МІНІСТЕРСТВО ОСВІТИ І НАУКИ,
МОЛОДІ ТА СПОРТУ УКРАЇНИ
ПРИКАРПАТСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВАСИЛЯ СТЕФАНИКА

В. М. ТКАЧУК

ПРОГРАМУВАННЯ НА C++

ЛАБОРАТОРНИЙ ПРАКТИКУМ

Івано-Франківськ
Прикарпатський національний університет імені Василя Стефаника
2011

УДК 004.41=93(075.8)
ББК 32
Т48

Рекомендовано до друку вченою радою факультету математики та інформатики Прикарпатського національного університету імені Василя Стефаника, (протокол № 2 від 20 жовтня 2011 р.)

Рецензенти:

Горєлов В.О. – кандидат технічних наук, доцент кафедри інформатики факультету математики та інформатики Прикарпатського національного університету імені Василя Стефаника;

Соломко А.В. – кандидат фізико-математичних наук, доцент кафедри математичного і функціонального аналізу факультету математики та інформатики Прикарпатського національного університету імені Василя Стефаника

Ткачук В. М.

Т48 Програмування на С++ : Лабораторний практикум / В. М. Ткачук. – Івано-Франківськ : Видавництво Прикарпатського національного університету імені Василя Стефаника, 2011. – 160 с.

Лабораторний практикум містить опис інтегрованого середовища розробки Microsoft Visual Studio, довідкову інформацію про основні елементи мови програмування С++, приклади розв'язку типових задач у вигляді готових програм з ілюстрацією результату їх роб,оти та набір задач для індивідуального виконання. Тематика робіт відповідає робочій програмі базового курсу «Інформатика і програмування» для студентів 3-го курсу спеціальності «Математика». Практикум орієнтований на студентів, що мають мінімальні навики роботи із сучасними засобами розробки програмного забезпечення й може використовуватися при проведенні лабораторних та практичних занять для студентів математичних і фізичних спеціальностей.

УДК 004.41=93(075.8)
ББК 32

© Ткачук В. М., 2011
© Видавництво Прикарпатського національного університету імені Василя Стефаника, 2011

Лабораторна робота №1

Тема робота: Робота в інтегрованому середовищі розробки Microsoft Visual Studio

Мета роботи: формування навичок організації роботи в середовищі **Microsoft Visual Studio** при написанні та відлагодженні програм на С++

Для виконання роботи необхідно знати:

- прийоми роботи в інтегрованому середовищі **Microsoft Visual Studio**;
- команди головного меню та їх призначення;
- порядок розробки консольного додатку;
- прийоми роботи в текстовому редакторі при введенні та редагуванні програми;
- призначення та можливості використання вбудованого відлагоджувача;
- порядок виконання програми та перегляд результатів її роботи.

Теоретичні відомості

Microsoft Visual Studio являє собою інтегровані в єдину оболонку засоби, що дозволяють створювати, відкривати, переглядати, редагувати, зберігати, компілювати, відлагоджувати та виконувати програми на С++.

Будь-яка програма, що створюється в середовищі **Microsoft Visual Studio,** оформляється у вигляді окремого проекту. Проект – це набір взаємозв'язаних вихідних файлів, призначених для розв'язку певної задачі. Їх компіляція та компоновка дозволяє отримати готову до виконання програму. До проекту входять як файли, що безпосередньо створюються програмістом, так і файли, які створюються та редагуються самим середовищем.

Після запуску **Microsoft Visual Studio** відкривається головне вікно програми, приведене на рис.1 (в залежності від налаштувань його вигляд може дещо відрізнятися від приведеного).

Робочий стіл **Microsoft Visual Studio** складається із трьох вікон:

1. вікно робочої області (*Project Workspace*), де відображається список файлів текучого проекту;
2. вікно редактора (*Editor*) для вводу та редагування вихідного коду на мові С++;
3. вікно виводу (*Output*), куди виводяться повідомлень про хід компіляції та компоновки програми. Зокрема в цьому вікні виво-

дяться всі повідомлення про помилки на етапах компіляції та компоновки програми.

