого долучається до `s1` решта рядка `s` без слова, яке має бути видаленим (адреса початку цієї частини рядка обчислюється за допомогою адресної арифметики – зсув на `k+n+1` символів від початку рядка `s`).

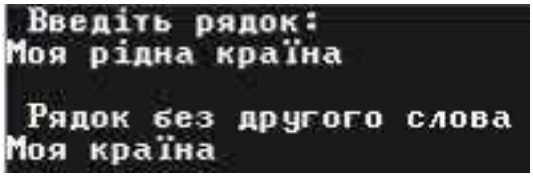
Текст програми

```
#include <iostream>
#include <string.h>
using namespace std;

void del_word(char *s, int k, int n)
{ char *s1=new char [strlen(s)+1];
  strncpy(s1, s, k);
  s1[k]='\0';
  strcat(s1, s+k+n+1);
  strcpy(s,s1);
  delete [] s1;
}
int main()
{ setlocale(LC_ALL,".1251");
 int i, kol=0, p1=-1, p2=-1;
 char *s=new char [100];
 puts(" Введіть рядок:"); gets_s(s, 100);
 strcat(s," ");
 for (i=0; s[i]!='\0'; i++)
  if (s[i]==' ')
   if (p1==-1) p1=i;
   else { p2=i; break; }
 del_word(s, p1+1, p2-p1-1);
 puts("\n Рядок без другого слова ");
 setlocale(0, ".OCP");
 puts(s);
```

```
 delete []s;
 system("pause>>void");
 return 0;
}
```

Результат виконання програми:





Приклад 6. Ввести рядок і видалили з нього слова, коротші 4-х символів.

Розв'язок. Для розв'язання задачі створимо новий рядок `s1` зі слів рядка `s`, довжина яких не менше 4-х символів.

Текст програми

```
#include <iostream>
#include <string.h>

int main()
{ setlocale(LC_ALL,".1251");
  char* s=new char [80], *t;   char* s1=new char [80];
  strcpy(s1,"\0");
  puts(" Введіть рядок:");  gets_s(s, 80);
  t=strtok(s, " .,;?!-");
  while (t != NULL)
    { if (strlen(t)>=4) { strcat(s1,t); strcat(s1, " \0"); }
      t = strtok(NULL, " .,;?!-");
    }
 puts("\n Рядок без коротких слів ");
 setlocale(0, ".OCP");
 strcpy(s, s1);
 puts(s);
 delete [] s;
 delete [] s1;
 system("pause>>void");
 return 0;
}
```

Приклад 7. Для введеного рядка поміняти місцями вказані номерами слова.

Текст програми

```
#include <iostream>
#include <string.h>
int main()
{ setlocale(LC_ALL,".1251");
  char s[100] = {'\0'}, rez[100] = {'\0'},*word;
  char swap[25], a[25][20];     // для переставляння слів
```

```
  int k = -1, i, l, r;
  printf("Введіть  рядок:\n");  gets(s);
  word = strtok(s, " \'\":-!,");
  while(word)
  {  k++;                        // кількість слів
     strcpy(a[k], word);
     word = strtok(NULL, " \'\":-!,");
  }
  printf("Всього %d слів(ова)\n" ,k+1);
  printf("\nВведіть номери слів у рядку для обміну: ");
  if(scanf("%d %d", &l, &r)==0 || l>r || r>k+1)
  {  printf("\nНеправильне введення номерів слів для обміну!!!\n");
     return 1;
  }
  strcpy(swap,a[l-1]);           // переставляння слів у масиві слів
  strcpy(a[l-1],a[r-1]);
  strcpy(a[r-1],swap);
  for(i=0;i<=k;i++)
  {  strcat(rez,a[i]);           // поєднання слів у рядок
     strcat(rez," ");
  }
  puts("\nПісля обміну слів:");
  setlocale(LC_ALL,".OCP");
  puts(rez);
  system("pause>>void");
  return 0;
}
```

```
Введіть  рядок:
Нехай все буде добре
Всього 4 слів(ова)

Введіть номери слів у рядку для обміну: 3 4


Після обміну слів:
Нехай все добре буде
```

Питання для самоконтролю

1) Навести відомі способи оголошення С-рядка.
2) Чи можна присвоїти С-рядку частину іншого рядка? Якщо так, навести приклад.
3) Яка функція дозволяє з'єднати два С-рядки? Навести приклад.
4) Навести способи введення С-рядка у консолі.

Завдання до самостійної роботи

1) Дати відповіді на контрольні запитання.

2) Написати мовою С++ програми для розв'язання індивідуальних завдань з використанням стандартних функцій для опрацювання рядків (завдання вибрати з табл. 12.2 … 12.3).

3) Створити на комп'ютері програмні проєкти мовою С++ для реалізації написаних програм. Занести результати обчислень до протоколу.

Таблиця 12.2

Індивідуальні завдання базового рівня складності

№ вар.	Завдання
1	Вивести перше слово введеного рядка
2	Знайти індекси початку й кінця другого слова у введеному рядку
3	Вивести перше слово введеного рядка у зворотному порядку
4	Вивести останнє слово введеного рядка
5	Визначити кількість слів у введеному рядку. Слова розділяються одним чи декількома пробілами
6	Для двох введених рядків побудувати третій, на початку якого стоятиме довший з рядків, а за ним через пробіл – коротший
7	Створити новий рядок з першого й останнього слів введеного рядка
8	Вивести перші літери всіх слів введеного рядка. Слова розділяються одним чи декількома пробілами
9	Створити новий рядок з цифр, які містяться у введеному рядку
10	Вивести індекси всіх входжень літери S до рядка
11	Визначити кількість слів рядка, які починаються з літер латиниці
12	Вести рядок з літер латиниці і виконати над ним такі перетворення регістру: якщо перша літера рядка є великою (верхній регістр) перевести до верхнього регістру весь рядок , і навпаки, якщо перша літера мала – до нижнього
13	Вивести останні літери усіх слів введеного рядка. Слова розділяються одним чи декількома пробілами
14	Вивести індекси всіх цифр, які є у введеному рядку
15	Вивести всі розділові знаки та їхні індекси у введеному рядку
16	Перевернути у зворотному порядку останнє зі слів введеного рядка
17	Перевести перше зі слів введеного латиницею рядка до верхнього регістру
18	Вивести усі маленькі латинські літери, які стоять на початку слів. Слова розділяються одним чи декількома пробілами
19	Вивести індекси всіх ком та крапок, які є у введеному рядку
20	Визначити кількість зайвих пробілів у рядку (подвійних і початкових)
21	Створити новий рядок з великих літер латиниці введеного рядка
22	Визначити кількість слів, після яких є розділові знаки, та вивести ці знаки
23	Визначити, чи є у рядку знаки арифметичних дій. Якщо так, то вивести на екран ці знаки
24	Перевернути у зворотному порядку перше зі слів введеного рядка
25	Видалити усі пробіли у введеному рядку
26	Створити новий рядок з першого та другого слів введеного рядка
27	Створити новий рядок з малих літер латиниці введеного рядка
28	Створити новий рядок з розділових знаків введеного рядка
29	Визначити, чи є у введеному рядку знаки арифметичних дій. Якщо так, то створити новий рядок з цих знаків
30	Перетворити останнє слово введеного латиницею рядка до нижнього регістру

Таблиця 12.3

Індивідуальні завдання підвищеного рівня складності

№ вар.	Завдання
1	Видалити ті слова рядка, які мають цифри
2	Вивести найдовше слово введеного рядка у зворотному порядку
3	Створити новий рядок зі слів введеного рядка, довжина яких більше 6-ти
4	Вивести на екран ті слова введеного рядка, які починаються й закінчуються на один і той самий символ
5	Видалити передостаннє слово введеного рядка
6	Вивести найкоротше слово введеного рядка
7	Ввести рядок та окремо слово, яке є в цьому рядку. Видалити з рядка перше входження цього слова
8	Видалити всі слова введеного рядка, які починаються на літеру А
9	Створити рядок зі слів введеного рядка, довжина яких менше 5-ти символів
10	Видалити з рядка слова, які починаються і закінчуються однаковою літерою
11	Для введеного рядка вставити після кожного слова знак оклику
12	Видалити ті слова рядка, які закінчуються цифрою
13	Видалити з рядка усі зайві пробіли (подвійні, початкові і кінцеві)
14	Переставити у введеному рядку перше й останнє слова
15	Для двох введених рядків поміняти місцями їхні перші слова
16	У введеному рядку видалити всі розділові символи
17	Створити новий рядок з тих слів введеного рядка, які починаються з малих літер латиниці
18	Переставити у введеному рядку перше і друге слова
19	У введеному рядку видалити всі символи, які не є цифрами та крапкою
20	Вставити перед та після кожного слова введеного рядка символ ‘*’
21	У введеному рядку видалити слова з парними номерами
22	Створити новий рядок зі слів введеного рядка, які починаються з великих літер латиниці
23	Створити новий рядок зі слів введеного рядка з непарними номерами
24	Вивести стовпчиком у зворотному порядку всі слова введеного рядка
25	Створити новий рядок з найкоротшого й найдовшого слів введеного рядка
26	Видалити з рядка найкоротше слово
27	Вивести ті слова введеного рядка, які не містять цифр
28	Ввести рядок і окремо слово. Вставити це слово після першого слова рядка
29	Створити новий рядок зі слів введеного рядка, які починаються з літери К
30	Видалити з рядка найдовше слово

Лабораторна робота № 14

Робота з текстовими файлами

Мета роботи: набути практичних навиків програмного створення та редагування текстових файлів засобами C++.

Теоретичні відомості

1. Загальні відомості про файли

Файлами є іменовані області пам'яті на зовнішньому носії, призначені для довготривалого зберігання інформації. Файли мають *імена* й є організовані в ієрархічну деревоподібну структуру з *каталогів* (тек) і простих файлів.

Часто використовується така метафора: якщо уявляти собі файли як книжки (лише зчитування) і блокноти (зчитування й записування), які стоять на полиці, то відкриття файлу – це вибір книжки чи блокнота за заголовком на його обкладинці й відкриття обкладинки (на першій сторінці). Після відкриття можна читати, дописувати, викреслювати і правити записи, перегортати книжку. Сторінки можна зіставити з блоками файлу, а полицю з книжками – з каталогом.

Для доступу до даних файлу з програми в ній слід прописати функцію відкриття чи створення цього файлу, тим самим встановити зв'язок між ім'ям файлу і певною файловою змінною у програмі. Доступ до даних файлу відбувається з так званої позиції зчитування-записування, яка автоматично просувається при операціях зчитування-записування, тобто файл є видимим послідовно. Існують, щоправда, функції для довільного змінювання цієї позиції.

C++ надає засоби опрацювання двох типів файлів: *текстових* і *бінарних*.

Текстові файли призначено для зберігання текстів, тобто сукупності символьних рядків змінної довжини. Кожен рядок завершується керувальною послідовністю `'\n'`, а розділювачами слів та чисел у рядку є пробіли й символи табуляції. Оскільки вся інформація текстового файлу є символьною, програмне опрацювання такого файлу полягає в читанні рядків, виокремлюванні з рядка слів і, за потреби, перетворюванні цифрових символьних послідовностей на числа відповідними функціями перетворювання. Створювати і редагувати текстові файли можна не лише в програмі, а і в якому завгодно текстовому редакторі, наприклад, Word, WordPad чи Блокноті.

Бінарні (двійкові) файли зберігають дані у тому самому форматі, в якому вони були оголошені, а їхній вигляд є такий самий, як і в пам'яті комп'ютера. І тому відпадає потреба у використанні розділювачів: пробілів, керувальних послідовностей, а отже, розмір використовуваної пам'яті порівняно з текстовими файлами з аналогічною інформацією є значно меншим. Окрім того, немає потреби у застосуванні функцій перетворювання числових даних. Але кожне опрацювання даних бінарних файлів можливе лише за наявності програми, якій має бути відомо, що саме і в якій послідовності зберігається у цьому файлі.

2. Робота з текстовими файлами у стилі С

У мові С *файл* розглядається як потік послідовності байтів. Вся робота з файлом виконується через файлову змінну – вказівник на структуру типу `FILE`, означену в стандартній бібліотеці `stdio.h`. Саме у цій бібліотеці описані усі базові функції для роботи з файлами. Тому, якщо в програмі планується робота з файлами, цей заголовний файл треба долучити:

```
#include <stdio.h>
```

Оголошення файлової змінної – вказівника потоку може мати вигляд:

```
FILE *f;
```

Функція `fopen()` для відкривання файлу має такий синтаксис:

```
FILE *fopen(const char *filename, const char *mode);
```

Перший параметр `filename` визначає ім'я фізичного файлу, який відкривається і пов'язується з вказівником `f`. Другий параметр `mode` задає режим відкривання файлу, тобто визначає, які дії будуть доступні для виконання з даними у відкритому файлі (табл. 14.1).

До зазначених у табл. 14.1 специфікаторів наприкінці чи перед знаком `"+"` може дописуватись символ чи то `"t"` – для текстових файлів, чи `"b"` – для бінарних (двійкових) файлів.

Функція `fopen()` повертає вказівник на об'єкт, який керує потоком, тобто адресу пам'яті початку потоку. Наприклад, створити файл з ім'ям `Prim.txt` можна так:

```
f = fopen ("Prim.txt", "wt");
```

Таблиця 14.1

Специфікатори режиму відкривання файлів

Параметр	Опис
`r`	відкрити файл лише для зчитування даних
`r+`	відкрити файл для зчитування й записування даних
`a`	відкрити чи створити файл для записування даних у кінець файлу
`a+`	відкрити чи створити файл для зчитування чи то записування даних у кінець файлу
`w`	створити файл для записування даних
`w+`	створити файл для зчитування і записування даних

Якщо файл відкрити не вдалося, `fopen()` повертає нульовий вказівник `NULL`. Для уникання помилок після відкриття файлу варто перевірити, чи насправді файл відкрився:

```
if (f == NULL) { cout<<"Файл не вдалось відкрити"; return; }
```

Припинити роботу з файлом можна за допомогою функції

```
fclose (FILE *f).
```

Ця функція закриває файл, на який посилається параметр функції, зберігаючи змінення.

Наведемо приклад відкривання текстового файлу для зчитування.

```
FILE *f;
// Перевіряємо, чи повертає ця функція нульовий вказівник
if ((f=fopen("t.txt","rt"))==NULL)
{ cout<<"Неможливо відкрити файл";    return; }
. . .       // Сюди вставляються команди зчитування з файлу
fclose(f);  // Закриття файлу
```

З текстового файлу можна читати цілі рядки й окремі символи. Варто звернути увагу на те, що тип FILE* перейшов в C++ зі стандартного C, і такий файл «розуміє» лише C-рядки, тобто рядки типу char*.

Зчитування рядка з файлу здійснюється функцією fgets():

```
char *fgets (char *s, int m, FILE *stream);
```

де s – перший параметр – рядок типу char*;
m – кількість читаних символів (байтів);
stream – вказівник на потік даних файлу.

Перевірка кінця файлу здійснюється функцією feof():

```
int feof (FILE *stream);
```

Розглянемо докладніше використання цих функцій:

```
char s[50];
do { fgets(s,50,f);
     cout<<s;
   } while (!feof(f));
```

Цей приклад ілюструє зчитування даних з файлу порядко́во за допомогою функції fgets() і виведення рядків на екран. Перший параметр функції fgets() – це C-рядок, в який читається черговий рядок текстового файлу. Зчитування рядка відбувається до появи символу кінця рядка '\n' або ж припиняється, коли прочитано m-1 символ. У нашому прикладі m=50. Отже, якщо файл містить рядок, який має понад 50 символів, то буде прочитано лише перші 49 символів. При цьому поточна позиція файлу залишиться в тому самому рядку й за подальшого використання функції fgets() читатиметься залишок рядка. Третій параметр зазначає потік, з якого здійснюється зчитування.

У даному прикладі при зчитуванні всіх даних з файлу f використовується функція feof(f), яка перевіряє, чи не прочитано символ кінця файлу. Після зчитування цього символу функція feof(f) поверне ненульове значення – і цикл перерветься.

До речі, інколи використання функції feof() призводить до появи дубліката останнього рядка файлу, тому рекомендовано використовувати в умові циклу функцію зчитування даних, наприклад fgets(), або контролювати добігання кінця файлу, тобто перевіряти, чи прочитано всі записані у файлі символи (нагадаємо, що 1 символ – 1 байт). Наведений приклад зчитування рядків з файлу можна тепер записати в такий спосіб:

```
while (fgets(s,50,f))
{ if (s[strlen(s)-1]=='\n')  s[strlen(s)-1] = '\0';
  puts(s);
}
```

Зчитування форматованих даних можна також здійснювати за допомогою функції fscanf():

```
int fscanf (FILE *stream, const char *format[, address.]);
```

Наведемо приклад використання цієї функції:

```
float r;
while ( fscanf(f,"%e",&r)>0 ) cout<<r;
```

У даному прикладі здійснюється зчитування даних з файлу, який містить дійсні числа. За розділювачі поміж числами вважаються пробіли. Перший параметр f функції fscanf() визначає файл, з якого відбувається зчитування. Другий параметр задає формат рядка аргументів, заданих їхніми адресами. Функція fscanf() є цілого типу і повертає, як своє вихідне значення, кількість прочитаних елементів. При форматованому читанні часто можуть виникати помилки через невідповідність форматів чи то кінець файлу. Для уникання таких помилок доцільно організовувати перевірки кількості прочитаних функцією fscanf() елементів. У наведеному прикладі формат "%e" визначає дійсне число з рухомою крапкою. Прочитане дійсне число з файлу зберігається за адресою змінної r типу float.

Рядок форматування параметра format будується з послідовності символів-специфікаторів типів читаних даних, найпоширеніші з яких наведено у табл. 14.2.

Записувати дані у текстовий файл можна за допомогою функції:

```
int fputs (const char *s, FILE *stream);
```

де s – рядок типу char;
stream – файловий потік.