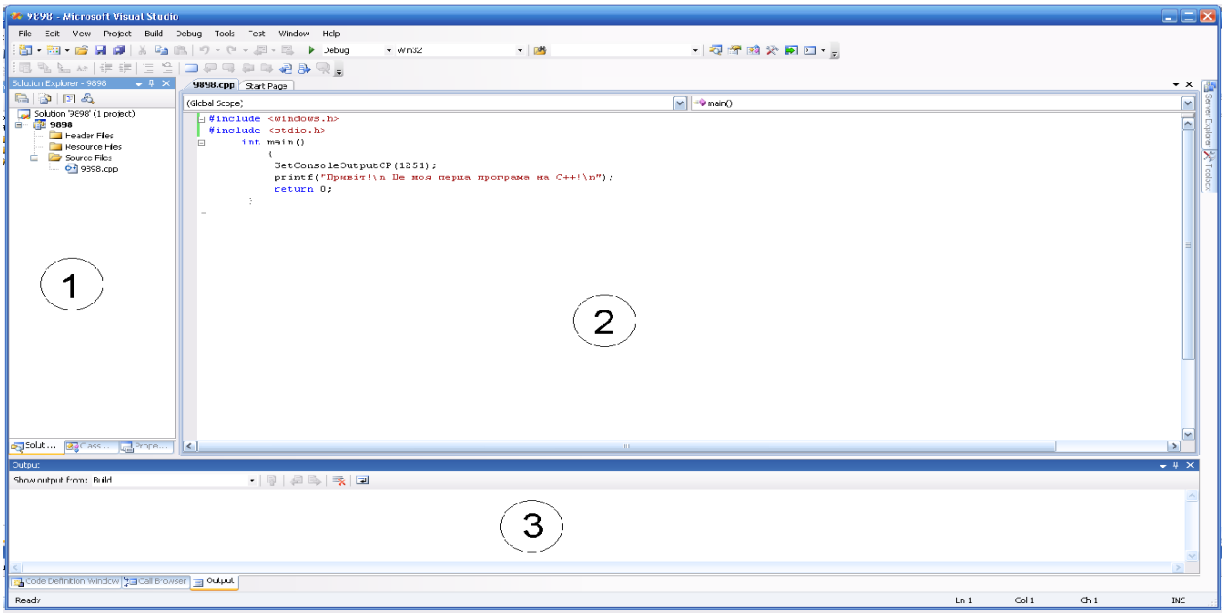
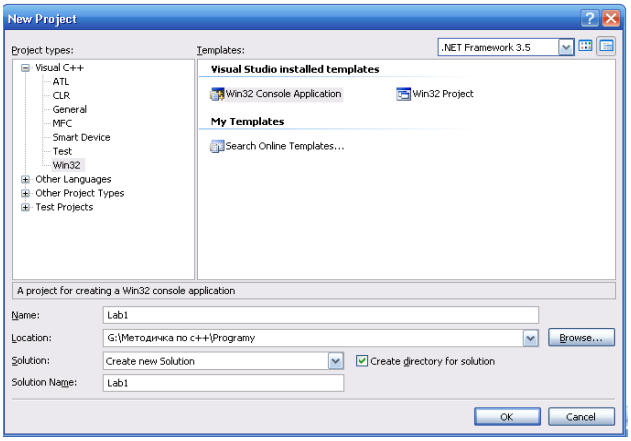


Рис.1. Вигляд вікна середовища **Microsoft Visual Studio** після створення нового проекту.

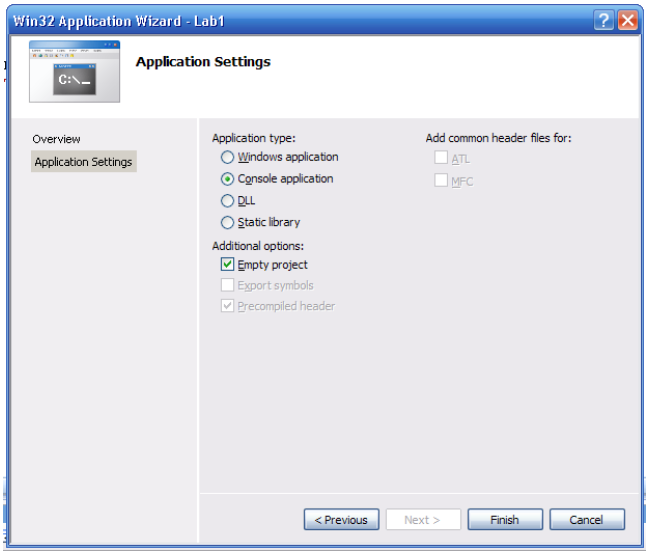
Середовище дозволяє будувати як проекти – віконні *Windows – додатки* із графічним інтерфейсом, так і проекти – *консольні додатки*. Консоль – це інтерфейс, що використовується програмою, яка працює в текстовому режимі. Програма має вхідний (зв'язаний із клавіатурою) та вихідний (зв'язаний з екраном монітора) буфери і взаємодіє з Windows через консоль, для роботи якої створюється вікно, властивості якого аналогічні звичайному вікну Windows. Якраз проект – консольний додаток є найзручнішим для вивчення мови програмування С++.

Для створення нового проекту типу *консольний додаток* необхідно виконати наступні дії:

- у рядку меню головного вікна **Microsoft Visual Studio** вибрати команду *File-New- Projects …*;

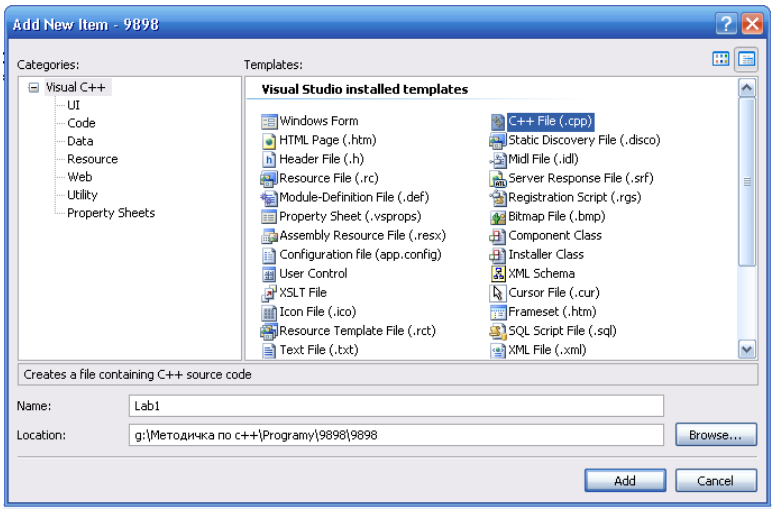


- у діалоговому вікні *New Projects* вибрати тип проекту: *Visual C++ - Win32*;
- задати тип створюваного проекту - *Win32 Console Application*;
- в полі *Project Name* ввести ім'я проекту (наприклад *Lab1*);
- в полі *Locations* вибрати каталог, де він буде зберігатися (діалогове вікно *Browse*);
- клікнути мишкою по кнопці *Ок*;
- у вікні майстра додатків *Win32 Console Application* вибрати пункт меню *Application Settings* та задати порожній проект (*Empty project*);



- клікнути мишкою по кнопці *Finish*.

В результаті буде створено порожній консольний проект. До нього необхідно додати новий, або вже існуючий файл з текстом програми на C++. Для того, щоби додати новий файлу до проекту, можна скористатися командою *Project - Add New Item…* : в полі *Name* ввести ім'я файлу (для зручності бажано задавати ім'я, що співпадає із іменем самого проекту) та клікнути мишкою по кнопці *Add*.



Для того, щоби добавити вже існуючий файл із програмою на С++, необхідно попередньо скопіювати його в робочу папку проекту, скористатися командою *Project - Add Existing Item…* та вибрати відповідний файл.

У папці проекту можуть знаходяться декілька файлів та дві вкладені папки. Файли та папки призначені для:

- файл із розширенням ***.dsw*** – файл проекту, що об'єднує всі файли, які входять до проекту (його може і не бути);
- файл із розширенням ***.dsp*** – для побудови окремого проекту або підпроекту;
- файл із розширенням ***.opt*** – містить всі налаштування даного проекту;
- файл із розширенням ***.ncb*** – службовий файл;
- ***Debug*** – папка, в якій знаходяться файли, створені на етапі компіляції та компоновки програми; виконуваний файл із розширенням ***.exe*** (якщо компіляція та компоновка програми пройшла успішно);
- папка із іменем програми на С++ включає в себе файл із розширенням ***.cpp*** – файл тексту програми на С++.