Таблиця 14.2

Специфікатори параметра format

Символ	Значення, що вводиться	Тип аргументу
i, l	десяткове, вісімкове чи шістнадцяткове ціле число	int, long
d, D	десяткове ціле число	int, long
e, E	дійсне число з рухомою крапкою	float
f, F	дійсне число з фіксованою крапкою	float
s	рядок символів	char s[]
c	символ	char

Для записування даних до файлу треба відкрити файл у форматі записування даних у кінець файлу, за потреби перетворити рядок на тип char і записати дані до файлу. Послідовність процесу показано у фрагменті програми:

```
FILE *f;          // Оголошення файлової змінної
f=fopen("a.rtf","at+");
if (f==0) { cout<<"Не вдається створити файл!";  return 0; }
char s[40];
gets(s);          // Введення рядка s
// Долучання до рядка символу [Enter], інакше у файлі
strcat(s,"\n");// все буде записано одним рядком
fputs(s,f);     // Записування рядка s у файл
fclose(f);
```

Записування у текстовий файл можна здійснити також за допомогою функції fprintf():

```
int fprintf (FILE *stream, const char *format [, argument]);
```

Ця функція є подібна до функції fscanf(), але має ширші можливості побудови рядка форматування, наприклад:

```
char s[20];
strcpy(s, "Іванов");
int year=1985;
fprintf(F, "ХАРАКТЕРИСТИКА\nспівробітник %s, %i р.н.\n", &s, year);
```

Унаслідок виконання цих команд до файлу буде записано таке:

```
ХАРАКТЕРИСТИКА
співробітник Іванов, 1985 р.н.
```

Як бінарні, так і текстові файли дозволяють переміщувати поточну позицію зчитування-записування. Для визначення поточної позиції файлу, яка автоматично зміщується на кількість опрацьованих байтів, використовується функція ftell():

```
long int ftell(FILE *stream);
```

А змінити поточну позицію файлу можна за допомогою функції fseek():

```
int fseek(FILE *stream, long offset, int whence);
```

Ця функція задає зсув на кількість байтів offset щодо точки відліку, яка задається параметром whence. Параметр whence може набувати таких значень:

Константа	whence	Точка відліку
SEEK_SET	0	початок файлу
SEEK_CUR	1	поточна позиція
SEEK_END	2	кінець файлу

Наприклад, для переміщення поточної позиції на початок файлу можна скористатися функцією

```
fseek(f, 0, SEEK_SET);
```

або

```
fseek(f, 0, 0);
```

За допомогою функцій ftell() та fseek() можна визначити сумарний обсяг пам'яті у байтах, який займає файл. Для цього є достатньо перемістити у кінець файлу:

```
fseek(f, 0, SEEK_END);
int d = ftell(f);
```

Отже, послідовність роботи з файлом при записуванні даних є така:

1) Відкрити чи то створити файл fname.txt для записування (або для зчитування й записування) у кінець файлу:

```
f=fopen("fname.txt", "at+");
```

2) Записати дані у файл f однією з команд:

```
fputs(s, f);  // Записування С-рядка s
// Форматоване записування С-рядка nazva, дійсного числа price і цілого kol
fprintf(f, "%s %6.2f %i\n", nazva, price, kol);
```

3) Закрити файл:

```
fclose(f);
```

Для зчитування даних з файлу треба виконати таку послідовність дій:

1) Відкрити файл для зчитування

```
f=fopen("fname.txt","rt+");
```

2) Здійснити зчитування даних з файлу однією з команд:

```
fgets(s, 80, f);  // Зчитування С-рядка s довжиною у 80 символів
// Форматоване зчитування за адресами відповідних змінних
fscanf(f, "%s %f %i\n", &nazva, &price, &kol);
```

3) Для зчитування послідовно всіх даних з файлу треба організувати цикл з умовою, яка перевіряє досягання кінця файлу, використовуючи чи то функцію `feof(f)`, яка є ідентична до `true` при досяганні кінця файлу, чи функцію зчитування даних, яка дає нульовий результат за досягання кінця файлу. Можна використовувати інформацію про сумарний розмір файлу і контролювати досягання кінця файлу функцією `ftell(f)`.

4) Закрити файл: `fclose(f);`

У прикладах, що наводяться нижче, всі ці можливості послідовно подано у програмах.

Приклади програм

Приклад 1. Створити в корневій папці диска D за допомогою програми *Блокнот* текстовий файл з ім'ям `myfile.txt` і ввести до нього кілька рядків. Написати програму, яка виводить зміст цього файлу на екран і обчислює кількість рядків у файлі та їхню середню довжину.

Текст програмного коду:

```
#include <iostream>
#include <stdio.h>
using namespace std;
int main()
{
  setlocale(0,".1251");
  char* name="d:\\myfile.txt";
  FILE* f;
  char s[100];
  int sum=0, kol=0;
  f=fopen(name, "rt"); // Відкрити файл для читання як текстовий
  if(f==NULL)
     {cout<<"Неможливо відкрити файл\n";
      system("pause"); return 0;
     }
  cout<<"\nПерегляд файлу"<<endl;
  while (fgets(s, 100, f)) // Зчитувати рядки із файлу,
```

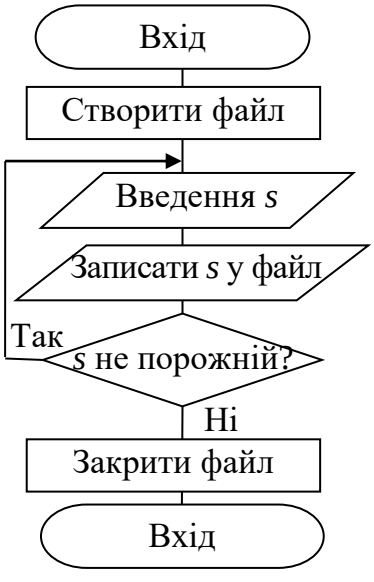
```
  {  // допоки вони не закінчаться (до кінця файлу)
  // Вилучити останній символ (інакше після кожного рядка буде виведено порожній)
     s[strlen(s)-1]='\0';
     puts(s);                    // Вивести рядок на екран
     // Додати довжину цього рядка до загальної довжини файлу
     sum+=strlen(s)-1;
     kol++;                      // Збільшити кількість рядків
  }
  fclose(f);
  cout<<"\nКількість рядків у файлі="<<kol<<endl;
  // Якщо у файлі є рядки, обчислити та вивести середню довжину рядків
  if (kol) cout<<"\nСередня довжина рядків="<<sum/kol<<endl;
  system("pause");
  return 0;
}
```

Результати виконання програми:

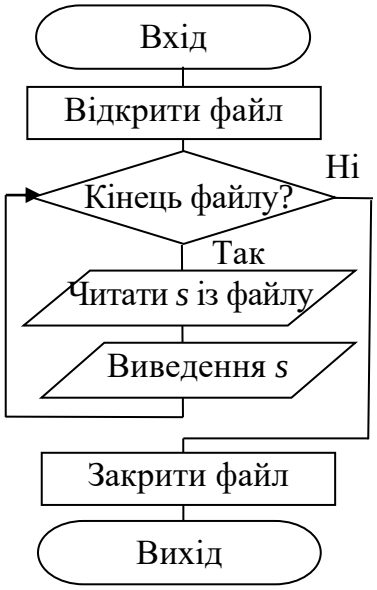
```
Перегляд файлу
У даному прикладі здійснюється зчитування даних з файлу.
За розділювачі поміж числами вважаються пробіли.
Перший параметр f функції fscanf() визначає файл.
Другий параметр задає формат рядка аргументів,
заданих їхніми адресами.

Кількість рядків у файлі=5

Середня довжина рядків=44
```

Приклад 2. Створити текстовий файл (кінець введення – порожній рядок). Вивести спочатку увесь вміст файлу на екран, а тоді рядки довжиною понад 20 символів.

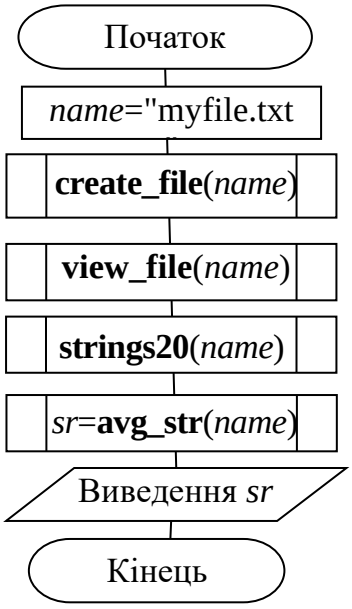
Схеми алгоритмів:



Блок-схема функції **create_file()**



Блок-схема функції **view_file()**



Блок-схема функції **main()**

Текст програмного коду:

```
#include <iostream>
#include <stdio.h>
using namespace std;
void create_file(char* name) // Функція створення файлу
{
  FILE* f;
  char s[100];
  f=fopen(name, "wt");   // Відкрити файл для створення як текстовий
  if (f==NULL)           // Перевірити, чи відкрився файл
   { cout<<"Неможливо створити файл\n"; return;}
  cout<<"Введіть рядки"<<endl;
  do                     // Цикл, поки не введено порожній рядок,
   { gets_s(s,100);      // ввести рядок із клавіатури,
     fputs(s, f);        // записати його до файлу
     fputs("\n", f);     // і перейти на новий рядок
   }
  while (strcmp(s,""));
  fclose(f);             // Закрити і зберегти файл
}
void view_file(char* name) // Функція перегляданння файлу
{
  FILE* f;
  char s[100];
  f=fopen(name, "rt"); // Відкрити файл для читання як текстовий
  if (f==NULL)
    { cout<<"Неможливо відкрити файл\n"; return; }
  cout<<"\nПерегляд файлу»<<endl;
  while (fgets(s, 100, f))
  {
     s[strlen(s)-1]='\0';
     puts(s);
  }
  fclose(f);
}
// Функція виведення рядків довжиною понад 20 символів
void strings20(char* name)
{ FILE* f;   char s[100];
  f=fopen(name, "rt");
  if (f==NULL) { cout<<"Cannot open file\n"; return; }
  cout<<"\nРядки довжиною понад 20 символів:"<<endl;
  while (fgets(s, 100, f))
  {  s[strlen(s)-1] = '\0';
     if (strlen(s)>20) puts(s);
  }
  fclose(f);
}
```

```
int main()
{
  char* name="myfile.txt";    // Ім'я фізичного файлу в папці проєкту
  create_file(name);
  view_file(name);
  strings20(name);
  system("pause");
  return 0;
}
```

Результати виконання програми:

```
Введіть рядки
There are a large number of functions
to handle file I/O (Input Output) in C
When dealing with files, there are two types of files:
1. Text files
2. Binary files

Перегляд файлу
There are a large number of functions
to handle file I/O (Input Output) in C
When dealing with files, there are two types of files:
1. Text files
2. Binary files

Рядки довжиною понад 20 символів:
There are a large number of functions
to handle file I/O (Input Output) in C
When dealing with files, there are two types of files:
```

Приклад 3. Створити файл з рядків (кінець введення – порожній рядок). Визначити кількість рядків, які містять цифри.

Розв'язок. Для створення та переглядання файлу скористаймося функціями `create_file()` і `view_file()` з прикладу 2. За потреби додамо команди змінення локалізації.

Текст програмного коду:

```
// Функція виведення рядків з цифрами та обчислення їх кількості.
int digits(char* name)
{
  FILE* f;
  char s[100];
  int k=0;
  f=fopen(name, "rt");
  if (f==NULL)
    { cout<<"Cannot open file\n"; return 0; }
  setlocale(0,".1251");
  cout<<"\nРядки, які містять цифри:"<<endl;
  setlocale(0,".OCP");
```

```
  while (fgets(s, 100, f))
  {
     s[strlen(s)-1]='\0';
     for (int i=0; i<strlen(s); i++)
       if (s[i]>='0' && s[i]<='9')
        { k++;
          puts(s);
          break;
        }
  }
  fclose(f);
  return k;
}

int main()
{
  setlocale(0,".1251");
  char* name = "myfile.txt";
  create_file(name);
  view_file(name);
  int kol = digits(name);
  setlocale(0,".1251");
  cout << "\nКількість рядків з цифрами=" << kol << endl;
  system("pause");
  return 0;
}
```

Результати виконання програми:

```
Введіть рядки
C++ - мова програмування високого рівня.
Розроблена Б'ярном Страуструпом
1979 року та початково отримала назву <Сі з класами>.
Згодом Страуструп перейменував мову на С++ у 1983 р.
Базується на мові С. Вперше описана стандартом
ISO/IEC 14882:1998, актуальним є стандарт ISO/IEC 14882:2014

Перегляд файлу
C++ - мова програмування високого рівня.
Розроблена Б'ярном Страуструпом
1979 року та початково отримала назву <Сі з класами>.
Згодом Страуструп перейменував мову на С++ у 1983 р.
Базується на мові С. Вперше описана стандартом
ISO/IEC 14882:1998, актуальним є стандарт ISO/IEC 14882:2014

Рядки, які містять цифри:
1979 року та початково отримала назву <Сі з класами>.
Згодом Страуструп перейменував мову на С++ у 1983 р.
ISO/IEC 14882:1998, актуальним є стандарт ISO/IEC 14882:2014

Кількість рядків з цифрами=4
```

Приклад 4. Створити файл з рядків (кінець введення – порожній рядок). Вивести слова, які починаються і закінчуються на той самий символ.

Розв'язок. Для створення та переглядання файлу скористаймося функціями `create_file()` і `view_file()` з прикладу 2. За потреби додамо команди змінення локалізації.

Текст програмного коду:

```
// Функція виведення слів, які починаються і закінчуються на той самий символ
void words(char* name)
{
  FILE* f;
  char s[100], *t;
  f = fopen(name, "rt");
  if (f == NULL)
    { cout<<"Cannot open file\n"; return; }
  setlocale(0,".1251");
  cout<<"\nСлова, які починаються та закінчуються на той самий символ:"
     <<endl;
  setlocale(0,".OCP");
  while (fgets(s, 100, f)>0)
  {
     s[strlen(s)-1]='\0';
     t = strtok(s, " .,;?!-");
     while (t != NULL)
     {
       if (t[0] == t[strlen(t)-1])
         puts(t);
       t = strtok(NULL, " .,;?!-");
     }
  }
  fclose(f);
}

int main()
{
  setlocale(0,".1251");
  char* name="myfile.txt";
  create_file(name);
  view_file(name);
  words(name);
  system("pause");
  return 0;
}
```

Результати виконання програми:

```
Введіть рядки
сос слон кіт
кошеня лелека апельсин
боб поле сніг піп

Перегляд файлу
сос слон кіт
кошеня лелека апельсин
боб поле сніг піп

Слова, які починаються та закінчуються на той самий символ:
сос
боб
піп
```

Приклад 5. Ввести відомості про товари у текстовий файл: назву, ціну товару та кількість товару на складі. Прочитати дані з файлу і віднайти відомості про товари, кількість яких є менше за 30. Визначити назву файлу.

Текст програми:

```
#include <stdio.h>
#include <iostream>
using namespace std;

int main( )
{
  system("color F0");
  FILE *f;                  // Оголосити файлову змінну
  char s[]="proba.txt";     // і визначити імені файлу
  f = fopen(s,"rt+");       // Відкрити файл для зчитування і записування
  if (f==NULL) f=fopen(s,"wt+");// якщо файл не було відкрито, створити його
  if (f==NULL)
    { cout<<"Не вдається відкрити файл!";  return 0; }
  fseek (f, 0, SEEK_END);  // Перейти у кінець файлу
  char nazva[15];
  float price;
  int kol;
  char yn;                  // Ознака бажання дописувати нові рядки у файл
rep:
  setlocale(0,".1251");
  cout<<"Будете вводити рядки для записування у файл? y/n"<<endl;
  cin>>yn;
  if (yn != 'y') goto peregljad;
  cout<<"Введіть рядок для записування у текстовий файл у форматі:
        \n назва товару, ціна, кількість на складі"<<endl;
  setlocale(0,".OCP");
  scanf("%s \t%f \t%i",&nazva,&price,&kol);
```

```
  fprintf(f,"%s \t%6.2f \t%i\n", nazva, price, kol); // Дописати у файл
  goto rep;
peregljad:
  fseek(f, 0, 0);
  setlocale(0,".1251");
  cout<<"\nВміст файлу:\n";
  setlocale(0,".OCP");
  while (!feof(f)) // Контроль досягання кінця файлу
  {
    // Зчитати рядок з файлу
    fscanf(f,"%s \t%f \t%i\n",&nazva,&price,&kol);
    for (int i=strlen(nazva);i<=15; i++)
      strcat(nazva," ");
    printf("%s \t%6.2f \t%i\n",nazva,price,kol); // Вивести дані за форматом
  }
  // Визначити розмір файлу
  fseek(f, 0, SEEK_END);
  int  len = ftell(f);
  setlocale(0,".1251");
  cout<<"Розмір файлу - "<< len << "\n\n";
  fseek(f, 0, 0); // Перейти на початок файлу
  setlocale(0,".OCP");
  int  n=0, sumn=0;
  while (!feof(f))
  {
    fscanf(f,"%s \t%f \t%i\n",&nazva,&price,&kol); // Зчитати рядок з файлу
    if (kol<30)
    { n++;
      sumn += kol;
      for (int i=strlen(nazva); i<=15; i++)
        strcat(nazva," ");
      printf("%s \t%6.2f \t%i\n", nazva, price, kol);
    }
  }
  setlocale(0,".1251");
  cout<<"\nВсього товарів, кількість яких є менше за 30, - "<< n
      <<", їхня кількість - "<< sumn << endl;
  fclose(f);        // Закрити файл
  system("pause");
  return 0;
}
```

Результати роботи програми:

```
Будете вводити рядки для записування у файл? y/n
y
Введіть рядок для записування у текстовий файл у форматі:
 назва товару, ціна, кількість на складі
 Календар       45,60   40
Будете вводити рядки для записування у файл? y/n
n

Вміст файлу:
Пензлики                12,56  70
Кнопки                  15,60  20
Маркер                   7,84  67
Пенал                   70,99  15
Зошит                   17,45  50
Клей                    23,10  10
Календар                45,60  40
Розмір файлу - 138

Кнопки                  15,60  20
Пенал                   70,99  15
Клей                    23,10  10

Всього товарів, кількість яких є менше за 30, - 3, їхня кількість - 45
```

Питання та завдання для самоконтролю

1) Який файл називають текстовим?
2) У чому полягає відмінність між бінарними і текстовими файлами?
3) В якому заголовному файлі означено тип `FILE*`?
4) Що відбувається при відкриванні файлу функцією `fopen()`?
5) Назвати режими відкривання файлів у стилі С.
6) Записати С-функцію створення текстового файлу.
7) Назвати функції файлового форматованого записування-зчитування даних у стилі С.

Лабораторне завдання

1) У протоколі лабораторної роботи дати відповіді на контрольні запитання.

2) У протоколі лабораторної роботи написати мовою С++ програмний код для створення текстового файлу з ім’ям Вашого прізвища і розширенням `rtf`, заповнення файлу даними (бажано не абракадаброю), переглядання вмісту сформованого файлу та розв’язання індивідуальних завдань, які вибрати з табл. 14.3 … 14.4.

3) Створити на комп’ютері програмні проєкти мовою С++ для реалізації написаних програм.