Компіляцію та компоновку проекту (програми на С++) можна виконати за допомогою меню головного вікна *Build* або за допомогою відповідних кнопок панелі інструментів. Основними командами меню *Build* є:

- *Compile* (***Ctrl+F7***) – компіляція файлу із програмою С++, що є складовою частиною проекту. Повідомлення про помилки на етапі компіляції виводиться у вікно *Output*. Якщо помилок на етапі компіляції не виявлено, то буде створено об'єктний файлу із розширенням ***.obj***;
- *Build Solution* (***F7***) – компоновка проекту. Компілюються всі файли, в яких відбулися зміни з моменту останньої компоновки. Після компіляції проходить компоновка (*link*) всіх об'єктних модулів, в тому числі бібліотечних, та створюється виконуваний файл. Повідомлення про помилки на етапі компоновки виводяться у вікно *Output*. Якщо обидва етапи завершились без помилок, то буде створено виконуваний файл із розширенням ***.exe***, який можна запустити на виконання.

Запустити проект на виконання можна за допомогою меню головного вікна *Debug* (або за допомогою відповідних кнопок панелі інструментів):

- *Start Without Debugging* (***Ctrl+F5***) – старт виконуваного файлу, створеного в результаті компоновки проекту. В результаті відкриється консольне вікно для роботи з програмою.

Для відкриття проекту, над яким ви працювали раніше, виберіть пункт меню *File –Open – Project/Solution…*, знайдіть папку із вашим проектом, а в ній виберіть файл із розширенням ***.dsw***.

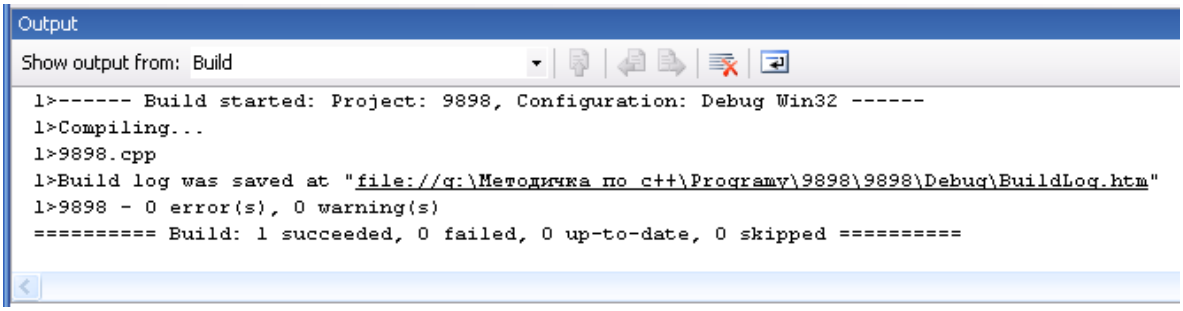
Завдання для виконання

Завдання №1

1. Запустіть інтегроване середовище розробки **Microsoft Visual Studio** та створіть новий консольний додаток. Проект зберігайте на диску у своїй папці. Добавте до проекту новий файл та введіть програму на C++ :

```
#include <stdio.h>
int main()
{
printf("Привіт!\n Це моя перша програма на C++!\n");
return 0;
}
```

2. Скомпілюйте введену програму, задавши відповідну команду. Якщо програма набрана правильно, то у вікні виводу результатів компіляції буде вказано, що помилок (*errors(s)*) та попереджень (*warnings(s)*) немає (стоять нулі) та що створено виконуваний файл (*Build: 0 succeeded, 0 failed, 1 up-to-date, 0 skipped*). Якщо при наборі програми були допущені помилки, то будуть видані відповідні повідомлення. Якщо у вікні виводу (*Output*) вибрати таке повідомлення та натиснути *Enter* (або два рази клікнути лівою клавішею миші), то у вікні програми рядок із помилкою буде відмічено маркером, а курсор буде поміщено у даний рядок. За повідомленням про помилку слідує номер помилки та її короткий опис. Компілятор не завжди може точно локалізує положення помилки, особливо якщо пропущено дужки чи крапки з комою – помилка може знаходитись дещо вище від вказаного місця. Попередження (*warnings(s)*), як правило, не є критичними, а виконуваний файл може бути створений та виконаний (бажано, щоби їх не було).



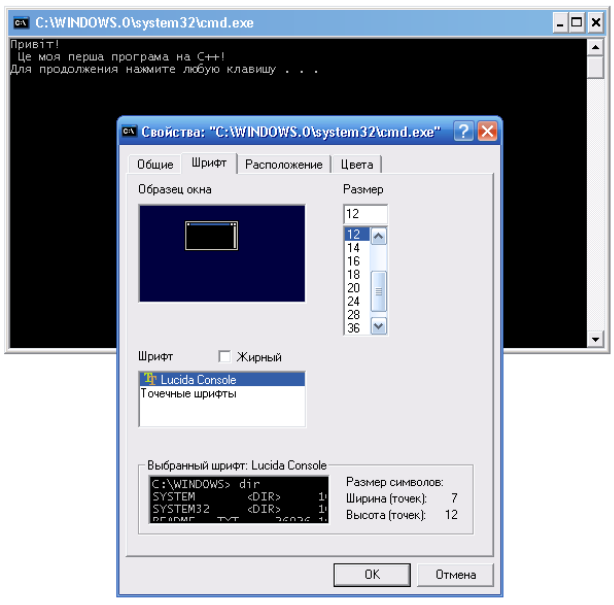
3. Після усунення всіх помилок запустіть програму на виконання. Зверніть увагу на те, як відображається кириличний текст на екрані.
4. Поверніться до режиму редагування програми натиснувши будь-яку клавішу на клавіатурі чи клікнувши мишкою.

Завдання №2

Консольне вікно Windows використовує для виводу інформації кодову сторінку 866, тому для коректного відтворення кириличного тексту на екрані необхідно модифікувати програму наступним чином:

```
#include <windows.h>
#include <stdio.h>
int main()
{
SetConsoleOutputCP(1251);
SetConsoleCP(1251);
printf("Привіт!\n Це моя перша програма на
C++!\n");
return 0;
}
```

Тут добавлено директиву включення файлу ***#include <windows.h>*** та поміняна таблиця кодування для консольного вводу ***SetConsoleCP(1251)*** та консольного виводу ***SetConsoleOutputCP(1251)***. Крім того у консольному вікні необхідно відкрити контекстне меню (права клавіша миші на верхній частині вікна), вибрати пункт меню *Свойства* та у діалоговому вікні, що відкриється, вибрати пункт меню *Шрифт* та задати шрифт *Lucida Console* (див малюнок нижче).



Лабораторна робота №2

Тема роботи: Ввід та вивід інформації в С++. Потокові операції мови С++

Мета роботи: Формування навичок та умінь організації операцій форматного вводу-виводу інформації засобами С++

Для виконання роботи необхідно знати:

- опис змінних стандартних типів;
- використання функцій вводу-виводу для даних стандартних типів;
- операції форматного представлення даних;
- потокові операції мови С++;
- керуючі символи та їх використання;
- структуру програми на мові С++.

Теоретичні відомості

Програма завжди починається директивами препроцесору (***#)***, які обробляється ним до початку компіляції. Задачею препроцесора є доповнення тексту програми бібліотечними функціями чи об'єктами С++, описаними у відповідних заготовочних (*header*) файлах. Найбільш часто використовувані заголовочні файли та їх призначення приведено в додатку №1.

Програма на мові С++ складається із набору окремих функцій. Обов'язковою є функція ***main()*** – вхідна точка програми (місце її розміщення у програмі значення немає):

```
# директиви препроцесора
      . . .
# директиви препроцесора
функція a()
{
тіло функції a
}
функція b()
{
тіло функції b
}
     . . .
void main()
/* функція, з якої починається
```

```
 виконання програми. */
// void – функія не повертає ніяких значень
{
тіло функції
}
```

Текст, що знаходиться між дужками **/*** та ***/** є коментарем до програми та на етапі компіляції ігнорується. Текст, що слідує за **//** є однорядковим коментарем (до кінця даного рядка). Такі коментарі можуть використовуватися в будь-яких місцях програми для її «документування». ***Тіло функції –*** послідовність описів змінних та виконуваних операторів, взятих у фігурні дужки. Кожний опис чи оператор повинен завершуватися крапкою з комою.