Таблиця 14.3

Індивідуальні завдання середнього рівня складності

№ вар.	Завдання
1	Обчислити кількість рядків текстового файлу, довжина яких більша за 20
2	Вивести на екран рядки текстового файлу, які не містять круглі дужки
3	Обчислити кількість рядків, які починаються і закінчуються на одну й ту саму літеру
4	Вивести на екран рядки текстового файлу, які не містять розділові знаки
5	Обчислити кількість рядків текстового файлу, які містять дефіс
6	Обчислити кількість розділових знаків у текстовому файлі
7	Вивести на екран рядки текстового файлу, які не містять великих літер
8	Обчислити кількість цифр у текстовому файлі
9	Вивести на екран рядки текстового файлу, які мають парну довжину
10	Обчислити кількість рядків текстового файлу, які не містять цифр
11	Вивести на екран рядки текстового файлу, в яких є слово "C++"
12	Обчислити середню довжину рядків текстового файлу
13	Вивести на екран найдовший рядок текстового файлу
14	Обчислити кількість рядків, які містять лише одне слово
15	Вивести на екран найкоротший рядок текстового файлу
16	Обчислити кількість рядків файлу, які починаються з великої літери
17	Вивести на екран рядки текстового файлу, які містять знак оклику
18	Обчислити кількість рядків, які закінчуються не на розділовий знак
19	Вивести на екран рядки текстового файлу, які не містять крапок
20	Вивести на екран всі рядки текстового файлу, видаливши з них зайві пробіли: початкові, кінцеві і подвійні
21	Вивести на екран рядки текстового файлу, які містять цифри
22	Обчислити кількість рядків у текстовому файлі, які закінчуються крапками
23	Вивести на екран всі рядки, переставивши символи у зворотному напряму
24	Вивести на екран всі рядки, перетворивши їхні літери до верхнього регістру
25	Вивести на екран рядки текстового файлу, які не містять літер ‘a’
26	Обчислити найбільшу кількість пробілів у рядках текстового файлу
27	Вивести на екран номера рядків, які містять літеру ‘R’
28	Обчислити кількість рядків у текстовому файлі з непарною довжиною
29	Вивести на екран рядки текстового файлу, які містять рівно 3 слова
30	Обчислити кількість рядків текстового файлу, які починаються з літери ‘A’

Таблиця 14.4

Індивідуальні завдання підвищеного рівня складності

№ вар.	Завдання
1	Вивести всі слова, які починаються з літери ‘a’ чи ‘A’, обчислити їх кількість
2	Вивести всі слова файлу, довжина яких понад 5, та обчислити їх кількість
3	Вивести слова, які розпочинаються й закінчуються на одну й ту саму літеру
4	Вивести найдовше слово у текстовому файлі та його довжину
5	Вивести з кожного рядка текстового файлу слово з найменшою довжиною
6	Вивести слова файлу, довжина яких менше 6 символів, та їх кількість
7	Вивести слова текстового файлу, які містять літеру ‘z’, та їх кількість
8	Вивести слова, які розпочинаються з великої літери, та визначити їх кількість
9	Вивести слова текстового файлу, після яких є кома, та визначити їх кількість
10	Вивести слова файлу, які розпочинаються з малої літери, та їх кількість
11	Вивести слова текстового файлу з парною довжиною та їх кількість
12	Вивести слова текстового файлу, які містять цифри, та їх кількість
13	Вивести з кожного рядка текстового файлу слово з найбільшою довжиною
14	Вивести найкоротше слово у текстовому файлі та його довжину
15	Вивести рядок текстового файлу, який містить найдовше слово
16	Вивести слова текстового файлу, які не містять літеру ‘o’, та їх кількість
17	Вивести слова файлу, які містять лише великі літери, та їх кількість
18	Вивести слова файлу, які містять лише латинські літери, та їх кількість
19	Обчислити середню довжину слів у текстовому файлі
20	Вивести рядок текстового файлу, який містить найкоротше слово
21	Вивести слова файлу з довжиною більше семи символів та їх кількість
22	Вивести слова файлу, які містять щонайменше дві літери ‘a’, та їх кількість
23	Обчислити середню довжину слів у перших трьох рядках текстового файлу
24	Вивести слова текстового файлу, які починаються на голосну літеру, та обчислити їх кількість
25	Вивести слова текстового файлу з непарною довжиною та їх кількість
26	Вивести слова текстового файлу, довжина яких кратна 3, та їх кількість
27	Вивести рядки текстового файлу, які містять слова лише з великих літер
28	Вивести рядки текстового файлу, які містять слова лише з латинських літер
29	Обчислити кількість слів текстового файлу, які містять лише літери кирилиці
30	Вивести слова текстового файлу, які закінчуються на голосну літеру, та обчислити їх кількість

Самостійна робота № 13

Опрацювання текстових файлів з числовими даними

Мета роботи: закріпити практичні навички програмного створення та редагування текстових файлів засобами С++.

Приклад програми

Приклад 1. Створити за допомогою програми *Блокнот* текстовий файл та заповнити його числовими даними у вигляді матриці розмірності 3×4. Написати мовою С++ програмний код для формування із даних текстового файлу числової матриці та розв'язання таких завдань:

– створити вектор середньоарифметичних максимуму і мінімуму стовпців;
– обчислити суму додатних елементів головної діагоналі матриці

Програмно записати результат обчислень у той самий файл.

Текст програмного коду:

```
#include <iostream>
#include <stdlib.h>
using namespace std;

const int M = 3, N = 5;
// Перегляд файлу
void view_file(char* name)
{
     char s[100];   FILE* f;
     f = fopen(name, "rt"); // відкрити файл для читання як текстовий
     if (f==NULL) { cout<<"Cannot open file to veiw\n"; return; }
     cout << "\nView file " << name << endl;
     while (fgets(s, 100, f))
     { // Зчитувати послідовно записи з файлу, допоки вони є у файлі
       puts(s); // Вивести рядок на екран
     }
     fclose(f);
}

// Заповнити матрицю значеннями із файлу
void create_matrix(char* name, double matr[M][N])
{
     FILE* f;
     char s[100], *t;
     int i = 0, j;
```

```
	f = fopen(name, "rt"); // Відкрити файл для читання як текстовий
	if (f==NULL) { cout<<"Cannot open file to veiw\n"; return; }
	while (fgets(s, 100, f)>0)
	{
		j = 0;
		t = strtok(s, " \t");
		while (t != NULL)
		{
			matr[i][j] = atof(t);
			t = strtok(NULL, " \t");
			j++;
		}
		i++;
	}
	fclose(f);
}
// Вивести матрицю на екран
void output_matrix(double matr[M][N])
{
	int i, j;
	cout << "\nView matrix:\n";
	for (i = 0; i < M; i++)
	{
		for (j = 0; j < N; j++)
			cout << matr[i][j] << "\t";
		cout << endl;
	}
}
//Створити вектор середньоарифметичних максимуму і мінімуму стовпців
void create_vector(double matr[M][N], double vekt[N])
{
	int i, j;
	double max, min;
	for (j = 0; j < N; j++)
	{
		max = matr[j][0];
		min = matr[j][0];
		for (i = 0; i < M; i++)
		{
			if (matr[i][j] > max)
				max = matr[i][j];
			if (matr[i][j] < min)
				min = matr[i][j];
		}
		vekt[j] = (max + min) / 2;
	}
}
```

```
// Вивести вектор на екран та у файл
void output_vector(double vekt[N])
{
     int i;
     cout << "\nView vector:\n";
     for (i = 0; i < N; i++)
          cout << vekt[i] << "\t";
     cout << endl;
}
// Сума додатних елементів головної діагоналі матриці
double sum_diagonal(double matr[M][N])
{
     int i;
     double sum = 0;
     for (i = 0; i < M; i++)
          sum += matr[i][i];
     return sum;
}
int main()
{
  double A[M][N], B[N];   int i, j;
  char name[] = "D:\\F1.txt";
  FILE * f;
  // Перегляд файлу
  view_file(name);
  // Заповнити матрицю числами із файлу
  create_matrix(name, A);
  // Вивести матрицю на екран та у файл
  output_matrix(A);
  // Створити вектор середньоарифметичних максимуму і мінімуму стовпців
  create_vector(A, B);
  // Вивести вектор на екран та у файл
  output_vector(B);
  // Сума елементів головної діагоналі матриці
  double s = sum_diagonal(A);
  cout << "\nSum of the main diagonal = " << s << endl;
  // Запис результатів обчислень у файл
  f = fopen(name, "at"); // Відкрити файл для читання як текстовий
  if (f==NULL) { cout<<"Cannot open file to append data\n"; return 1; }
  fprintf(f, "\n\nСтворений вектор:");
  for (int i=0; i<N; i++)
    fprintf(f, "%7.1f", B[i]);
  fprintf(f, "\n\nSum = %7.1f\n", s);
  fclose(f);
  system("pause");
  return 0;
}
```

Результати виконання програми:

```
View file D:\F1.txt
5 8 -2 1 6

-12 -1 0 0 1

2 9 -1 0 2

View matrix:
5       8       -2      1       6
-12     -1      0       0       1
2       9       -1      0       2

View vector:
-3.5    -1.5    0       0.5     3

Sum of the main diagonal = 3
```

F1.txt – Блокнот

Файл Правка Формат Вид Справка

```
5 8 -2 1 6
-12 -1 0 0 1
2 9 -1 0 2

Створений вектор:   -3.5   -1.5    0.0    0.5    3.0

Sum =      3.0
```

Питання та завдання для самоконтролю

1) Як працюватиме умова у циклі `while(fgets(s, 100, f)>0)`…?
2) Чим відрізняється дія попередньої команди від такої:
   `while (!feof(f)) fgets(s, 100, f) ;`
3) Яке призначення функцій `fscanf()` та `fgets()`? Чи вони схожі і чим відрізняються?
4) Записати С-функцію для відкривання текстового файлу для дописування даних у кінець файлу.
5) Записати С-функцію для відкривання текстового файлу для читання даних.
6) Записати С-функцію для відкривання чи створення текстового файлу для дописування даних.

Завдання до самостійної роботи

1) Дати відповіді на контрольні запитання.

2) Створити за допомогою програми *Блокнот* текстовий файл та заповнити його числовими даними у вигляді матриці.

3) Написати мовою С++ програмний код для формування із даних текстового файлу числової матриці та розв'язання індивідуальних завдань, які вибрати з табл. 13.3 … 13.4. Програмно записати результат обчислень у той самий файл.

4) Створити на комп'ютері програмні проєкти мовою С++ для реалізації написаних програм.

Таблиця 13.3

Індивідуальні завдання середнього рівня складності

№ вар.	Тип елементів	Завдання для обчислення
1	double	Обчислити кількість від'ємних елементів у перших трьох рядках матриці
2	double	Знайти суму модулів від'ємних елементів матриці
3	double	Обчислити елементи вектора як суми елементів парних рядків матриці
4	int	Обчислити вектор як суми парних елементів стовпців матриці
5	double	Обчислити добуток елементів, які розташовані над головною діагоналлю матриці
6	double	Обчислити елементи вектора як добутки елементів стовпців, абсолютне значення яких не перевищує 5
7	double	Знайти суму модулів елементів, розташованих під головною діагоналлю
8	double	Обчислити елементи вектора як квадрати елементів неголовної діагоналі матриці
9	int	Обчислити елементи вектора як рядок матриці з максимальним елементом
10	double	Обчислити елементи вектора як кількості додатних елементів рядків матриці
11	double	Обчислити значення максимального елемента головної діагоналі
12	double	Обчислити елементи вектора як найбільші за модулем елементи рядків
13	double	Обчислити кількість елементів зі значенням, більшим за середнє арифметичне елементів матриці
14	int	Обчислити вектор як рядок матриці з мінімальним елементом
15	double	Обчислити кількість від'ємних елементів у непарних рядках матриці
16	double	Обчислити значення середньоарифметичного елементів головної діагоналі матриці
17	double	Обчислити елементи вектора як середнє арифметичне у кожному рядку матриці

Закінчення табл. 13.3

№ вар.	Тип елементів	Завдання для обчислення
18	int	Обчислити вектор як модуль сум непарних елементів рядків матриці
19	double	Обчислити елементи вектора як стовпець матриці з максимальним елементом
20	double	Обчислити елементи вектора як суми елементів стовпців матриці, значення яких більше за 10
21	int	Обчислити значення добутку елементів неголовної діагоналі матриці
22	double	Обчислити елементи вектора як максимальні елементи рядків
23	double	Обчислити суму додатних елементів в останніх двох стовпцях матриці
24	double	Обчислити елементи вектора як середнє арифметичне стовпців матриці
25	double	Обчислити елементи вектора як квадрати сум від'ємних елементів кожного стовпця матриці
26	double	Обчислити кількість елементів матриці, модуль яких менше 6
27	double	Обчислити елементи вектора як різницю максимального та мінімального елементів рядків матриці
28	int	Обчислити значення мінімального додатного елемента матриці
29	double	Обчислити значення найменшого елемента матриці
30	int	Обчислити елементи вектора як середнє арифметичне парних елементів рядків матриці

Таблиця 13.4

Індивідуальні завдання підвищеної складності

№ вар.	Тип елементів	Завдання для обчислення
1	double	Визначити кількість від'ємних елементів тих рядків, які містять хоча б один нульовий елемент
2	double	Визначити мінімум серед сум елементів діагоналей, паралельних головній діагоналі матриці
3	double	Визначити номер стовпця, в якому розташована щонайдовша серія однакових елементів
4	int	Визначити кількість стовпців, що не мають жодного нульового елемента
5	double	Визначити кількість елементів у тих рядках, які не містять від'ємних елементів
6	double	Визначити суму елементів у тих рядках, які містять хоча б один від'ємний елемент
7	double	Визначити добуток стовпців, що містять хоча б один елемент, більший за 100
8	double	Визначити кількість рядків, що містять принаймні один нульовий елемент

Закінчення табл. 13.4

№ вар.	Тип елементів	Завдання для обчислення
9	double	Визначити суму елементів у тих рядках, які не містять від'ємних елементів
10	int	Характеристикою стовпця матриці назвемо суму модулів його від'ємних непарних елементів. Переставити стовпці матриці у порядку спадання характеристик
11	double	Визначити номер першого зі стовпців, що містять хоча б один нульовий елемент
12	double	Знайти максимальне з чисел, що зустрічаються у матриці більше одного разу
13	double	Знайти мінімум серед сум модулів елементів діагоналей, паралельних побічній діагоналі матриці
14	int	Визначити кількість рядків, що не містять жодного парного елемента
15	double	Характеристикою стовпця матриці назвемо квадрат суми його елементів. Переставити стовпці матриці за спаданням їхніх характеристик
16	double	Визначити суму елементів тих стовпців, які не містять від'ємних елементів
17	double	Знайти кількість стовпців, які не містять від’ємних елементів
18	double	Знайти номер рядка з найдовшою серією однакових елементів
19	double	Знайти суму елементів матриці, які зустрічаються понад двічі
20	double	Обчислити елементи вектора як скалярний добуток елементів рядків матриці на головну діагональ
21	int	Характеристикою рядка матриці назвемо суму її додатних парних елементів. Переставити рядки матриці за зростанням характеристик
22	double	Перестановкою рядків упорядкувати за спаданням елементи першого стовпця матриці
23	int	Обчислити суму елементів рядка матриці, який не містить чисел, кратних 10
24	double	Упорядкувати рядки матриці за зростанням кількості однакових елементів у кожному рядку
25	int	Характеристикою рядка матриці назвемо суму її від'ємних парних елементів. Переставити рядки матриці за спаданням характеристик
26	double	Обчислити елементи вектора як скалярний добуток елементів рядків матриці на останній стовпець
27	double	Знайти номер першого з рядків, що не містить жодного додатного елемента
28	double	Обчислити елементи вектора як скалярний добуток елементів стовпців матриці на перший рядок
29	double	Ущільнити задану матрицю, видаляючи з неї рядки і стовпці, заповнені нулями
30	int	Знайти номер першого зі стовпців, що не містить жодного від'ємного елемента

Лабораторна робота № 15

Програмування з використанням структур (struct)

Мета роботи: набути практичних навиків організації структур і роботи з їхніми полями мовою С++.

Теоретичні відомості

1. Поняття структури

Структура – це тип даних, який може поєднувати у собі різнотипні елементи. Елементи структури називаються *полями структури* і можуть мати будь-який тип, крім типу тієї самої структури, але можуть бути вказівником на нього.

Опис структури має синтаксис:

```
struct  <ім'я_типу_структури>
{ <тип1> <поле1>;
  <тип2> <поле2>;
   . . .
  <типN> <полеN>;
};
```

Наприклад, структуру `worker`, яка має чотири поля: прізвище співробітника (поле `name`), посада (поле `position`), вік (поле `age`) та зарплатня (поле `salary`), можна описати так:

```
struct worker
{ char name[15], position[10];
  int age; double salary;
};
```

Такий запис не виділяє пам'ять, а лише створює новий тип даних `worker`, ім'я якого може використовуватись при оголошуванні змінних. Наприклад, оголошення змінної `z`, масиву структур `Mas` та вказівника на структуру `ps`:

```
worker z, Mas[3], *ps;
```

При оголошуванні змінної типу структури виділяється пам'ять під усі поля структури послідовно для кожного поля. У наведеному прикладі структури `worker` під змінну `z` послідовно буде виділено 15, 10, 4, 8 байтів. Однак, розмір структури не завжди дорівнює сумі розмірів її полів, оскільки внаслідок вирівнювання об'єктів різної довжини у структурі можуть з'являтися безіменні "дірки". Правильне значення розміру допоможе визначити операція `sizeof`.

При оголошуванні структур їхні поля можна ініціалізовувати початковими значеннями, наприклад:

```
worker Mas[3] = {"Брончук","менеджер", 20, 2575.3,
                 "Ханін","програміст", 27, 34567,
                 "Яшкіна","бухгалтер", 50, 2699.99, };
```

2. Доступ до полів структури

Доступ до полів структури здійснюється за допомогою операцій вибору: операції "." (крапка) при звертанні до полів через ім'я структури й операції "->" при звертанні за вказівником, наприклад для вище оголошених змінних:

```
strcpy(z.name,"Бойко");
z.age = 34;
Mass[0].salary = 3750.5;
ps->salary = 4000;
```

До речі, для вказівника попередньо слід задати об'єкт посилання командою `ps = &z`, тобто записати адресу конкретної змінної типу структури.

Якщо елементом структури є інша структура, то доступ до її полів здійснюють через дві операції вибору:

```
struct A {int a; double x;};
struct B {A a; double x;} x[2];
x[0].a.a = 1;
x[1].x = 0.1;
```

Як видно з прикладу, поля різних структур можуть мати однакові імена, оскільки в них різна область видимості, головне при цьому не заплутатись.

Якщо полем структури є функція, то виклик цієї функції відбувається при звертанні до неї як до будь-якого іншого поля цієї структури. Наприклад, в структурі `student` одним з полів є функція `Show()`, яка будує рядок класу `String`, поєднуючи всі поля цієї структури для подальшого зручного виведення:

```
ref struct student
{ String ^prizv, ^gr;
  int math, inf;
  String^ Show()
   { return prizv+"  "+gr+"  "+math.ToString()+"  "+inf.ToString();
   }
};
```

В основній програмі цю функцію можна викликати в такий спосіб:

```
student z;
z.prizv = "Іванов";  z.gr = "ПІ-11";  z.math = 60;  z.inf = 90;
richTextBox1->Text = z.Show();
```

3. Дії над структурами

1) Для змінних одного й того самого структурного типу визначена *операція присвоювання*, при виконанні якої відбувається поелементне копіювання.

```
worker a, b = {"Хан","менеджер", 23, 1750};
a = b;
```

Команда `a = b` скопіює відразу всі чотири поля структури `b` в поля структури `a`.

```
struct  tovar
{ char  nazva[12],  virobnik[10];  int col; float vart; } ;
```

```
tovar x, y;
scanf("%s %s %i %g", x.nazva, x.virobnik, &x.col, &x.vart);
y = x;
printf("%s %s %i %5.2f\n", y.nazva, y.virobnik, y.col, y.vart);
```

2) Структуру можна передавати до функції та повертати в якості результату роботи функції.

```
Show(worker z)
{ cout<<z.name<<"  "<<z.position<<"  "<<z.age<<"  "<<z.salary<<endl;
}
worker Put()
{ worker w;
  cout<< "Введіть рядок з даними про робітника \n
         Прізвище  Посада Вік  Зарплатня" << endl;
  return cin >> w.name >> w.position >> w.age >> w.salary;
}
void main ()
{ date a, b = {"Лі","бухгалтер", 33, 2700};
  Show(b);   // виклик функції з передаванням структури b до функції
  a = Put(); // отримання структури в якості результату виконання функції
  Show(a);
  . . . . .
```

Приклади програм зі структурами

Приклад 1. Створити програму для опрацювання результатів сесії студентів: прізвище студента, група і результати двох екзаменів. У програмі передбачити можливість введення і виведення даних про довільну кількість студентів та відбір даних про студентів, які успішно здали сесію й вийшли на стипендію (не мають незадовільних оцінок, а середній бал склав понад 75 балів), також визначити прізвище студента з найбільшим середнім балом.

Розв'язок. У наведеному нижче програмному коді для одночасного введення всіх даних (чотирьох полів структури: прізвище, група і результати двох екзаменів) про одного студента використано функцію `scanf`, а виведення таких даних організовано за допомогою функції `printf`.

Ще однією особливістю цього програмного коду є почергове використання різних кодувань (команди `setlocale(0,".1251")` і `setlocale(0,".OCP")`) для коректного виведення літер кирилиці в консолі. Без використання цих команд кирилиця виводитиметься у вигляді абракадабри. Специфіка полягає у тому, що C++ по-різному інтерпретує коди введених символьних даних і набраного у програмі тексту. Так, для виведення на екран текстових повідомлень у Visual C++ слід використовувати кодування Windows 1251 (команда `setlocale(LC_ALL,".1251")` або `setlocale(0,".1251")`), а для виведення раніш введених символьних даних слід повернутися до початкових налаштувань командою `setlocale(0,".OCP")`.