Всі змінні, що використовуються в програмі, повинні бути оголошені до їх першого використання (див. додаток №2). Для іменування змінних використовують ідентифікатори. При їх формуванні можуть використовуватися великі та малі букви англійського алфавіту (***A*** та ***a*** сприймаються як різні символи), цифри та символи підкреслення. Ідентифікатор не може починатися із цифри чи містити в собі пробіли. Ідентифікатори також не можуть співпадати із ключовими словами мови С++. Синтаксис оголошення змінних є наступним:

<тип> <імя> = <значення>

де:

<тип> – тип змінної (один із базових);

<імя> – ідентифікатор змінної;

<значення> – значення, яке буде присвоєно змінній після її ідентифікації (не є обов'язковим).

Наприклад:

```
int a;
unsigned int b=2300;
double x, y;
double N_Avag =6.022045e23;
char c1='a', c2 ='A';
int c3 =5+5*2;
const float pi = 3.1415926;
```

Ключове слово ***const*** вказує на величину, яка не міняє свого значення в ході виконання програми.

Кожна змінна має певну область видимості. Якщо змінна оголошена за межами будь-якої функції, то вона називається глобальною і може ви-

користовуватися (є видима) в будь-якому місці програми. Якщо змінна оголошена в програмному блоці (чи деякій функції), то вона є локальною і може використовуватися (видима) тільки в межах даного блоку (функції).

У С++ немає вбудованих засобів для організації вводу-виводу. Вони здійснюються за допомогою функцій вводу-виводу, опис яких міститься міститься у заголовочних файлах ***stdio.h*** та ***iostream***, які підключаються на початку програми за допомогою дерективи препроцесора ***#include***.

При використанні файлу ***stdio.h*** (ввід-вивід у стилі С) для вводу використовується функція

scanf(<список форматів>,<список вводу>)

де:

scanf – ім'я функції форматного вводу;

<список форматів> – специфікатори, що задають формат вводу даних того чи іншого типу (див додаток №3);

<список вводу> – через кому адреси змінних (***&<список вводу>***), в які вводяться дані із клавіатури. Кількість специфікаторів форматів та їх тип повинні відповідати кількості та типу адрес змінних, значення яких необхідно ввести. Символ ***&*** означає унарну операцію одержання адреси змінної. Для вводу значень рядкових змінних ***&*** не використовується.

Ввід символьних даних із клавіатури здійснюється за допомогою наступних функцій:

getchar() – із відображенням введеного символу на екрані;

getch() – без відображення введеного символу на екрані.

Наприклад:

```
#include < stdio.h> //підключення файлу
void main()
{
int k,m;     //опис змінних
scanf(" %d%d",&k,&m);  //ввід даних
}
```

Вивід даних здійснюється за допомогою функції ***printf***:

printf(<список форматів>,<список виводу>)

Функція ***printf*** в якості ***<список форматів>*** використовує ті ж формати, що і функція вводу ***scanf*** (див додаток №3). Додатково можна використати наступні керуючі символи (escape-символи):

\n – перехід на новий рядок;

\r – повернення на початок рядка;

\v – вертикальна табуляція;

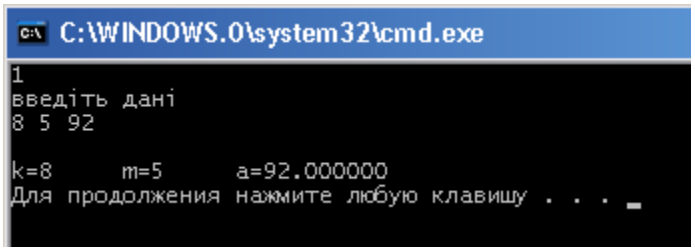
\t –горизонтальна табуляція.

В ***<список виводу>*** через кому задаються ідентифікатори змінних, значення яких виводяться. Для виводу на екран символу служить функція ***putchar()***.