Текст програмного коду:

```
#include <iostream>
using namespace std;
int main()
{
  setlocale(LC_ALL,".1251");
  struct student
  { char surname[22], gr[8];
    int ex1, ex2;
  } ;
 int kol=0;           // загальна кількість студентів
 cout<<"Введіть кількість  студентів - "; cin>> kol;
 student *z = new student[kol];
 cout<< "Введіть почергово рядки з відомостями про успішність " << kol
     << " студентів: \nПрізвище Група Оцінка1 Оцінка2" << endl;
 for(int i=0; i<kol; i++)
   scanf("%s %s %i %i",z[i].surname,z[i].gr, &z[i].ex1,&z[i].ex2);
 cout<< "\n   Успішність "<<kol
     << " студентів: \nПрізвище        Група Оцінка1 Оцінка2" << endl;
 setlocale(0, ".OCP");
 for(int i=0; i<kol; i++)
    printf("%s\t%s\t%i\t%i\n",z[i].surname,z[i].gr, z[i].ex1,z[i].ex2);
 setlocale(0, ".1251");
 cout<< endl << "Успішно здали сесію та вийшли на стипендію:" << endl;
 int n=0;             // кількість стипендіатів
 double sr,max(0);    // середній бал кожного та його максимальне значення
 char maxname[22];    // прізвище студента з максимальним середнім балом
 for(int i=0; i<kol; i++)
 { if(z[i].ex1>=60 && z[i].ex2>=60 && (z[i].ex1+z[i].ex2)/2.0 >= 75)
   { n++;
     setlocale(0, ".OCP");
     printf("%s\t%s\t%i\t%i\n",z[i].surname,z[i].gr,z[i].ex1,z[i].ex2);
   }
   sr=(z[i].ex1+z[i].ex2)/2.0;
   if(sr>max) { max=sr; strcpy(maxname,z[i].surname); }
 }
 setlocale(0, ".1251");
 cout<< endl << "Кількість стипендіатів - " << n << endl;
 cout<< endl << "Максимальний середній бал "<< max <<" має ";
 setlocale(0, ".OCP"); cout<< maxname << endl << endl;
 delete []z;
 system ("pause>>void");
 return 0;
}
```

Результат виконання програми:

```
Введіть кількість  студентів - 3
Введіть почергово рядки з відомостями про успішність 3 студентів:
Прізвище Група Оцінка1 Оцінка2
Костенко КТ-16  60      85
Халанчук ПІ-01  95      90
Мацкевич КТ-16  45      78

   Успішність 3 студентів:
Прізвище        Група Оцінка1 Оцінка2
Костенко        КТ-16   60      85
Халанчук        ПІ-01   95      90
Мацкевич        КТ-16   45      78

Успішно здали сесію та вийшли на стипендію:
Халанчук        ПІ-01   95      90

Кількість стипендіатів - 1

Максимальний середній бал 92.5 має Халанчук
```

Приклад 2. Створити програму для опрацювання даних телефонної книги (не менше п’яти абонентів): прізвище, ім’я, номер телефону, дата народження. Передбачити можливість виведення всіх даних телефонної книги та відбір даних про абонентів, номери яких розпочинаються з 067 або 068 (абоненти Київстар).

Розв’язок. У наведеному нижче програмному коді дані телефонної книги заповнюються самою програмою під час ініціалізації масиву структур.

Текст програмного коду:

```
#include <iostream>
#include <clocale>
using namespace std;
int main()
{ setlocale(0,".1251");
  struct abonent
  { char surname[21], name[15], tel[15], d[11];
  } z[]={"Харитонова","Іванна","066-99-55-444","11.05.1985",
         "Лавриненко","Петро","093-10-12-500","23.12.87",
         "Кононова","Олена","067-55-10-123","30.12.1990",
         "Петранчук","Олег","067-55-10-123","03.02.56",
         "Холоденко","Тетяна","068-333-11-850","15.08.79" };
 int kol = sizeof(z) / sizeof(abonent);
 cout<<"   Телефонна книга з "<<kol
     <<" абонентами: \nПрізвище  Ім’я  Номер телефона  Дата народження"<<endl;
 for(int i=0; i<kol; i++)
   cout<<z[i].surname<<"\t"<<z[i].name<<"\t"<<z[i].tel<<"\t"<<z[i].d<<endl;
 cout<<endl<<" Абоненти, номери яких розпочинаються з 067 або 068:"<<endl;
 int n=0;
 for (int i=0; i<kol; i++)
  if (!strncmp(z[i].tel,"067",3)||!strncmp(z[i].tel,"068",3))
   { n++;
     cout<<z[i].surname<<"\t"<<z[i].name<<"\t"<<z[i].tel<<"\t"<<z[i].d<<endl;
```

```
   }
 cout<<endl<<"Кількість абонентів Київстар - "<<n<<endl;
 system ("pause>>void");
 return 0;
}
```

Результати роботи:

```
   Телефонна книга з 5 абонентами:
Прізвище        Ім'я    Номер телефону  Дата народження
Харитонова      Іванна  066-99-55-444   11.05.1985
Лавриненко      Петро   093-10-12-500   23.12.87
Кононова        Олена   067-55-10-123   30.12.1990
Петранчук       Олег    067-55-10-123   03.02.56
Холоденко       Тетяна  068-333-11-850  15.08.79

 Абоненти, номера яких розпочинаються з 067 чи 068:
Кононова        Олена   067-55-10-123   30.12.1990
Петранчук       Олег    067-55-10-123   03.02.56
Холоденко       Тетяна  068-333-11-850  15.08.79

Кількість абонентів Київстар - 3
```

Приклад 3. Ввести дані про товари: назва, фірма-виробник, кількість і ціна. Визначити сумарну вартість кожного з товарів фірми LG та середню вартість таких товарів.

Текст програмного коду:

```
#include <clocale>
#include <iostream>
using namespace std;

int main()
{
  struct  tovar
  { char  nazva[12], pr[10];  int cnt;  float vart; };
  int  kol = 0;
  setlocale( LC_ALL, ".1251" );
  cout << "Введіть кількість товарів -  ";  cin >>  kol;
  tovar *z =  new  tovar[kol];
  cout << "Введіть почергово рядки з відомостями про " << kol
  << " товари(ів):\nНайменування Виробник Кіл-ть Ціна (десяткова кома)"<<endl;
  setlocale( LC_ALL, ".OCP" );
  for (int i=0; i<kol; i++)
    scanf("%s %s %i %g",z[i].nazva,z[i].pr,&z[i].cnt,&z[i].vart);
  double allsum=0, Svart, skol=0; int ktel=0;
  setlocale(LC_ALL, ".1251");
  cout << "\nПерегляд товарів фірми  LG:
           \n№№ Найменування Виробник  Кіл-ть  Ціна Вартість" << endl;
  setlocale(LC_ALL, ".OCP");
  for (int  i=0; i<kol; i++)
   if (!stricmp(z[i].pr,"LG"))
   { ktel++;
```

```
      skol + =z[i].cnt;
      Svart = z[i].vart * z[i].cnt;
      allsum += Svart;
      printf("%i  %s\t%s\t%i\t%5.2f\t%6.2f\n",
             ktel, z[i].nazva, z[i].pr, z[i].cnt, z[i].vart, Svart);
    }
  allsum /= skol;
  setlocale(LC_ALL, ".1251");
  printf("\nСередня вартість товарів фірми LG - %5.2f\n", allsum);
  delete []z;
  system ("pause>>void");
  return 0;
}
```

Результати роботи:

```
Введіть кількість товарів -  3
Введіть почергово рядки з відомостями про 3 товари(ів):
Найменування Виробник  Кіл-ть  Ціна (десяткова кома)
Телефон         LG       5       599,50
Телевізор      Sony      3      4199,99
Планшет         LG      10      2400,00

Перегляд товарів фірми  LG:
№№ Найменування Виробник  Кіл-ть  Ціна Вартість
1  Телефон      LG       5       599,50  2997,50
2  Планшет      LG       10      2400,00 24000,00

Середня вартість товарів фірми LG - 1799,83
```

Питання та завдання для самоконтролю

1) Дати визначення структури як типу даних.
2) Як здійснюється доступ до полів структури?
3) Як визначається обсяг пам’яті, потрібний для зберігання структури?
4) Чи можуть в одній програмі збігатися імена полів структури і змінних?
5) Навести оголошення структури, яка описуватиме для деякого електричного приладу такі характеристики: назва приладу, споживана потужність і номінальна напруга.

Лабораторне завдання

1) У протоколі лабораторної роботи дати відповіді на контрольні запитання.
2) У протоколі лабораторної роботи написати мовою С++ програмний код для формування 5 – 10 даних типу структури, поля якої подано в табл. 15.1 відповідно до індивідуального варіанта.
3) Створити на комп’ютері програмні проєкти мовою С++ для реалізації написаних програм.

Таблиця 15.1

Варіанти завдань для роботи зі структурами

№ вар.	Поля структури	Індивідуальні завдання
1	*Сесійні дані про студентів:* – прізвище, – група, – фізика – інформатика – історія	Визначити середній бал оцінок кожного студента і відібрати студентів, середній бал яких більше 75
2		Визначити середній бал оцінок усіх студентів з фізики і відібрати студентів, які склали екзамен з інформатики на "відмінно" (≥90)
3		Визначити кількість студентів, які не склали екзамен з інформатики, та визначити їх середній бал
4		Відібрати студентів-відмінників та визначити їхню кількість
5		Відібрати студентів, які здали сесію, але не "дотягують" до стипендії, та обчислити відсотковий вміст таких студентів серед решти представлених студентів
6		Відібрати студентів, які мають незадовільну оцінку хоча б з одного екзамену, та визначити їхню кількість
7	*Дані про працівників:* – прізвище, – посада, – освіта – рік народження, – зарплатня	Визначити працівників з найбільшою та найменшою зарплатнею і визначити їх сумарну зарплатню
8		Відібрати працівників, яким до пенсії (до 6-ти років) лишилося менше 3-х років, та обчислити їх відсотковий вміст серед решти працівників
9		Визначити наймолодшого і найстаршого працівників
10		Визначити працівників, які в поточному році святкують ювілей – вік кратний 5 чи 10-ти
11		Відібрати працівників молодших 30-ти років та обчислити їхню кількість
12		Відібрати працівників, зарплатня яких більше середнього значення зарплатні всіх працівників

Закінчення табл. 15.1

№ вар.	Поля структури	Індивідуальні завдання
13	*Товари на складі*: – найменування, – виробник, – ціна, – кількість	Визначити найдорожчий товар на складі, вивести всі дані про нього та обчислити його сумарну вартість
14		Обчислити загальну кількість та середню ціну товарів
15		Обчислити сумарну вартість кожного товару та сумарну вартість усіх товарів на складі
16		Відібрати товари, кількість яких менше 10-ти, обчислити кількість найменувань і сумарну кількість таких товарів
17		Визначити товар з найбільшою загальною вартістю на складі
18		Визначити товар з найбільшою кількістю на складі та обчислити його сумарну вартість
19	*Характеристики вебсайту:* – повне доменне ім’я; – категорія (особистий, корпоративний, інформаційний, промосайт, блог, Інтернет-магазин, соціальна мережа тощо); – кількість відвідувань в день; – рейтинг PR (Page Rank – число від 0 до 10)	Відібрати сайти з рейтингом понад 5 та обчислити їхню кількість серед представлених
20		Відібрати сайти на доменах "ua" та "com", обчислити їхню кількість
21		Обчислити середній рейтинг та середню кількість відвідувань за день усіх представлених сайтів
22		Відібрати сайти з категорії "blog" та обчислити їхню кількість серед представлених
23		Відібрати сайти на домені "ua" з рейтингом PR понад 7 та обчислити їхню кількість серед представлених
24		Відібрати сайти з кількістю відвідувань понад 50 та обчислити кількість таких сайтів серед представлених
25		Визначити сайт з найбільшим рейтингом
26		Відібрати сайти на домені "net" або "ru" та обчислити їхню кількість
27	*Література у видавництві*: – автор; – назва книги; – рік видання; – тираж; – кількість сторінок	Відібрати книги з тиражом до 1000 екземплярів та обчислити кількість таких книг серед представлених
28		Відібрати книги, видані у поточному році та обчислити кількість таких книг
29		Визначити найтовстішу книгу (максимальна кількість сторінок)
30		Відібрати книги з кількістю сторінок понад 150 та обчислити кількість таких книг

Самостійна робота № 14

Опрацювання даних типу "дата-час" у текстових файлах

Мета роботи: набути практичних навиків програмного опрацювання даних типу "дата-час".

Теоретичні відомості

Тип «дата-час» у C++

Календарний час містить в собі рік, місяць, день місяця, години, хвилини, секунди. Функції й типи даних, потрібні для опрацювання дати і часу, оголошено у заголовному файлі `time.h`. Зокрема цей файл містить визначення типів даних `time_t`:

```
typedef long time_t;
typedef long clock_t;
```

і структури `tm`, оголошеної в такий спосіб:

```
struct tm
{ int   tm_sec;    // секунди 0..60
  int   tm_min;    // хвилини 0..59
  int   tm_hour;   // години 0..23
  int   tm_mday;   // день місяця 1..31
  int   tm_mon;    // місяць 0..11 (0=січень)
  int   tm_year;   // кількість років від 1900 р.
  int   tm_wday;   // день тижня 0..6 (0=неділя)
  int   tm_yday;   // день року 0..365
  int   tm_isdst;  // літній час (булеве значення)
};
```

Час у форматі `time_t` по суті є числом – кількістю секунд, рахуючи від 00:00 годин 1 січня 1970 р. (unix timestamp).

Час у форматі `struct tm` подано структурою з полями: секунди, хвилини, години, день, місяць, рік тощо. Звертаємо увагу, що рік рахується від 1900, тобто 2019 р. має зберігатися як 119, а місяць рахується від 0, отже квітень це 3.

Основні функції для опрацювання дати і часу з описом їхньої роботи наведено у табл. 14.5.

Таблиця 14.5

Функції опрацювання дати і часу

Синтаксис функції	Опис роботи функції
char *asctime (struct tm time);	Перетворює дату і час time з формату структури типу tm на символьний рядок з 26-ти символів, який може мати вигляд Mon Jan 20 02:03:55 2019\n\0 Функція повертає вказівник на цей рядок
clock_t clock ();	Повертає процесорний час, який сплинув від початку виконання програми, чи –1, якщо він є невідомий
char *ctime (long *time);	Перетворює час time формату long* у символьний рядок з 26 символів, який може вигляд: Mon Jan 20 02:03:55 2019\n\0 Функція повертає вказівник на цей рядок
double difftime (time_t time2, time_t time1);	Повертає різницю поміж time2 і time1 типу time_t, тобто час у секундах, який сплине від time2 до time1, як число з подвійною точністю
struct tm *gmtime (long *time);	Перетворює time типу long* до формату структури tm
struct tm *localtime (long *time);	Перетворює time типу long* до формату структури tm з урахуванням зони місцевого часу, якщо попередньо було задано глобальну змінну часової зони TZ
time_t mktime (struct tm *time);	Перетворює час і дату time формату структури tm до формату time_t
char * _strdate (char *date);	Перетворює поточну дату на символьний рядок time з 9 символів формату MM/DD/YY
char * _strtime (char *time);	Перетворює поточний час на символьний рядок time з 9 символів формату HH:MM:SS
size_t strftime (char *s, size_t maxsize, char *fmt, struct tm *t);	Перетворює дату і час t формату структури tm на символьний рядок s розміром maxsize у форматі fmt
time_t time (time_t *timer);	Повертає поточний календарний час (тобто час в секундах, який сплинув з півночі (00:00:00) 1 Січня 1970 за Грінвічем)
void tzset ();	Функція встановлює значення глобальних змінних daylight (ненульове значення типу int означає дозвіл переходу на літній час), timezone (значення типу long задає різницю в секундах поміж місцевим часом і часом за Грінвічем (GMT)) і tzname (рядок типу *char трилітерних часових зон)

Розглянемо докладніше специфікатори формату для функції strftime, яка перетворює дату і час t формату структури tm на символьний рядок s розміром maxsize у форматі fmt і зазвичай використовується для виведення дати,

поданої у форматі `struct tm`. Ця функція має чотири параметри: `s, maxsize, fmt, t`. До рядка `s` записується результат виконання функції. Число `maxsize` задає максимальну довжину рядка-результату. Формат `fmt` – це рядок, який містить будь-яку комбінацію специфікаторів, які починаються зі знаку `%`. Функція перетворить ці специфікатори на відповідні рядкові значення. Деякі зі специфікаторів формату, які використовуються у функції `strftime`, подано у табл. 14.6.

Таблиця 14.6

Специфікатори формату для функції `strftime`

Специфікатор	Значення специфікатора	Приклад
%a	Скорочена назва дня тижня*	Thu
%A	Повна назва дня тижня*	Thursday
%b	Скорочена назва місяця*	Aug
%B	Повна назва місяця*	August
%c	Дата і час*	08/20/19 14:55:02
%d	День місяця у вигляді двоцифрового числа (01-31)	20
%D	Скорочена дата у форматі MM/DD/YY, еквівалентна до %m/%d/%y	08/20/19
%e	День місяця, в одноцифрових числах перший знак замінено на пробіл (1-31)	20
%H	Години у 24-годинному форматі (00-23)	14
%I	Години у 12-годинному форматі (01-12)	02
%m	Місяць у числовому вигляді (01-12)	08
%M	Хвилини (00-59)	55
%p	AM або PM	PM
%r	Час у 12-годинному форматі*	02:55:02 pm
%R	Час у 24-годинному форматі HH:MM, еквівалентно до %H:%M	14:55
%S	Секунди (00-61)	02
%w	Номер дня тижня, неділя = 0 (0-6)	4
%W	Номер тижня з першим понеділком як першим днем тижня (00-53)	34
%x	Дата*	08/23/01
%X	Час*	14:55:02
%y	Останні дві цифри року (00-99)	19
%Y	Рік	2019

* Специфікатори, позначені зірочкою (*), залежать від локалізації.

Приклади програм

Приклад 1. Вивести поточну системну дату і час.

Текст програми:

```
#include <iostream>
#include <time.h>
using namespace std;
int main ()
{ time_t t;          // Оголошення змінної t
  t = time(NULL);    // Запис поточного часу до t
  printf("%s\n", ctime(&t));
  system("pause");
  return 0;
}
```

Результат виконання програми:

```
Wed Dec 11 16:33:16 2019
```

Приклад 2. Проілюструємо роботу деяких з функцій, наведених у табл. 14.5, на прикладі консольної програми для виведення поточної дати і часу у заданому форматі. Час виводитиметься у 12-годинному форматі.

Текст програми:

```
#include <iostream>
#include <time.h>
using namespace std;
int main()
{ struct tm *newtime;   char *am_pm="PM";
  time_t long_time; // Оголошення змінної long_time.
  time(&long_time); // Запис поточного часу до long_time.
  // Перетворення long_time на структуру newtime.
  newtime=localtime (&long_time);
  if (newtime->tm_hour<12) // Якщо проминуло менше за 12 годин доби,
    am_pm="AM";   // відбудеться відповідне визначення рядку формату.
  if (newtime->tm_hour>12) // Якщо проминуло більше за 12 годин доби,
    newtime->tm_hour-=12;  // відбудеться віднімання 12.
  printf("%.20s %s\n", asctime (newtime), am_pm);
  cin.get();
  return 0;
}
```

Результати виконання програми:

```
Wed Dec 11 08:17:55 PM
```

Приклад 3. Проілюструємо роботу функції `strftime`, яка перетворює введені значення дати і часу (формату структури `tm`) на символьний рядок у заданому форматі день/місяць/рік.

Текст програми:

```
#include <iostream>
#include <time.h>
using namespace std;
```

```
int main ()
{ setlocale(0,".1251");
  time_t t;
  struct tm tmstr;
  int day, month, year;
  char str[256];
  cout<<"Введіть день: "; cin>>day; tmstr.tm_mday = day;
  cout<<"Введіть місяць: "; cin>>month;
  tmstr.tm_mon = month-1; // Позаяк нумерація місяців починається з 0
  cout<<"Введіть рік: "; cin>>year;
  tmstr.tm_year = year-1900; // Оскільки роки починаються з 1900
  tmstr.tm_hour = 0;   tmstr.tm_min = 0;   tmstr.tm_sec = 0;
  // Виведення tmstr у форматі "день/місяць/рік"):
  strftime ( str, sizeof(str), "%d/%m/%Y", &tmstr );
  cout<<str<<endl;
  system("pause");
  return 0;
}
```

Результати виконання програми:

```
Введіть день: 25
Введіть місяць: 12
Введіть рік: 2019
25/12/2019
```

Вводити дату і час можна так, як показано у прикладі 3 (окремо день, місяць, рік, години, хвилини, секунди). Але інколи зручніше задавати дату (час) у вигляді рядку певного формату, який треба спочатку розбити на день, місяць та рік (години, хвилини, секунди) за допомогою, наприклад, функції `sscanf`. Розглянемо це на прикладі.

Приклад 4. Дату і час, записані у вигляді рядків у форматі "тільки дата", "тільки час" та "дата і час", перетворити на тип `time_t`.

Текст програми:

```
#include <iostream>
#include <time.h>
using namespace std;
int main ()
{ setlocale(0,".1251");
  char *sd="12/04/2020";
  char *st="08:14:45";
  char *sdt="12/04/2020 08:14:45";
  struct tm tmstr;
  puts(sd);
  sscanf(sd,"%d/%d/%d", &tmstr.tm_mday, &tmstr.tm_mon, &tmstr.tm_year);
  tmstr.tm_mon-=1; tmstr.tm_year-=1900;
  // Позаяк час не задано, задаємо нулі замість годин, хвилин і секунд
  tmstr.tm_hour=0; tmstr.tm_min=0; tmstr.tm_sec=0;
```

```
  time_t t1=mktime (&tmstr); printf("%s\n", ctime(&t1));
  puts(st);
  sscanf(st, "%d:%d:%d", &tmstr.tm_hour, &tmstr.tm_min, &tmstr.tm_sec);
  // Замість дати задаємо будь-яку, наприклад, 01.01.2020
  tmstr.tm_mday=1; tmstr.tm_mon=0; tmstr.tm_year=120;
  t1=mktime (&tmstr);   printf("%s\n", ctime(&t1));
  puts(sdt);
  sscanf(sdt, "%d/%d/%d %d:%d:%d", &tmstr.tm_mday, &tmstr.tm_mon,
     &tmstr.tm_year, &tmstr.tm_hour, &tmstr.tm_min, &tmstr.tm_sec);
  tmstr.tm_mon-=1; tmstr.tm_year-=1900;
  t1=mktime (&tmstr);
  printf("%s\n", ctime(&t1));
  system("pause");
  return 0;
}
```

Результат виконання програми:

```
12/04/2020
Sun Apr 12 00:00:00 2020

08:14:45
Wed Jan 01 11:04:05 2020

12/04/2020 08:14:45
Sun Apr 12 09:14:45 2020
```

Приклад 5. Ввести дату народження людини і визначити її вік (кількість повних років).

Розв'язок. Звертаємо увагу, що для дат до 01.01.1970 р. тип `time_t` використовувати не можна. Тому для обчислення віку в секундах потрібно розраховувати окремо проміжок у секундах між поточною датою та днем народження у 1970 році й окремо кількість секунд у роках між роком народження і 1970.

Текст програми:

```
#include <iostream>
#include <time.h>
using namespace std;
int main ()
{ setlocale(0,".1251");
  char birthday[30];
  time_t t, t1;
  struct tm tmstr;
  int year, seconds, yearinseconds;
  yearinseconds=(int)365.25*24*60*60;
  t = time(NULL);  // поточна дата
  cout<<"Введіть дату народження: ";
  gets_s(birthday,30);
  sscanf_s(birthday, "%d/%d/%d", &tmstr.tm_mday, &tmstr.tm_mon,
                                 &tmstr.tm_year);
```

```
  tmstr.tm_mon-=1; tmstr.tm_year-=1900;
  tmstr.tm_hour=0; tmstr.tm_min=0; tmstr.tm_sec=0;
  if (tmstr.tm_year>=70)
    { t1=mktime(&tmstr); seconds=difftime(t,t1); }
  else
    {
     year=70-tmstr.tm_year; tmstr.tm_year=70;
     t1=mktime(&tmstr);
     seconds=difftime(t,t1)+(int)year*365.25*24*60*60;
    }
  cout<<"Кількість повних років = "<<seconds/yearinseconds<<endl;
  system("pause");
  return 0;
}
```

Результати виконання програми:

1) Дата народження після 1970 року:

```
Введіть дату народження: 12/07/1997
Кількість повних років = 22
```

2) Дата народження до 1970 року:

```
Введіть дату народження: 12/07/1969
Кількість повних років = 50
```

Приклад 6. Ввести час початку кіносеансу і визначити час його завершення, якщо тривалість показу 1 година 45 хвилин.

Розв'язок. Введемо час початку сеансу у форматі чч:мм у рядкову змінну `st_time`, а час закінчення сеансу – у рядкову змінну `end_time`.

Структура `tmstr` містить поточну дату, години і хвилини (початок кіносеансу), які вводяться з клавіатури, а `endstr` вказує на час закінчення сеансу.

Текст програми:

```
#include <iostream>
#include <time.h>
using namespace std;
int main ()
{ setlocale(0,".1251");
  char st_time[30], end_time[30];
  time_t end_t, t, t1;    struct tm tmstr, *endstr;
  t = time(NULL);
  tmstr=*localtime(&t);
  cout<<"Введіть час початку сеансу: ";
  gets_s(st_time,30);
  sscanf_s(st_time, "%d:%d", &tmstr.tm_hour, &tmstr.tm_min);
  t1=mktime (&tmstr);
  end_t=t1+(45+60)*60;
  endstr=localtime(&end_t);
  strftime(end_time,30,"%H:%M", endstr);
```

```
  puts(end_time);
  system("pause");
  return 0;
}
```

Результат виконання програми:

```
Введіть час початку сеансу: 15:40
Час закінчення сеансу: 17:25
```

Приклад 7. Ввести дату виготовлення продукту і термін його зберігання. Визначити, чи закінчився його термін придатності.

Текст програми:

```
#include <iostream>
#include <time.h>
using namespace std;
int main ()
{
  setlocale(0,".1251");
  char st_time[30], term[30];
  time_t t, t1, end_t;
  struct tm tmstr, termstr;   int d, m, y, iterm;
  t = time(NULL);
  cout<<"Введіть дату виготовлення: ";
  gets_s(st_time,30);
  cout<<"Введіть термін придатності: ";
  gets_s(term,30);
  sscanf_s(st_time, "%d/%d/%d", &tmstr.tm_mday, &tmstr.tm_mon,
                                &tmstr.tm_year);
  tmstr.tm_mon-=1; tmstr.tm_year-=1900;
  tmstr.tm_hour=0; tmstr.tm_min=0; tmstr.tm_sec=0;
  sscanf_s(term, "%d/%d/%d", &d, &m, &y);
  iterm=(int)(y*365.25+m*30+d)*24*60*60;
  t1=mktime (&tmstr);
  end_t=t1+iterm;
  printf("Кінцевий термін: %s\n", ctime(&end_t));
  if(t>end_t) cout<<"Термін зберігання закінчився"<<endl;
  else  cout<<"Термін зберігання ще не закінчився"<<endl;
  system("pause");
  return 0;
}
```

Результати виконання програми:

1) Термін зберігання 5 днів:

```
Введіть дату виготовлення: 12/11/2019
Введіть термін придатності: 5/0/0
Кінцевий термін: San Nov 17 00:00:00 2019

Термін зберігання закінчився
```

2) Термін зберігання 3 місяці:

```
Введіть дату виготовлення: 12/11/2019
Введіть термін придатності: 0/3/0
Кінцевий термін: Sun Jun 12 01:00:00 2020

Термін зберігання ще не закінчився
```

3) Термін зберігання 2 роки:

```
Введіть дату виготовлення: 12/03/2019
Введіть термін придатності: 0/0/2
Кінцевий термін: Tue Mar 11 00:00:00 2021

Термін зберігання ще не закінчився
```

Приклад 8. Написати мовою С++ програмний код для створення текстового файлу з розширенням rtf, заповнення файлу даними про працівників (прізвище, ім'я, по батькові посада, дата прийому на роботу, зарплатня), перегляданння вмісту сформованого файлу та відбору із загального переліку на певній посаді з певним стажем (назву посади та стаж у роках вводитиме користувач).

Текст програми:

```
#include <iostream>
#include <time.h>
using namespace std;

struct employee
{
   char surname[15];
   char name[15];
   char patronymic[15];
   char position[20];
   char start_date[10];
   float salary;
 };
void create_file(char* name) // Функція створення файлу
{
  FILE* f;
  char s[100];
  f=fopen(name, "at+");  // Відкрити або створити текстовий файл
  setlocale(0,".1251");
  if (f==NULL)           // Перевірити, чи відкрився файл
   { cout<<"Неможливо створити файл\n"; return;}
  cout<<"Введіть рядки такої структури:"<<endl;
  cout<<"Прізвище \t Ім'я \t По батькові \t Посада \t Дата прийому
        \t Зарплатня\n";
  setlocale(0,".OCP");
  do                          // Цикл, поки не введено порожній рядок,
   { gets_s(s,100);           // ввести рядок із клавіатури,
```

```
     if (strlen(s)>1)
     {  fputs(s, f);        // записати його до файлу
        fputs("\n", f);     // і перейти на новий рядок
   } }
  while (strcmp(s,""));
  fclose(f);                // Закрити і зберегти файл
}
void select_data(char* name)
{
  setlocale(0,".1251");
  char pos[15];   int y;
  cout<<"\nВведіть назву посади для відбору працівників - ";
  cin>>pos;
  cout<<"\nВведіть стаж у роках для відбору - ";
  cin>>y;
  employee z; FILE* f;
  time_t t, t1;   struct tm tmstr;
  int seconds, yearinseconds=365.25*24*60*60;
  t = time(NULL);           // Поточна дата й час
  f=fopen(name, "rb");
  if (f==NULL) { cout<<"Неможливо відкрити файл \n"; return;}
  cout<<"\nВідібрані дані:"<<endl;
  cout<<"Прізвище \t Ім'я \t По батькові \t Посада \t Дата прийому \t
        Зарплатня\n"<<endl;
  while (!feof(f))
  { setlocale(0,".OCP");
    fscanf (f, "%s \t%s \t %s\t %s \t%s \t%f\n", z.surname, z.name,
           z.patronymic, z.position, z.start_date, &z.salary);
    sscanf(z.start_date,"%d.%d.%d", &tmstr.tm_mday, &tmstr.tm_mon,
           &tmstr.tm_year);
    tmstr.tm_mon-=1; tmstr.tm_year-=1900;
    tmstr.tm_hour=0; tmstr.tm_min=0; tmstr.tm_sec=0;
    t1=mktime ( &tmstr ); seconds=difftime(t,t1);
    if ( seconds>yearinseconds*y && strcmp(z.position,pos)==0)
       printf ("%s \t %s\t %s \t %s \t %s \t  %6.2f\n", z.surname,
         z.name, z.patronymic, z.position, z.start_date, z.salary);
  }
  fclose(f);
}
void view_text_file(char* name)
{
  employee z; FILE* f;
  f=fopen(name, "rt");
  setlocale(0,".1251");
  if (f==NULL)
      { cout<<"Неможливо відкрити файл\n"; return;}
  cout<<"\nПерегляд файлу"<<endl;
```

```
  cout<<"Прізвище \t Ім'я \t По батькові \t Посада \t Дата прийому
        \t Зарплатня\n";
  setlocale(0,".OCP");
  while(!feof(f))
  {
    fscanf (f, "%s\t%s\t%s\t%s\t%s\t%f\n", z.surname, z.name,
        z.patronymic, z.position, z.start_date, &z.salary);
    printf ("%s \t %s\t %s \t %s \t %s \t  %6.2f\n", z.surname,
        z.name, z.patronymic, z.position, z.start_date, z.salary);
  }
fclose(f);
}
int main()
{
    system("color F0");
    char fn[]="pr.rtf";
    create_file(fn);
    view_text_file(fn);
    select_data(fn);
    system("pause");
    return 0;
}
```

Результати роботи:

```
Введіть рядки такої структури:
Прізвище        Ім'я    По батькові     Посада  Дата прийому    Зарплатня
Єщенко          Федір   Миколайович     інженер   23.10.2000    10000


Перегляд файлу
Прізвище        Ім'я    По батькові     Посада  Дата прийому    Зарплатня
Антонюк          Олег    Петрович        електрик        11.02.1999       4500,00
Василюк          Василь  Олегович        слюсар          12.04.2019       4300,00
Донченко         Ольга   Іванівна        бухгалтер       12.11.2019       6000,00
Кравчук          Данило  Маркович        бухгалтер       12.09.2009       10000,00
Лещенко          Василь  Андрійович      програміст      03.08.2005       9900,00
Максимча         Наталя  Андріївна       бухгалтер       25.05.2015       6000,00
Петренко         Петро   Петрович        менеджер        09.11.2014       7000,00
Тищенко          Олег    Ігорович        директор        22.12.2013       15000,00
Якименко         Микола  Петрович        бухгалтер       12.09.2000       6500,00
Зайченко         Тамара  Федорівна       бухгалтер       18.05.1998       7800,00
Єщенко   Федір   Миколайович     інженер         23.10.2000       10000,00

Введіть назву посади для відбору працівників - бухгалтер

Введіть стаж у роках для відбору - 5

Відібрані дані:
Прізвище        Ім'я    По батькові     Посада  Дата прийому    Зарплатня

Кравчук          Данило  Маркович        бухгалтер       12.09.2009       10000,00
Якименко         Микола  Петрович        бухгалтер       12.09.2000       6500,00
Зайченко         Тамара  Федорівна       бухгалтер       18.05.1998       7800,00
Press any key to continue . . .
```

Вигляд створеного текстового файлу "pr.rtf".

```
Антонюк     Олег        Петрович       електрик     11.02.1999    4500
Василюк     Василь      Олегович       слюсар       12.04.2019    4300
Донченко    Ольга       Іванівна       бухгалтер    12.11.2019    6000
Кравчук     Данило      Маркович       бухгалтер    12.09.2009    10000
Лещенко     Василь      Андрійович     програміст   03.08.2005    9900
Максимча    Наталя      Андріївна      бухгалтер    25.05.2015    6000
Петренко    Петро       Петрович       менеджер     09.11.2014    7000
Тищенко     Олег        Ігорович       директор     22.12.2013    15000
Якименко    Микола      Петрович       бухгалтер    12.09.2000    6500
Зайченко    Тамара      Федорівна      бухгалтер    18.05.1998    7800
Єщенко      Федір       Миколайович    інженер      23.10.2000    10000
```

Питання та завдання для самоконтролю

1) Як зберігається час у форматі time_t?
2) Як подано час у форматі struct tm?
3) Які методи дозволяють визначити поточну системну дату?
4) Записати команди, які дозволять збільшити (чи зменшити) дату:
   а) на один день; б) на один тиждень; в) на один місяць?

Завдання до самостійної роботи

1) Надати відповіді на контрольні запитання.

2) Написати програмний код мовою C++ для формування текстового файлу даними (вміст файлу зазначено у табл. 14.7), переглядання вмісту сформованого файлу та розв'язання індивідуальних завдань з відбирання тих записів файлу, які задовольняють певній умові (завдання вибрати з табл. 14.7).

3) Створити на комп'ютері програмні проєкти мовою C++ для реалізації написаних програм.

Таблиця 14.7

Індивідуальні завдання

№ вар.	Вміст текстового файлу	Завдання
1	Список для автоінспекції: відомості про викрадені автомобілі: державний номер, марка автомобіля, колір, дата заяви	Вивести відомості про викрадені автомобілі марки "BMW", які було викрадено понад два роки тому
2	Відомості про сплату комунальних послуг: вулиця, номер будинку, прізвище мешканця, дата сплати, борг	Вивести відомості боржників з боргом понад 500 грн., які не сплачували послуги вже 3 місяці
3	Список людей, які стоять у черзі на встановлення телефону: прізвище, дата подання заяви, пільги ("Так" або "Ні")	Вивести список тих, хто має пільги і стоїть у черзі понад два роки

Продовження табл. 14.7

№ вар.	Вміст текстового файлу	Завдання
4	Літній розклад руху потягів: номер потяга, пункт відправлення, пункт призначення, час відправлення	Вивести відомості про потяги напрямку Одеса – Київ, які відправляються з 10:00 до 17:00
5	Журнал подій операційної системи: назва запущеної програми, рівень події (*помилка*, *попередження* тощо), дата події, час події	Вивести дані про помилки та визначити скільки днів пройшло від кожної з помилок на поточний момент часу
6	Список ліній інтенсивного зв'язку метеорного каналу: напрямок зв'язку, час початку (hh:mm) й закінчення (hh:mm) інтенсивного зв'язку, місяць	Вивести відомості про напрямок з найбільшим інтервалом зв'язку у травні-місяці
7	Відомості про медикаменти в аптеці: назва ліків, дата кінцевого терміну придатності, ціна	Вивести дані про медикаменти, термін придатності яких закінчується: 1) у поточному році; 2) в межах 30-ти днів
8	Список співробітників підприємства: табельний номер, прізвище, стать (ч/ж), дата народження, посада	Вивести відомості про усіх співробітників-жінок старше 55-ти років та чоловіків старше 60-ти років
9	Відомості про товари: назва ліків, дата виготовлення, дата кінцевого терміну реалізації, ціна	Вивести відомості про товари з терміном реалізації: 1) менше 10-ти днів; 2) в поточному місяці
10	Розклад телепередач: назва, початок у форматі чч:хх, тривалість у форматі чч:хх (бажано вводити передачі у порядку їх показу)	Визначити для кожної передачі час закінчення, а також вивести відомості про вечірні передачі (з 18:00 до 22:00)
11	Розклад руху приміських потягів: номер потяга, кінцевий пункт призначення, час відправлення, тривалість подорожі	Обчислити час прибуття потягів та вивести відомості про маршрути тривалістю більше трьох годин
12	Список співробітників підприємства: табельний номер, прізвище, посада, дата народження, дата прийому на роботу	Вивести відомості про співробітників, які пропрацювали понад десять років на підприємстві
13	Список святкових днів у календарі: назва свята, дата у форматі (дд.мм.гггг)	Вивести дані про зимові свята та визначити найближче свято
14	Розклад руху літаків: номер рейсу, пункт призначення, час відправлення	Вивести дані про початок і закінчення реєстрації (відповідно за 2 години і 40 хвилин до відправлення)
15	Список студентів групи: прізвище, ім'я, дата народження, середній бал успішності	Визначити вік кожного студента. Знайти наймолодшого студента
16	Репертуар оперного театру: назва вистави, жанр, дата, початок	Вивести відомості про вистави жанру балет, які ще не відбулися, а також дитячі вистави (початок до 15 годин)

Продовження табл. 14.7

№ вар.	Вміст текстового файлу	Завдання
17	Майстерня телефонної мережі: прізвище абонента, номер телефону, дата поломки, дата усунення	Вивести дані про поломки у минулому місяці, а також визначити термін усунення кожної поломки у днях
18	Список співробітників підприємства: табельний номер, прізвище, стать, дата народження, посада	Вивести відомості про співробітників, які святкують у цьому році ювілей (кількість років кратна 5)
19	Розклад руху потягів: номер потяга, кінцевий пункт призначення, час відправлення, час прибуття	Визначити найдовший та найкоротший маршрути
20	Замовлення на стаціонарне підключення до Інтернет: прізвище, номер контактного телефону, дата замовлення	Вивести для кожного замовлення дату можливого підключення, якщо термін складає 5 днів. Вивести усі замовлення минулого місяця
21	Статистика відправлень SMS у телефоні: номер телефону адресата, дата і час відправлення	Вивести дані про SMS, відправлені минулого тижня, а також SMS, відправлені вночі (з 23:00 до 4:00)
22	Список товарів у магазині: назва, виробник, дата виготовлення, термін зберігання у днях	Визначити для кожного продукту дату закінчення терміну зберігання та вивести дані про прострочені продукти
23	Звіт про сплату комунальних платежів: особистий рахунок, прізвище, сума сплати, дата сплати	Вивести дані про сплачені послуги за останні три місяці
24	Список співробітників підприємства: табельний номер, прізвище, дата народження, посада, дата прийому на роботу	Вивести дані про найстаршого за віком співробітника та найменшого за стажем на підприємстві співробітника
25	Розклад руху пасажирських потягів на проміжній станції: номер потяга, назва станції призначення, час прибуття, час відправлення	Вивести дані про стоянки понад 15 хвилин та віднайти найдовшу стоянку
26	Відомості про реєстрацію співробітників перед початком робочого дня: прізвище, посада, час прибуття на роботу	Вивести дані про робітників, які спізнилися (робочий день починається о 9:00), за винятком директора та начальників відділів
27	Результати змагань плавців: прізвище учасника, дистанція (у м), час у форматі чч:хх:сс	Визначити переможця на дистанції 100 м

Закінчення табл. 14.7

№ вар.	Вміст текстового файлу	Завдання
28	Репертуар кінотеатру: назва фільму, назва залу, початок сеансу у форматі чч:хх, тривалість у форматі чч:хх	Вивести дані про фільми, які починаються увечері з 16:00 до 19:00, а також фільми тривалістю понад 2 години
29	Військкомат: прізвище, ім’я, звання, дата призову, дата звільнення у запас	Вивести дані про військовозобов'язаних, які ще не звільнені (дата звільнення не заповнена)
30	Журнал перезавантажень сервера: дата, час, причина, прізвище відповідального	Вивести відомості про перезавантаження системи протягом минулого тижня та ті перезавантаження, які відбулися вночі з 23:00 до 5:00

Лабораторна робота № 16
Бінарні файли

Мета роботи: набути практичних навиків програмного створення та редагування бінарних файлів засобами C++.