Приклад програми, що ілюструє використання операцій вводу-виводу:

```
#include<stdio.h>
#include <windows.h>
void main()
        {
char ch='1';
int k,m;            //опис змінних
float a;             //опис змінних
SetConsoleOutputCP(1251);
SetConsoleCP(1251);
putchar(ch);        //вивід символа на екран
printf("\nведіть дані\n"); //вивід даних на екран
scanf("%i%i%f",&k,&m,&a );      //ввід даних
printf("\nk=%d\tm=%d\ta=%f\n",k,m,a); /*вивід
даних на екран*/
    }
```

Результат роботи програми:



Операції вводу - виводу можна також реалізувати із використанням бібліотеки потокового вводу - виводу ***iostream*** (функції вводу - виводу з використанням класів С++):

ввід даних із клавіатури: ***cin>><ідентифікатор змінної>***;

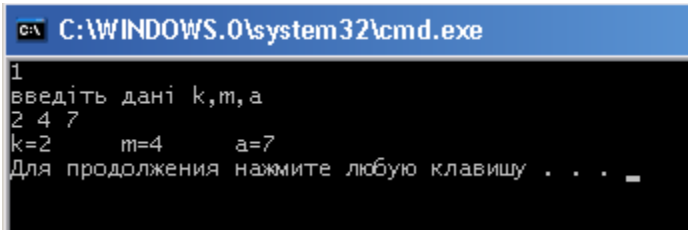
вивід даних на екран: ***cout<< <вираз>.***

тут ***<вираз>*** **-** ідентифікатор змінної, рядок символів або арифметичний вираз.

Для включення у потік символу нового рядка (еквівалентний **\n**) служить ***endl***. Приклад програми, що ілюструє використання операцій потокового вводу - виводу :

```
#include<iostream>
#include <windows.h>
using namespace std;
void main()
{
char ch='1';
int  k,m;              //опис змінних
float a;               //опис змінних
SetConsoleOutputCP(1251);
SetConsoleCP(1251);
cout << ch;          //вивід символа на екран
//ввід підказки для вводу даних
//та перехід на новий рядок
cout <<endl<< "введіть дані k,m,a"<<endl;
    cin >> k >> m >> a ;  //ввід даних
//вивід даних та перехід на новий рядок
cout << "k=" << k << "\tm=" << m << "\ta=" << a
<<endl;
        }
```

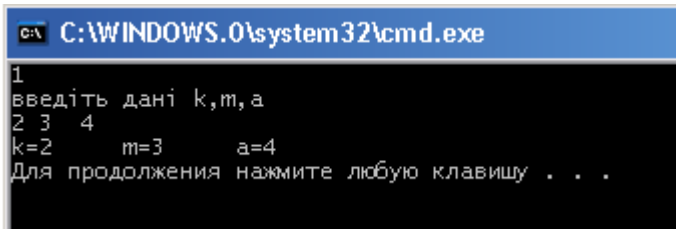
Результат роботи програми:



Примітка: *деректива* ***using namespace std*** *вказує, що ми будемо працювати із іменами зі стандартної бібліотеки. При відсутності такої дерективи замість* ***cin*** *необхідно писати* ***std::cin,*** *а замість* ***cout*** *-* ***std::cout****. Попередній приклад при відсутності такої дерективи матиме вигляд (тут також замість потокового символ нового рядка* ***endl*** *використано керуючий символ функції* ***printf*** *переходу на новий рядок* ***\n*** *):*

```
#include<iostream>
#include <windows.h>
void main()
    {
char ch='1';
int  k,m;            //опис змінних
float a;             //опис змінних
SetConsoleOutputCP(1251);
SetConsoleCP(1251);
std::cout << ch;     //вивід символа на екран
std::cout << "\nведіть дані k,m,a\n";
//вивід даних на екран
std::cin >> k >> m >> a ;      //ввід даних
std::cout <<"k="<< k << "\tm="<<m<<"\ta="<< a <<"\n";
//вивід даних на екран
}
```

Результат роботи програми:



Потокові операції вводу - виводу можуть бути пов'язані не тільки із клавіатурою та екраном, а і із зовнішніми файлами. Для цього в командному рядку виклику програми (***exe*** – файл) необхідно задати імена відповідних файлів. Наприклад:

- ***lab2.exe < dani.in*** – при виконанні програми ***lab2.exe*** дані вводяться не з клавіатури, а читаються з файлу ***dani.in***;
- ***lab2.exe > dani.out*** – при виконанні програми ***lab2.exe*** дані виводяться не на екран, а в файл ***dani.out***;
- ***lab2.exe < dani.in > dani.out*** – при виконанні програми ***lab2.exe*** вхідні дані читаються із файлу ***dani.in***, а результат виводиться у файл ***dani.out***.