Теоретичні відомості

1. Загальні відомості про бінарні файли

Бінарні файли зберігають дані у тому самому форматі, в якому вони були оголошені, і їхній вигляд є такий самий, як і в пам'яті комп'ютера. І тому відпадає потреба у використанні розділювачів: пробілів, керувальних послідовностей, а отже, розмір бінарного файлу порівняно з текстовим файлом з аналогічними даними буде значно меншим. Окрім того, немає потреби у застосуванні функцій перетворювання числових даних при зчитуванні-записуванні. Слід зазначити, що опрацювання (перегляд, запис та ін.) бінарних файлів можливе лише з програми, яка знає, що саме і в якій послідовності зберігається у цьому файлі.

2. Робота з бінарними файлами у стилі C

З двійковими файлами можна виконувати ті ж самі дії, що і з текстовими. Для відкривання бінарного файлу використовується та сама команда `fopen()`, лише у другому параметрі (режимі відкривання файлу) замість літери «`t`» треба записати літеру «`b`». Наприклад, бінарний файл з ім'ям `tmp.dat` можна відкрити для зчитування такою командою:

```
f = fopen("tmp.dat", "rb");
```

де `f` – вказівник типу `FILE*`.

Записування і зчитування у двійкових файлах найчастіше здійснюються за допомогою відповідно функцій `fwrite()` та `fread()`.

Прототип функції зчитування `fread()`:

```
size_t fread
( char *buffer,        // Масив для зчитування даних,
  size_t elemSize,     // розмір одного елемента,
  size_t numElems,     // кількість елементів для зчитування
  FILE *f              // і вказівник потоку
);
```

Тут `size_t` означений як беззнаковий цілий тип у системних заголовних файлах. Функція намагається прочитати `numElems` елементів з файлу, який задається вказівником `f`, розмір кожного елемента становить `elemSize`. Функція повертає реальну кількість прочитаних елементів, яка може бути менше за `numElems`, у разі завершення файлу чи то помилки зчитування.

Приклад використання функції `fread()`:

```
FILE *f;
double buff[100];
```

```
size_t res;
f = fopen ("tmp.dat", "rb");    // Відкриття файлу
if (f == 0)
 {  cout<<"Не вдається відкрити файл для зчитування"<<endl;
    exit (1);                   // Завершення роботи з кодом 1
 }
// Зчитування 100 дійсних чисел з файлу
res = fread(buff, sizeof(double), 100, f);
// res дорівнює реальній кількості прочитаних чисел
```

У даному прикладі файл `tmp.dat` відкривається для зчитування як бінарний, з нього читаються 100 дійсних чисел розміром 8 байтів кожне. Функція `fread()` повертає реальну кількість прочитаних чисел, яка є менше чи то дорівнює 100.

Функція `fread()` читає інформацію у вигляді потоку байтів і в незмінному вигляді розміщує її в пам'яті. Варто розрізнювати текстове подавання чисел і їхнє бінарне подавання. У наведеному вище прикладі числа у файлі має бути записано у *бінарному форматі*, а не у вигляді тексту. Для текстового введення чисел треба використовувати функції введення за форматом, які було розглянуто вище.

Функція бінарного записування у файл `fwrite()` є аналогічна до функції зчитування `fread()`. Вона має такий прототип:

```
size_t fwrite
( char *buffer,       // Масив записуваних даних,
  size_t elemSize,    // розмір одного елемента,
  size_t numElems,    // кількість записуваних елементів
  FILE *f             // і вказівник потоку
);
```

Функція повертає кількість реально записаних елементів, яка може бути менше за `numElems`, якщо при записуванні виникла помилка: наприклад, не вистачило вільного простору на диску.

Приклад використання функції `fwrite()`:

```
FILE *f;
double buff[100];
size_t res;
. . .   // Введення елементів масиву buff[100]
f = fopen("tmp.dat", "wb");  // Створення бінарного файлу
if (f == 0)
  { cout<<"Не вдається відкрити файл для запису"<<endl;
    cin.get();
    exit(1);        // Завершення  роботи програми з кодом 1
  }
// Записування 100 дійсних чисел у файл
res = fwrite(buff, sizeof(double), 100, f);
// У разі успішного записування  res = 100
```

До обох функцій передається вказівник на дані, які виводяться чи вводяться, параметр `elemSize` задає розмір передаваних даних у байтах, а параметр `numElems` визначає кількість таких даних.

Так само, як і в текстових, у бінарних файлах можна визначати позицію зчитування-записування і переміщувати її у довільне місце файлу командами відповідно `ftell()` та `fseek()`. Можливість переміщувати позицію є корисна для файлів, які складаються з однорідних записів однакового розміру. Наприклад, якщо у файлі записано лише дійсні числа типу `double`, то для того, щоб прочитати `i`-те число, треба виконати оператори:

```
fseek(F, sizeof(double)*(i-1), 0);
fread(&a, sizeof(double), 1, F);
```

У такий спосіб можна читати які завгодно записи у будь-якій послідовності. За допомогою переміщування позиції можна редагувати записи у файлі. Нехай, приміром, треба змінити у файлі одне з чисел, помноживши його на 10. Це можна зробити, якщо відкрити файл у режимі зчитування і записування (наприклад, у форматі `"rb+"`), переміститись у відповідну позицію і виконати оператори:

```
fread(&z, sizeof(double), 1, F);
z *= 10;
fseek(F, -sizeof(double), 1);
fwrite(&z, sizeof(double), 1, F);
```

Перший з цих операторів читає число з файлу у дійсну змінну `z` типу `double`, другий – помножує її на 10. Третій оператор повертає позицію на один запис назад, оскільки після виконання `fread()` позиція просунулась уперед. Останній оператор пише в ту позицію, з якої було прочитано число, нове значення.

Те ж саме завдання можна розв'язати в інший спосіб:

```
long int pos = ftell(F);      // Запам'ятовування позиції
fread(&z,sizeof(double),1,F);// Зчитування числа з файлу до змінної
z *= 10;
fseek(F,pos,0);               // Відновлення позиції
fwrite(&z,sizeof(double),1,F);//Записування числа зі змінної z у файл
```

Тут функція `ftell()` визначає поточну позицію файлу, з якої читається число, а функція `fseek()` відновлює цю позицію перед записуванням зміненого числа.

Як було зазначено вище, за допомогою двійкових файлів можна записувати й зчитувати не лише числа та рядки, а й структури (див. далі приклад 1).

Приклади програм

Приклад 1. Ввести відомості про екзаменаційні оцінки студентів: прізвище та оцінка. Вивести відомості про студентів, які отримали «незадовільно» (менше за 60 балів).

Текст програми:

```
#include <iostream>
#include <stdio.h>
using namespace std;
```

```
struct ved                  // Оголошення структури
{
  char priz[15];
  int oc;
}
v;

void writing(char* fname)
{
  FILE *f;                  // Оголошення файлової змінної
  f=fopen(fname,"rb+");     // Відкривання файлу для зчитування і записування
  if (f==NULL)              // Якщо файл не вдалось відкрити,
    f=fopen(fname,"wb+"); // створити його
  fseek(f, 0, SEEK_END);  // Перехід у кінець файлу
  char yn;                  // Ознака бажання дописувати нові рядки у файл
rep:
  setlocale(0,".1251");
  cout<<"Будете вводити дані у файл? y/n"<<endl;
  cin>>yn;
  if (yn != 'y')
    { fclose(f); return; }
  cout<<"Введіть прізвище та оцінку через пробіл"<<endl;
  setlocale(0,".OCP");
  cin>>v.priz>>v.oc;
  fwrite (&v, sizeof(ved),1,f); // Дописування структури у файл
  goto rep;
}

void look(char* fname)
{
  FILE *f;
  if ((f=fopen(fname,"rb"))==0) // Відкриття файлу для зчитування
   { cout<<"Не вдається відкрити файл для читання"<<endl;
     cin.get(); return;
   }
  setlocale(0,".1251");
  cout<<"\nВміст файлу:\n";
  setlocale(0,".OCP");
  while (fread(&v, sizeof(ved),1,f)) // Зчитування з файлу в змінну v
  {
    for (int i=strlen(v.priz);i<15; i++)
      strcat(v.priz," ");
    cout<<v.priz<<" "<<v.oc<<endl;
  }
  fclose(f);
  return ;
 }
```

```
void notpassed (char* fname)
{
  int k=0;
  FILE *f;
  if ((f=fopen(fname,"rb"))==0)
   {
     cout<<"Не вдається відкрити файл для читання"<<endl;
     cin.get();
     return;
   }
  setlocale(0,".1251");
  cout<<"\nНезадовільну успішність мають:\n";
  setlocale(0,".OCP");
  while (fread(&v, sizeof(ved), 1, f)>0)
   if (v.oc < 60)
    {
      cout << v.priz << "  " << v.oc << endl;
      k++;
    }
  setlocale(0,".1251");
  cout<<"Всього студентів, які мають незадовільну успішність - "
      << k << endl;
  fclose(f);
  return;
}

int main()
{
  char s[]="ocinki.dat"; // визначення імені файлу
  writing(s);
  look(s);
  notpassed(s);
  system ("pause>>void");
  return 0;
}
```

Результат виконання програми:

```
Будете вводити дані у файл? у/п
у
Введіть прізвище та оцінку через пробіл
Чабаненко  74
Будете вводити дані у файл? у/п
п

Вміст файла:
Абрамов          90
Артеха           90
Балух            89
Качур            65
Кучер            34
Лантінов         25
Логінова         98
Іванішин         20
Гулько           90
Шкраба           18
Чабаненко        74

Незадовільну успішність мають:
Кучер  34
Лантінов  25
Іванішин  20
Шкраба  18
Всього студентів, які мають незадовільну успішність - 4
```

Питання та завдання для самоконтролю

1) Який файл називають бінарним?
2) В який спосіб задається режим відкривання бінарного файлу у стилі С?
3) Записати команду для створення бінарного файлу.
4) Записати команду для переміщення на початок бінарного файлу і ще одну команду для переміщення у кінець цього файлу. В яких випадках ці команди будуть доречні для використання?
5) Назвати функції файлового записування-зчитування даних у стилі С.

Лабораторне завдання

1) У протоколі лабораторної роботи дати відповіді на контрольні запитання.

2) У протоколі лабораторної роботи написати мовою С++ програмний код для створення бінарного файлу з ім’ям Вашого прізвища і розширенням dat, заповнення файлу даними, перегляддання вмісту сформованого файлу та розв’язання індивідуальних завдань, які вибрати з табл. 16.1.

3) Створити на комп’ютері програмні проєкти мовою С++ для реалізації написаних програм.

Таблиця 16.1

Варіанти завдань для роботи зі структурами

№ вар.	Поля структури	Індивідуальні завдання
1	*Сесійні дані про студентів:* – прізвище, – група, – фізика – інформатика – історія	Визначити середній бал оцінок кожного студента і відібрати студентів, середній бал яких більше 75
2		Визначити середній бал оцінок усіх студентів з фізики і відібрати студентів, які склали екзамен з інформатики на "відмінно" (≥90)
3		Визначити кількість студентів, які не склали екзамен з інформатики, та визначити їх середній бал
4		Відібрати студентів-відмінників та визначити їхню кількість
5		Відібрати студентів, які здали сесію, але не "дотягують" до стипендії, та обчислити відсотковий вміст таких студентів серед решти представлених студентів
6		Відібрати студентів, які мають незадовільну оцінку хоча б з одного екзамену, та визначити їхню кількість
7	*Дані про працівників:* – прізвище, – посада, – освіта – рік народження, – зарплатня	Визначити працівників з найбільшою та найменшою зарплатнею і визначити їх сумарну зарплатню
8		Відібрати працівників, яким до пенсії (до 6-ти років) лишилося менше 3-х років, та обчислити їх відсотковий вміст серед решти працівників
9		Визначити наймолодшого і найстаршого працівників
10		Визначити працівників, які в поточному році святкують ювілей – вік кратний 5 чи 10-ти
11		Відібрати працівників молодших 30-ти років та обчислити їхню кількість
12		Відібрати працівників, зарплатня яких більше середнього значення зарплатні всіх працівників

Закінчення табл. 16.1

№ вар.	Поля структури	Індивідуальні завдання
13	*Товари на складі*: – найменування, – виробник, – ціна, – кількість	Визначити найдорожчий товар на складі, вивести всі дані про нього та обчислити його сумарну вартість
14		Обчислити загальну кількість та середню ціну товарів
15		Обчислити сумарну вартість кожного товару та сумарну вартість усіх товарів на складі
16		Відібрати товари, кількість яких менше 10-ти, обчислити кількість найменувань і сумарну кількість таких товарів
17		Визначити товар з найбільшою загальною вартістю на складі
18		Визначити товар з найбільшою кількістю на складі та обчислити його сумарну вартість
19	*Характеристики вебсайту:* – повне доменне ім’я; – категорія (особистий, корпоративний, інформаційний, промосайт, блог, Інтернет-магазин, соціальна мережа тощо); – кількість відвідувань в день; – рейтинг PR (Page Rank – число від 0 до 10)	Відібрати сайти з рейтингом понад 5 та обчислити їхню кількість серед представлених
20		Відібрати сайти на доменах "ua" та "com", обчислити їхню кількість
21		Обчислити середній рейтинг та середню кількість відвідувань за день усіх представлених сайтів
22		Відібрати сайти з категорії "blog" та обчислити їхню кількість серед представлених
23		Відібрати сайти на домені "ua" з рейтингом PR понад 7 та обчислити їхню кількість серед представлених
24		Відібрати сайти з кількістю відвідувань понад 50 та обчислити кількість таких сайтів серед представлених
25		Визначити сайт з найбільшим рейтингом
26		Відібрати сайти на домені "net" або "ru" та обчислити їхню кількість
27	*Література у видавництві*: – автор; – назва книги; – рік видання; – тираж; – кількість сторінок	Відібрати книги з тиражем до 1000 екземплярів та обчислити кількість таких книг серед представлених
28		Відібрати книги, видані у поточному році та обчислити кількість таких книг
29		Визначити найтовстішу книгу (максимальна кількість сторінок)
30		Відібрати книги з кількістю сторінок понад 150 та обчислити кількість таких книг

Самостійна робота № 15

Робота з бінарними файлами

Мета роботи: закріпити практичні навички програмного створення та редагування бінарних файлів засобами C++.

Приклад програми

Приклад. Створити програмний проєкт, який реалізує такі завдання:

1) програмне створення бінарного файлу про успішність студентів і заповнення його двійковими даними: прізвище, ім'я, група, дата нагородження, оцінки з трьох екзаменів;
2) обчислити середній бал кожного студента;
3) програмне переглядання даних створеного файлу;
4) відбір даних файлу про ювілярів (вік кратний 5) у найближчі три місяці;
5) відбір повнолітніх студентів, яким виповнилось 18 років.

Програмний код:

```
#include <iostream>
#include <conio.h>
#include <stdio.h>
#include <math.h>
#include <ctime>
using namespace std;

// Оголошення структури student
struct student
{
   char surname[20];
   char name[20];
   char group[20];
   char date_birth[10];
   int ex1,ex2,ex3;
   float avg;
};
// Дописування одного запису у бінарний файл.
// Параметри – вказівник на файл та ім'я файлу
void add_record(char* name)
{
  char ozn;
next:
  setlocale(0,".1251");
  cout<<"Якщо бажаєте дописати дані у файл, натисніть клавішу <y> ->";
  cin>>ozn;
```

```
  if (ozn == 'y')
  {
    student z; FILE* f;
   // Відкрити файл та перевірити успішність відкривання
    f = fopen(name, "ab");
    if (f==NULL) { cout<<"Не вдається відкрити файл\n"; return; }
    cout<<"\nВведіть рядок такого формату"<<endl;
    cout<<"Прізвище Ім'я   Група Дата народж.  Оцінка1 Оцінка2 Оцінка3\n";
    setlocale(0,".OCP");
   // Зчитати запис з клавіатури
    scanf("%s %s %s %s %i %i %i", z.surname, z.name, z.group,
          z.date_birth, &z.ex1, &z.ex2, &z.ex3);
    z.avg = (z.ex1+z.ex2+z.ex3)/3.0;
    fwrite (&z, sizeof(student), 1, f);    // Записати запис у файл
    fclose(f);    // Закрити файл
    goto next;
  }
  else return;
}

// Перегляд файлу. Параметри – вказівник на файл та ім'я файлу
void view_file(char* name)
{ student z;  FILE* f;
  f=fopen(name, "rb");
  setlocale(0,".1251");
  if (f==NULL) { cout<<"Не вдається відкрити файл\n"; return;}
  cout<<"\nПерегляд вмісту файлу"<<endl;
  cout<<"Прізвище \tІм'я   \tГрупа \tДата народж. Оцінка1 Оцінка2
        Оцінка3 Середній бал\n";
  setlocale(0,".OCP");
  // Зчитувати послідовно записи з файлу, допоки вони є у файлі
  while (fread(&z,sizeof(student), 1, f)>0)
  {  // Вивести запис на екран у рядок через табуляцію
     printf ("%s\t%s\t%s\t%s\t%i\t%i\t%i\t%5.1f\n", z.surname,
        z.name, z.group, z.date_birth, z.ex1, z.ex2, z.ex3,z.avg);
  }
  fclose(f);
}

// Визначення кількості записів у файлі. Параметри – вказівник на файл
// та ім'я файлу. Результат – ціле число (кількість записів)
int number_of_records(char* name)
{ int n;  FILE* f;
  f = fopen(name, "rb");
  setlocale(0,".1251");
  if (f==NULL) { cout<<"Не вдається відкрити файл\n"; return 0;}
  // Переставити курсор у кінець файлу (0 – зсув у байтах,
  // 2 – позиція - від кінця файлу)
```

```
  fseek(f,0,2);
  // Визначити поточне розташування (на скільки байтів від початку
  // файлу зміщений курсор) і розділити на розмір одного запису
  n=ftell(f)/sizeof(student);
  return n;
}

// Заповнення (створення) масиву з бінарного файлу.
// Параметри – масив, вказівник на файл, ім'я файлу.
void create_array(student *a, char* name)
{ int i=0; // індекс масиву
  student z; FILE* f;
  f = fopen(name, "rb");
  setlocale(0,".1251");
  if (f==NULL) { cout<<"Не вдається відкрити файл\n"; return; }
  // Зчитати по черзі усі записи, записати їх в елементи масиву,
  // збільшуючи індекси масиву на 1
  while (fread(&z,sizeof(student), 1, f))
  {
     a[i]=z; i++;
  }
  fclose(f);
}

//Сортування масиву за групами
void sort_array(student *a, int kol)
{ int i,j; student x; bool f;
  // Цикл допоки масив не впорядкований. Ознакою того, що масив вже
  // впорядкований буде значення 0 змінної f, після чергового проходу по масиву
  do
  { f=0;
    // При кожному проході по масиву порівнюються два сусідні елементи.
    // Вони обмінюються місцями, якщо стоять у неправильному порядку.
     for (i=0; i<kol-1; i++)
       if (a[i].avg <a[i+1].avg)
       { x=a[i]; a[i]=a[i+1]; a[i+1]=x;
         f=1;
       }
  } while(f);
}

// Створення бінарного файлу з динамічного масиву
void array_to_file(student *a, int kol, char* name)
{ FILE* f;
  f=fopen(name, "wb");
  setlocale(0,".1251");
  if (f==NULL) { cout<<"Не вдається відкрити файл\n"; return;}
  setlocale(0,".OCP");
```

```
  for (int i=0; i<kol; i++)
     fwrite (&a[i],sizeof(student), 1, f);
  fclose(f);
}

// Виведення даних за умовою – ювіляри цього року
void select_data1(char* name)
{
  student z; FILE* f;
  time_t t, t1;    struct tm tmstr;
  int day, month, year, seconds, yearinseconds;
  double age;
  char sday[3], smonth[3], syear[5], strdate[256];
  // Кількість секунд в одному році (в середньому)
  yearinseconds=365.25*24*60*60;
  t = time(NULL);  // Поточні дата й час
  setlocale(0,".1251");
  f=fopen(name, "rb");
  if (f==NULL) { cout<<"Не вдається відкрити файл\n"; return; }
  cout<<"\nВідібрані дані про ювілярів"<<endl;
  cout<<"Прізвище \tІм'я   \tГрупа \tДата народж. Вік\n";
  setlocale(0,".OCP");
  while(fread(&z,sizeof(student), 1, f)>0)
  {// Виокремити з рядка перші 2 символи (день) і перетворити їх на ціле число
   strncpy(sday,z.date_birth,2); sday[2]='\0'; day=atoi(sday);
   // Виокремити з рядка 4 і 5 символи (місяць) і перетворити їх на ціле число
   strncpy(smonth,&z.date_birth[3],2); smonth[2]='\0';
   month=atoi(smonth);
   // Копіювати залишок рядка з 7 символу (рік) і перетворити їх на ціле число
   strcpy(syear,&z.date_birth[6]); year=atoi(syear);
  // Присвоїти значення полям структури tmstr (для часу задати нулі)
   tmstr.tm_mday = day;
   tmstr.tm_mon = month-1;
   tmstr.tm_year = year-1900;
   tmstr.tm_hour = 0;   tmstr.tm_min = 0;   tmstr.tm_sec = 0;
  // Перетворити структуру tmstr на час типу time_t
   t1=mktime ( &tmstr );
  // Обчислити різницю у секундах між поточною датою і датою народження
   seconds=difftime(t,t1);
   age=(double)seconds/yearinseconds;
  // Якщо вік кратний 5 і до дня народження менше 3-х місяців вивести на екран
   if (int(ceil(age))%5 == 0 &&
      (int)yearinseconds*ceil(age) - seconds  < yearinseconds/4)
          printf ("%s\t%s\t%s\t%s\t%5.1f\n", z.surname, z.name,
                  z.group, z.date_birth, age);
  }
  fclose(f);
}
```

```
// Виведення даних за умовою – спосіб другий (повнолітні, виповнилось 18 років)
void select_data2(char* name)
{
  student z; FILE* f;
  time_t t, t1;   struct tm tmstr; double age;
  int seconds, yearinseconds=365.25*24*60*60;
  t = time(NULL);
  f=fopen(name, "rb");
  setlocale(0,".1251");
  if (f==NULL) { cout<<"Не вдається відкрити файл\n"; return;}
  cout<<"\nПовнолітні"<<endl;
  cout<<"Прізвище \tІм'я   \tГрупа \tДата народж. Оцінка1 Оцінка2
        Оцінка3 Середній бал\n";
  setlocale(0,".OCP");
  while (fread(&z,sizeof(student), 1, f))
  {
   // Розбити рядок на частини за форматом: 3 цілих числа (%d), розділені крапками
   // і записати їх у поля структури tmstr день, місяць і рік
    sscanf(z.date_birth,"%d.%d.%d", &tmstr.tm_mday, &tmstr.tm_mon,
           &tmstr.tm_year);
    // Відняти 1 від місяця і 1900 від року, часовим полям надати нулі
    tmstr.tm_mon-=1; tmstr.tm_year-=1900;
    tmstr.tm_hour=0; tmstr.tm_min=0; tmstr.tm_sec=0;
    // Перетворити у час типу time_t та обчислити різницю дат у секундах
    t1=mktime ( &tmstr ); seconds=difftime(t,t1);
    age=(double)seconds/yearinseconds;
    // Якщо вік (різниця у часі) понад 18 років (у секундах), вивести на кран
    if (age>18)
     printf ("%s\t%s\t%s\t%i\t%i\t%i\t%5.1f\n", z.surname,
     z.name, z.group, z.date_birth, z.ex1, z.ex2, z.ex3,z.avg);
  }
  fclose(f);
}

void sort(char* name)
{
  // Визначення кількості записів у файлі
  int kol = number_of_records(name);
  setlocale(0,".1251");
  student *a=new student [kol]; // Оголошення динамічного масиву
  create_array(a, name); // Заповнення динамічного масиву значеннями
  sort_array(a, kol);  // Сортування масиву за групами
 // Повернення відсортованих даних у бінарний файл
  array_to_file(a, kol, name);
 delete []a; // Очищення пам'яті від масиву
  setlocale(0,".1251");
  cout<<"\nВідсортований за рейтингом файл:"<<endl;
  view_file(name);  // Виведення файлу на екран
 }
```

```
// Головна функція
int main()
{
  system("color F0");
  setlocale(0,".1251");
  char* bname="stud.dat";
  add_record(bname);
  view_file(bname);  // Виведення файлу на екран
  sort(bname); // Сортування даних файлу за рейтингом студентів
  // Виведення даних за умовою – ювіляри цього року
  select_data1(bname);
  // Виведення даних за умовою – повнолітні, виповнилось 18 років
  select_data2(bname);
  system("pause>>void");
  return 0;
}
```

Результат виконання програми:

```
Якщо бажаєте дописати дані у файл, натисніть клавішу <y> ->y

Введіть рядок такого формату
Прізвище Ім'я     Група  Дата народж.  Оцінка1 Оцінка2 Оцінка3
Якименко  Дмитро  КТ-1.21  05.09.2002   90       89      90
Якщо бажаєте дописати дані у файл, натисніть клавішу <y> ->n

Перегляд вмісту файлу
Прізвище        Ім'я            Група   Дата народж. Оцінка1 Оцінка2 Оцінка3 Середній бал
Фоменчук        Костянтин       КТ-1.21 02.01.2005      80      90      90      86,7
Федченко        Рустам          ПЗ-1.01 01.01.2000      80      90      85      85,0
Донченко        Данило          ПЗ-1.01 21.09.2004      80      90      70      80,0
Буберенко       Петро   ПЗ-1.01 02.01.2005      60      90      74       74,7
Терещенко       Василь          КТ-1.23 01.02.1999      90      40      69      66,3
Василенко       Микола          ПЗ-1.02 03.04.2002      50      60      60      56,7
Якименко        Дмитро          КТ-1.21 05.09.2002      90      89      90      89,7

Відсортований за рейтингом файл:
Перегляд вмісту файлу
Прізвище        Ім'я            Група   Дата народж. Оцінка1 Оцінка2 Оцінка3 Середній бал
Якименко        Дмитро          КТ-1.21 05.09.2002      90      89      90      89,7
Фоменчук        Костянтин       КТ-1.21 02.01.2005      80      90      90      86,7
Федченко        Рустам          ПЗ-1.01 01.01.2000      80      90      85      85,0
Донченко        Данило          ПЗ-1.01 21.09.2004      80      90      70      80,0
Буберенко       Петро   ПЗ-1.01 02.01.2005      60      90      74       74,7
Терещенко       Василь          КТ-1.23 01.02.1999      90      40      69      66,3
Василенко       Микола          ПЗ-1.02 03.04.2002      50      60      60      56,7

Відібрані дані про ювілярів
Прізвище        Ім'я            Група   Дата народж. Вік
Фоменчук        Костянтин       КТ-1.21 02.01.2005      14,9
Федченко        Рустам          ПЗ-1.01 01.01.2000      19,9
Буберенко       Петро   ПЗ-1.01 02.01.2005      14,9

Повнолітні
Прізвище        Ім'я            Група   Дата народж. Оцінка1 Оцінка2 Оцінка3 Середній бал
Федченко        Рустам          ПЗ-1.01 01.01.2000      80      90      85      85,0
Терещенко       Василь          КТ-1.23 01.02.1999      90      40      69      66,3
```

Питання та завдання для самоконтролю

1) Яке призначення функції `fseek(f, 0, 0)`?
2) Як за допомогою функцій `ftell()` та `fseek()` можна визначити сумарний обсяг пам'яті у байтах, який займає файл?
3) Про що свідчить команда `FILE* f = fopen("tmp.dat", "rb")`?
4) Чим попередня команда відрізнятиметься від `FILE* f = fopen("tmp.dat", "ab+")`?
5) Навіщо після попередньої команди потрібна така перевірка:
```
if (f==NULL) {cout<<"Не вдається відкрити файл\n"; return;} ?
```

Завдання до самостійної роботи

1) Дати відповіді на контрольні запитання.
2) Написати мовою С++ програмний код для:
   – створення бінарного файлу з ім'ям Вашого прізвища і розширенням `bin`, заповнення файлу даними, які вибрати з табл. 15.2 (стовпець *Вміст бінарного файлу*);
   – програмного перегляду даних створеного файлу;
   – впорядкування (сортування) даних згідно з параметром, заданим у табл. 15.2;
   – відбирання даних файлу за певною умовою, заданою у табл. 15.2.

2) Створити на комп'ютері програмні проєкт мовою С++ для реалізації написаного програмного коду.

Таблиця 15.2

Індивідуальні завдання

№ вар.	Вміст бінарного файлу	Параметри сортування	Відбір даних за умовою
1	Товари в магазині електроніки: код, назва, фірма-виробник, рік випуску, вартість	За спаданням вартості	Дані про телевізори фірми Samsung
2	Дані про успішність студентів: прізвище та ініціали, дата народження, назва групи, оцінки з фізики, математики, історії	За зростанням оцінок з математики	Студентів, які не здали сесію – мають двійку хоча б з одного предмета
3	Абоненти телефонної станції: ПІБ абонента, номер телефону, дата встановлення телефону, розмір фіксованої абонплати на місяць, сума заборгованості	За спаданням заборгованості	Абонентів із боргом понад 500 грн.

Продовження табл. 15.2

№ вар.	Вміст бінарного файлу	Параметри сортування	Відбір даних за умовою
4	Прайс-лист ноутбуків у магазині: фірма-виробник, тип (ноутбук, ультрабук, нетбук, трансформер тощо), діагональ дисплею у дюймах, ціна, дата поставки, тип дисплея	За зростанням ціни	Ноутбуки типу ультрабук
5	Заявки на ремонт комп'ютерної техніки: найменування комп'ютера, назва ремонту, ПІБ власника, дата взяття на ремонт, кількість днів для ремонтних робіт, вартість ремонту	За спаданням вартості	Передплачені заявки, для яких кількість днів ремонту є меншою семи
6	Список студентів: номер, ПІБ, група, дата народження, рейтинг з історії, математики, основ програмування, кількість пропусків	За спаданням рейтингу з основ програмування	Студенти, які претендують на стипендію: мають середній бал понад 75 балів та не мають пропусків занять
7	Прокат автомобілів: державний номер, марка, кілометраж пробігу, дата здачі у прокат, сума платні	За спаданням кілометражу	Автомобілі Ford
8	Список робітників підприємства: табельний номер, ПІБ, посада, дата народження, розмір зарплатні	За зростанням розміру зарплатні	Співробітників, вік яких до 40 років
9	Дані в ЖЕКу про мешканців: адреса, назва ділянки чи району, ПІБ власника, дата укладення договору про надання послуг, розмір заборгованості	За спаданням розміру заборгованості	Боржників, сума боргу яких понад 500 грн.
10	Список робітників: табельний номер, прізвище та ініціали, дата народження, посада, оклад	За спаданням окладу	Робітники пенсійного віку (понад 60 років)
11	Прайс-лист планшетів у магазині: фірма-виробник, модель, діагональ екрана в дюймах, ціна, дата поставки	За спаданням значень розміру екрана	Планшети Apple
12	Список аварійних будинків у районі міста: вулиця, номер будинку, кількість мешканців у будинку, рік зведення, дата постановки на облік	За зростанням кількості мешканців	Вивести відомості про будинки, побудовані до 1941 року, в яких понад 50 мешканців
13	Список послуг операторів мобільного телефонного зв'язку: номер телефону, назва тарифу, дата ініціалізації пакета, баланс рахунку (грн.)	За спаданням балансу рахунку	Дані про абонентів з балансом рахунку понад 100 грн.

Продовження табл. 15.2

№ вар.	Вміст бінарного файлу	Параметри сортування	Відбір даних за умовою
14	Список підприємств, які виготовляють електронне обладнання: назва підприємства, дата реєстрації підприємства, вид продукції, кількість продукції за останній квартал	За зростанням кількості продукції за останній квартал	Підприємства, які виробляють TV-тюнери
15	Список книг у бібліотеці: інвентарний номер, назва книги, автор, жанр, рік видання, ціна, дата інвентаризації	За зростанням року видання	Книги, у назві яких є "C++"
16	Прайс-лист магазину мобільних телефонів: фірма-виробник, модель, діагональ екрана в дюймах, ціна, дата поставки	За спаданням ціни	Телефони Samsung
17	Послуги турагенції: назва туру, країна, дата виїзду, кількість днів, вартість туру	За зростанням вартості турів	Тури до Польщі
18	Список комп'ютерів у комп'ютерному класі: номер, тип ПК, тактова частота, оперативна пам'ять, ємність диска, дата встановлення	За спаданням значень тактової частоти	ПК, в яких ємність диска не менше 500 Гб
19	Дані про наявність квитків на автостанції: пункт призначення, дата відправлення, час відправлення, ціна, кількість вільних місць	За спаданням кількості вільних місць	Дані про рейси до Одеси
20	Каталог ПК: фірма-виробник, тип (настільний, моноблок, неттоп, ігровий), ємність RAM, ємність диска	За зростанням ємності диска	ПК типу моноблок
21	Список автомобілів в автосалоні: марка, модель, рік випуску, дата реєстрації, пробіг, ціна	За спаданням ціни	Автомобілі марки Mazda
22	Прайс-лист WIFI-адаптерів у магазині: фірма-виробник, модель, швидкість WIFI у Мбіт/с, ціна, дата поставки	За спаданням швидкості	WIFI-адаптери фірми Asus
23	Анкетні дані про співробітників: табельний номер, ПІБ, посада, дата прийому на роботу, сімейний стан, кількість неповнолітніх дітей	За спаданням кількості неповнолітніх дітей	Співробітники-чоловіки, які працюють понад 10 років
24	Список книг у книжковому магазині: назва книги, автор, категорія, дата надходження, ціна	За спаданням ціни	Книги категорії "Програмування"
25	Список команди "Чорноморець": номер гравця, ПІБ, дата народження, амплуа (воротар, захисник тощо), кількість зіграних ігор за минулий сезон	За спаданням кількості зіграних ігор	Дані про нападників команди

Закінчення табл. 15.2

№ вар.	Вміст бінарного файлу	Параметри сортування	Відбір даних за умовою
26	Список принтерів для продажу: тип принтера, фірма-виробник, марка, ціна принтера, дата поставки	За зростанням ціни	Принтери фірми HP з ціною до 4000 грн.
27	Список товарів на складі: назва, категорія, кількість, ціна за одиницю товару, дата завезення на склад	За спаданням ціни	Товари, завезені понад місяць тому
28	Список студентів курсу: ПІБ, дата народження, група, середній бал успішності	За зростанням середнього балу	Студентів із середнім балом понад 80 балів
29	Співробітники підприємства: табельний номер, ПІБ, дата народження, посада, відділ, оклад	За спаданням окладу	Робітники з відділу реалізації
30	Список моніторів: модель, фірма-виробник, діагональ у дюймах, дата поставки	За зростанням розміру діагоналі	Монітори фірми LG

Самостійна робота № 16

Відбір даних бінарного файлу за умовою з формуванням текстового документа

Мета роботи: закріпити практичні навички програмного відбирання даних бінарного файлу та створення на їх основі текстового звітного документа.

Приклад програми

Приклад. У програмному проєкті попередньої самостійної роботи додати можливість програмного відбирання даних за певною умовою та створення на основі відібраних даних текстових звітних документів:

1) студенти, які здали сесію (усі оцінки >= 60);

2) студенти, успішність яких з першого екзамену вище за середню.

Програмний код доданих функцій та зміненої функції main():

```
// Створення звіту (текстового файлу) з даними про студентів, які здали сесію
// (усі оцінки >= 60)
void create_text_file1(char* bname, char* tname)
{
  setlocale(0,".1251");
  student z;
  FILE* fb; FILE* ft;
  // Виклик функції, наведеної у попередній самостійній роботі
  int k= number_of_records(bname);
  // Відкрити бінарний файл для читання
  fb=fopen(bname, "rb");
  if(fb==NULL) { cout<<"Не вдається відкрити файл\n"; return;}
  // Створити текстовий файл для записування
  ft=fopen(tname, "wt");
  if(ft==NULL) { cout<<"Не вдається створити файл\n"; return;}
  setlocale(0,".OCP");
  // Перебирати записи бінарного файлу
  while(fread(&z,sizeof(student), 1, fb)>0)
  {
    // Якщо усі оцінки >= 60, то записати запис у текстовий файл
    if (z.ex1>=60 && z.ex2>=60 && z.ex3>=60)
         fprintf (ft, "%s\t%s\t%s\t%s\t%i\t%i\t%i %5.1f\n",
                  z.surname, z.name, z.group, z.date_birth, z.ex1,
                  z.ex2, z.ex3 , z.avg);
  }
  fclose(ft);
  fclose(fb);
}
```

```
// Створення текстового файлу з даними про студентів,
// успішність яких з першого екзамену вище за середню
void create_text_file2(char* bname, char* tname)
{
  setlocale(0,".1251");
  student z;
  FILE* fb; FILE* ft;
  float sum=0, av;
  fb=fopen(bname, "rb");  // Відкрити бінарний файл для читання
  if (fb==NULL) { cout<<"Не вдається відкрити файл\n"; return;}
  int k= number_of_records(bname);
  // Пройти по файлу й обчислити сумарний бал усіх студентів з першого екзамену
  while (fread(&z,sizeof(student), 1, fb))
     sum += z.ex1;
  av=sum/k;   // Обчислити середній бал з першого екзамену
  cout<<"Середній бал з першого екзамену - "<<av<<endl;
  // Перейти на початок файлу, щоб знову по ньому пройти
  fseek(fb,0,0);
  // Створити текстовий файл для записування
  ft=fopen(tname, "wt");
  if (ft==NULL) { cout<<"Не вдається відкрити файл\n"; return; }
  setlocale(0,".OCP");
  // Перебирати записи бінарного файлу
  while (fread(&z,sizeof(student), 1, fb)>0)
  {
    // Якщо оцінка вища за середню, то записати запис у текстовий файл
     if (z.ex1 >= av)
       fprintf (ft, "%s\t%s\t%s\t%s\t%i\t%i\t%i %5.1f\n", z.surname,
        z.name, z.group, z.date_birth, z.ex1, z.ex2, z.ex3 , z.avg);
  }
  fclose(ft); fclose(fb);
}

// Перегляд текстового файлу
void view_text_file(char* name)
{
  setlocale(0,".1251");
  student z; FILE* f;
  f=fopen(name, "rt");
  if (f==NULL) {cout<<"Не вдається відкрити файл\n"; return;}
  fseek(f,0,2);
  if (ftell(f)>0) // Якщо файл не порожній
  {   fseek(f,0,0);
      char s[150];
      cout<<"\nПерегляд вмісту текстового файлу"<<endl;
      cout<<"Прізвище \tІм'я  \tГрупа  \tДата народж.  Оцінка1
            Оцінка2 Оцінка3 Середній бал\n";
      setlocale(0,".OCP");
```

```
      while(!feof(f))
      {
         fscanf (f, "%s\t%s\t%s\t%s\t%i\t%i\t%i %f\n", z.surname,
                z.name,  z.group, z.date_birth, &z.ex1, &z.ex2,
                &z.ex3, &z.avg);
         printf ("%s\t%s\t%s\t%s\t%i\t%i\t%i\t%5.1f\n", z.surname,
                z.name,  z.group, z.date_birth, z.ex1, z.ex2,
                z.ex3,z.avg);
      }
  }
  fclose(f);
}
int main()
{
  system("color F0");
  setlocale(0,".1251");
  char* bname="stud.dat";
  char* tname1="passed the session.rtf"; // імена файлів
  char* tname2="above average students.rtf";
  add_record(bname);
  // Виведення файлу на екран
  view_file(bname);
  // Сортування даних файлу
  sort(bname);
  //Виведення даних за умовою – ювіляри цього року
  select_data1(bname);
  //Виведення даних за умовою – повнолітні, виповнилось 18 років
  select_data2(bname);
  //Створення звіту (текстового файлу) з даними про студентів, які здали сесію
  setlocale(0,".1251");
  cout<<"\nСтуденти, які здали сесію";
  create_text_file1(bname, tname1);
  // Виведення вмісту текстового файлу
  view_text_file(tname1);
  setlocale(0,".1251");
  cout<<"\nСтуденти, успішність яких з першого екзамену вище за середню\n";
  create_text_file2(bname, tname2);
  // Виведення вмісту текстового файлу
  view_text_file(tname2);
  system("pause>>void");
  return 0;
}
```

Результат виконання програми:

```
Якщо бажаєте дописати дані у файл, натисніть клавішу <y> ->n

Перегляд вмісту файлу
Прізвище        Ім'я            Група   Дата народж. Оцінка1 Оцінка2 Оцінка3 Середній бал
Якименко        Дмитро          КТ-1.21 05.09.2002      90      89      90      89,7
Фоменчук        Костянтин       КТ-1.21 02.01.2005      80      90      90      86,7
Федченко        Рустам          ПЗ-1.01 01.01.2000      80      90      85      85,0
Донченко        Данило          ПЗ-1.01 21.09.2004      80      90      70      80,0
Буберенко       Петро   ПЗ-1.01 02.01.2005      60      90      74       74,7
Терещенко       Василь          КТ-1.23 01.02.1999      90      40      69      66,3
Василенко       Микола          ПЗ-1.02 03.04.2002      50      60      60      56,7

Відсортований за рейтингом файл:
Перегляд вмісту файлу
Прізвище        Ім'я            Група   Дата народж. Оцінка1 Оцінка2 Оцінка3 Середній бал
Якименко        Дмитро          КТ-1.21 05.09.2002      90      89      90      89,7
Фоменчук        Костянтин       КТ-1.21 02.01.2005      80      90      90      86,7
Федченко        Рустам          ПЗ-1.01 01.01.2000      80      90      85      85,0
Донченко        Данило          ПЗ-1.01 21.09.2004      80      90      70      80,0
Буберенко       Петро   ПЗ-1.01 02.01.2005      60      90      74       74,7
Терещенко       Василь          КТ-1.23 01.02.1999      90      40      69      66,3
Василенко       Микола          ПЗ-1.02 03.04.2002      50      60      60      56,7

Відібрані дані про ювілярів
Прізвище        Ім'я            Група   Дата народж. Вік
Фоменчук        Костянтин       КТ-1.21 02.01.2005       14,9
Федченко        Рустам          ПЗ-1.01 01.01.2000       19,9
Буберенко       Петро   ПЗ-1.01 02.01.2005       14,9

Повнолітні
Прізвище        Ім'я            Група   Дата народж. Оцінка1 Оцінка2 Оцінка3 Середній бал
Федченко        Рустам          ПЗ-1.01 01.01.2000      80      90      85      85,0
Терещенко       Василь          КТ-1.23 01.02.1999      90      40      69      66,3

Студенти, які здали сесію
Перегляд вмісту текстового файлу
Прізвище        Ім'я    Група   Дата народж. Оцінка1 Оцінка2 Оцінка3 Середній бал
Якименко        Дмитро          КТ-1.21 05.09.2002      90      89      90      89,7
Фоменчук        Костянтин       КТ-1.21 02.01.2005      80      90      90      86,7
Федченко        Рустам          ПЗ-1.01 01.01.2000      80      90      85      85,0
Донченко        Данило          ПЗ-1.01 21.09.2004      80      90      70      80,0
Буберенко       Петро   ПЗ-1.01 02.01.2005      60      90      74       74,7

Студенти, успішність яких з першого екзамену вище за середню
Середній бал з першого екзамену - 75.7143

Перегляд вмісту текстового файлу
Прізвище        Ім'я    Група   Дата народж. Оцінка1 Оцінка2 Оцінка3 Середній бал
Якименко        Дмитро          КТ-1.21 05.09.2002      90      89      90      89,7
Фоменчук        Костянтин       КТ-1.21 02.01.2005      80      90      90      86,7
Федченко        Рустам          ПЗ-1.01 01.01.2000      80      90      85      85,0
Донченко        Данило          ПЗ-1.01 21.09.2004      80      90      70      80,0
Терещенко       Василь          КТ-1.23 01.02.1999      90      40      69      66,3
```

Питання та завдання для самоконтролю

1) У чому полягає відмінність між бінарними і текстовими файлами?
2) Навести засоби роботи з файлами у C++.
3) У чому полягає специфіка програмного опрацювання файлів у стилі C?
4) В якому заголовному файлі означено тип `FILE*`?
5) Назвати режими відкривання бінарних і текстових файлів у стилі C.

Завдання до самостійної роботи

1) Дати відповіді на контрольні запитання.

2) Написати мовою С++ програмний код функцій для відбирання з бінарного файлу попередньої самостійної роботи даних за умовою, яку вибрати з табл. 16.2, та створити на основі відібраних даних текстовий звітний документ.

3) На комп'ютері у програмному проєкті попередньої самостійної роботи ввести і налагодити розроблений програмний код та впевнитись у правильності його роботи.

Таблиця 16.2

Індивідуальні завдання

№ вар.	Умова на відбір даних з формуванням документа (текстового файлу)
1	Заявки на ремонт комп'ютерної техніки, з дати взяття на ремонт яких минуло понад 20 днів
2	Студенти, яким виповнилось 18 років
3	Абоненти, телефони яких встановлені понад три роки тому
4	Ноутбуки, з дати поставки яких минуло понад чотири місяці
5	Комп'ютери, здані в ремонт понад місяць тому
6	Студенти, які святкують день народження у цьому місяці
7	Автомобілі, здані у прокат понад тиждень тому
8	Робітники підприємства, які святкують у цьому році ювілей (вік кратний 5)
9	Мешканці, з якими договори укладені понад три роки тому
10	Робітники, оклади яких більше середнього арифметичного всіх робітників
11	Планшети з діагоналлю 7″, поставлені у цьому місяці
12	Аварійні будинки, поставлені на облік понад 10 років тому
13	Дані про абонентів, телефони яких ініціалізовані понад рік тому
14	Підприємства, зареєстровані понад 6 років тому
15	Книги, інвентаризовані у поточному календарному році
16	Телефони, з дати поставки пройшло менше місяця
17	Тури, до дати виїзду яких лишилось менше двох тижнів
18	Комп'ютери, встановлені у комп'ютерному класі понад 4 роки тому
19	Автобусні рейси до Вільнюса, які відправляються у найближчі 10 днів
20	Неттопи фірм Acer або Lenovo
21	Автомобілі, зареєстровані минулого місяця
22	WIFI-адаптери, з датою поставки у минулому місяці
23	Сімейні співробітники, які мають неповнолітніх дітей
24	Книги, які надійшли за минулий календарний рік
25	Гравці, старші за 30 років
26	Принтери, з датою поставки у цьому календарному місяці
27	Товари з ціною до 70 грн., кількість яких на складі перевищує 1000
28	Неповнолітні студенти (вік до 18-ти років)
29	Співробітники, яким до пенсії (60 років) лишилося менше 5-ти років
30	Монітори, з датою поставки понад півроку тому

Додаток А
Основні математичні функції

Функція C++	**Опис**
int **abs**(int i)	модуль (абсолютне значення) цілого числа x – $\|x\|$
double **fabs**(double x)	модуль дійсного числа x – $\|x\|$
double **sqrt**(double x)	корінь квадратний – $\sqrt{x}$
double **pow**(double x, double y)	піднесення x до степеня y – x^y
double **exp**(double x)	експонента e^x
double **log**(double x)	натуральний логарифм – $\ln(x)$
double **log10**(double x)	десятковий логарифм – $\lg(x)$
double **cos**(double x)	косинус – $\cos(x)$
double **sin**(double x)	синус – $\sin(x)$
double **tan**(double x)	тангенс – $\mathrm{tg}(x)$
double **acos**(double x)	арккосинус – $\arccos(x)$
double **asin**(double x)	арксинус – $\arcsin(x)$
double **atan**(double x)	арктангенс – $\mathrm{arctg}\ (x)$
double **cosh**(double x)	гіперболічний косинус – $(e^x + e^{-x})/2$
double **sinh**(double x)	гіперболічний синус – $(e^x - e^{-x})/2$
double **ceil**(double x)	округлення доверху: найменше ціле, не менше за x
double **floor**(double x)	округлення донизу: найбільше ціле, не більше за x

Додаток Б
Основні типи даних C++

Тип	Назва	Розмір, байт	Діапазон	Приклади можливих значень	Типи чисел
`char`	символьний (знаковий)	1	–128…127	'a', '\n', '9'	цілі
`unsigned char`	беззнаковий символьний	1	0…255	1, 233	
`short`	короткий цілий	2	–32 768…32 767	1, 153, –349	
`unsigned short`	беззнаковий короткий	2	0…65 535	0, 4, 65 000	
`int`	цілий (знаковий)	4*	–2 147 483 648… …2 147 483 647	–30 000, 0, 690	
`unsigned int`	беззнаковий цілий	4	0…4 294 967 295	2 348, 60 864	
`long`	цілий (знаковий)	4	–2 147 483 648… …2 147 483 647	–30 000, 0, 690	
`float`	дійсний одинарної точності	4	$3.4\cdot10^{-38}$…$3.4\cdot10^{38}$	3.23, –0.2 100.23, 12, –0.947, 0.0001, 6.34e–3, 4e5	дійсні
`double`	дійсний подвійної точності	8	$1.7\cdot10^{-308}$ … …$1.7\cdot10^{308}$		
`long double`	довгий дійсний	10	$3.4\cdot10^{-4932}$… …$1.1\cdot10^{4932}$		
`bool`	логічний	1	false чи true	false(0), true(>=1)	
`enum`	перераховний	2 або 4			
`void`	порожній, без значення				

* – залежно від налагоджень компілятора й апаратних характеристик тип `int` може мати 4 або 2 байти.

Додаток В
Операції мови С++

Таблиця В.1

Операція	Опис	Напрямок
Унарні операції		
`::`	доступ до області видимості	
`.`	вибір елемента через об’єкт	
`->`	вибір елемента через вказівник	⇒
`[ ]`	індекс масиву	
`( )`	дужки, виклик функції	
`<`*тип*`>()`	конструювання	
`++`	постфіксний інкремент	
`--`	постфіксний декремент	
`typeid`	ідентифікація типу часу виконання	
`dynamic_cast`	перетворення типу з перевіркою на етапі виконування	
`static_cast`	перетворення типу з перевіркою на етапі компіляції	
`reinterpret_cast`	перетворення типу без перевірки	
`const_cast`	константне перетворення типу	
`sizeof`	розмір об’єкта чи типу в байтах	
`--`	префіксний декремент	
`++`	префіксний інкремент	
`~`	порозрядне НЕ (заперечення, інверсія)	⇐
`!`	логічне НЕ (заперечення, інверсія)	
`-`	унарний мінус	
`+`	унарний плюс	
`&`	адреса	
`*`	розадресація	
`new`	динамічне виділення пам’яті	
`delete`	динамічне звільнення пам’яті	
`(<`*тип*`>)`	зведення типів	
`.*`	вибір на елемент через об’єкт	
`->*`	вибір на елемент через вказівник	
Бінарні й тернарна операції		
`*`	множення	
`/`	ділення	⇒
`%`	остача від ділення	
`+`	додавання	⇒
`-`	віднімання	
`<<`	порозрядний зсув ліворуч	⇒
`>>`	порозрядний зсув праворуч	
`<`	менше	
`<=`	менше чи дорівнює	⇒
`>`	більше	
`>=`	більше чи дорівнює	

Закінчення табл. В.1

Операція	**Опис**	**Напрямок**
`==` `!=`	дорівнює не дорівнює	⇒
`&`	порозрядна кон’юнкція (І)	⇒
`^`	порозрядне виключне АБО	⇒
`|`	порозрядна диз’юнкція (АБО)	⇒
`&&`	логічне І	⇒
`||`	логічне АБО	⇒
`? :`	умовна операція (тернарна)	⇐

`=`
`*=`
`/=`
`%=`
`+=`
`-=`
`<<=`
`>>=`
`&=`
`|=`
`^=`
`throw`
	присвоювання
	множення з присвоюванням
	ділення з присвоюванням
	остача від ділення з присвоюванням
	додавання з присвоюванням
	віднімання з присвоюванням
	зсув ліворуч з присвоюванням
	зсув праворуч з присвоюванням
	порозрядне І з присвоюванням
	порозрядне АБО з присвоюванням
	порозрядне виключне АБО з присвоюванням
	генерування виняткової ситуації
		⇐

`,`	кома, послідовне обчислювання	⇒
Операції наведено у порядку зменшення пріоритету. Операції з різними пріоритетами розділені рискою		

Список рекомендованої літератури

1. С++. Алгоритмізація та програмування : підручник / О.Г. Трофименко, Ю.В. Прокоп, Н.І. Логінова, О.В. Задерейко. 2-ге вид. перероб. і доповн. – Одеса : Фенікс, 2019. – 477 с.

2. С++. Теорія та практика: навч. посіб. з грифом МОНУ/ [О. Г. Трофименко, Ю. В. Прокоп, І. Г. Швайко, Л. М. Буката та ін.] ; за ред. О. Г. Трофименко. – Одеса : ВЦ ОНАЗ, 2011. – 587 с.

3. С++. Основи програмування. Теорія та практика: підручник / [О. Г. Трофименко, Ю. В. Прокоп, І. Г. Швайко, Л. М. Буката та ін.] ; за ред. О. Г. Трофименко. – Одеса : Фенікс, 2010. – 544 с.

4. Страуструп Б. Язык программирования С++. Специальное издание ; пер. с англ. / Страуструп Б. – М. : ООО "Бином-Пресс", 2015. – 1166 с.

5. Стивен Прата. Язык программирования С++. Лекции и упражнения : учебник ; пер с англ. / Стивен Прата. – СПб.: ООО "ДиаСофтЮП", 2005. – 1104 с.

6. Зиборов В. В. MS Visual C++ 2010 в среде .NET. Библиотека программиста / Зиборов В. В. – СПб. : Питер, 2012. – 320 с.

7. Хортон А. Visual C++ 2010: полный курс.; пер. с англ. / Хортон А. – М. : ООО "И.Д. Вильямс", 2011. – 1216 с.

8. Довбуш Г. Ф. Visual C++ на примерах / Г.Ф. Довбуш, А.Д. Хомоненко ; под ред. проф. А.Д. Хомоненко. – СПб. : БХВ-Петербург, 2007. – 528 с.

9. Дейтел Х. М. Как программировать на С++; пер с англ. / Х. М. Дейтел, П. Дж. Дейтел. – М. : ООО "Бином-Пресс", 2008. – 1456 с.

10. Основи програмування. Базові алгоритми : метод. вказівки для лаб. і практ. робіт / О. Г. Трофименко, Ю. В. Прокоп, І. Г. Швайко, Л. М. Буката. – Ч. 1. – Одеса: ВЦ ОНАЗ ім. О. С. Попова, 2014. – 108 с.

11. Основи програмування. Опрацювання структурованих типів : метод. вказівки для лаб. і практ. робіт / О. Г. Трофименко, Ю. В. Прокоп , І. Г. Швайко, Л. М. Буката. – Ч. 2. – Одеса: ВЦ ОНАЗ ім. О. С. Попова, 2014. – 130 с.

12. Основи програмування. Програмне опрацювання файлів : метод. вказівки для лаб. і практ. робіт / О. Г. Трофименко, Ю. В. Прокоп, І. Г. Швайко, Л. М. Буката. – Ч. 3. – Одеса: ВЦ ОНАЗ ім. О. С. Попова, 2015. – 80 с.

13. Трофименко О. Г. Створення багатомодульних програмних проєктів для опрацювання даних у файлах засобами С++: метод. вказівки для виконання курсової роботи з дисципліни "Основи програмування" / Трофименко О. Г., Прокоп Ю. В. – Одеса: ВЦ ОНАЗ ім. О. С. Попова, 2015. – 44 с.

ЗМІСТ

Навчальне видання

ТРОФИМЕНКО Олена Григорівна
ПРОКОП Юлія Віталіївна
ЗАДЕРЕЙКО Олександр Владиславович

Алгоритмізація та програмування

Навчально-методичний посібник

Підписано до друку 15.02.2020.
Формат 60x84/16. Ум-друк. арк. 18,48.
Зам. № 2002-05.

Видано і віддруковано в ПП «Фенікс»
(Свідоцтво суб'єкта видавничої справи ДК № 1044 від 17.09.02).
Україна, м. Одеса, 65009, вул. Зоопаркова, 25.
Тел. +38 050 7775901 +38 048 7959160
e-mail: fenix-izd@ukr.net
www.fenixbooks.com