Завдання для виконання

Написати програми, що вводять-виводять змінні всіх стандартних типів (див. додаток №2). Перша програма – із використанням дерективи пре-

процесору ***#include <stdio.h>***, а друга – із використанням директиви ***#include <iostream>***. Використати також операції форматного виводу даних та перенаправлення потоків для роботи із файлами.

Примітка: *якщо на виконання запускається* ***exe*** *- файл, то для можливості перегляду результатів виконання програми необхідно зупинити закриття консольного вікна командою* ***system("pause")****, добавивши її в кінці програми!!!*

Лабораторна робота №3

Тема роботи: Лінійні програми. Обчислення арифметичних виразів та математичних функцій

Мета роботи: Формування навичок та умінь програмування арифметичних виразів, обчислення математичних функцій та написання простих лінійних програм

Для виконання роботи необхідно знати:

- алфавіт мови С++;
- типи змінних та їх опис;
- математичні операції та їх пріоритет;
- оператори вводу-виводу;
- стандартні математичні функції та правила їх використання;
- операції присвоєння;
- структура програми на мові С++.

Теоретичні відомості

Оператор присвоєння **=** дозволяє замінити значення операнду, що стоїть зліва від знаку рівності, значенням виразу, обчисленим справа від нього. Синтаксис оператора присвоєння:

<ідентифікатор змінної>=<вираз>

Допустимим є запис виду:

<ідентифікатор змінної1>=<ідентифікатор змінної2>=<вираз>

Основні математичні операції С++ приведено в додатку №4. У складних виразах порядок виконання операцій визначається дужками та пріоритетом операцій. Можна використовувати декілька рівнів вкладення дужок: обчислення проходить від внутрішніх дужок до зовнішніх. Порядок виконання операцій у виразі має значення при наявності декількох операцій із різним пріоритетом: операції із однаковим пріоритетом виконуються раніше операцій із нижчим пріоритетом незалежно від того, де вони знаходяться у виразі. Операції виконуються зліва направо в порядку їх пріоритету від найвищого до найнижчого:

1. виклик математичних функцій;
2. **++, --**;
3. ***, /, %**;
4. **+, -**;
5. операції присвоєння: **=, +=, -=, *=, /=, %=**.

В операторі присвоєння змінні, що входять до складу виразу, повинні бути одного типу. Якщо вони різного типу, то необхідно виконати перетворення змінних до одного типу. При цьому перетворення проводиться так, щоби змінні, що займають менший об'єм оперативної пам'яті були перетворені до типу змінних, що займають більший об'єм. Наприклад:

```
int a;
float y,b;
y=a+b;
```

У операторі присвоєння ***y=a+b*** необхідно писати:

```
y=float(a)+b.
```

Операція ***float(a)*** називається операцією приведення типів: вона перетворює змінну цілого типу до дійсного типу. При виконанні операції приведення типів може відбуватися втрата інформації. Наприклад, в результаті виконання приведеного нижче франменту програми змінна ***k*** прийме значення 4.

```
double a=4.24;
int k;
k=int(a);
```

Для «зв'язування» декількох виразів використовується операція слідування **,** (кома). Вирази, розділені між собою комами, обчислюються зліва направо. Наприклад:

```
a=4, b=b+a+5, c=b/5;
```

Математичні функції знаходяться в бібліотеці ***math***, яка підключається на початку програми деректівою препроцесора ***#include <math.h>*** (див. додаток №5). Необхідно памятати, що всі тригонометричні функції в С++ працюють із кутовим величинами, заданими в радіанах.

Приклади виконання завдання

Приклад №1

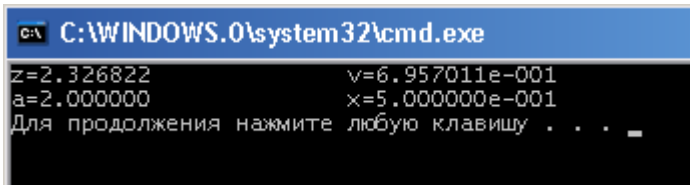
Написати програму обчислення арифметичного виразу та форматного виводу результату на екран. Сталі величини, що використовуються при обчисленні арифметичного виразу, задати як константи. На екран вивести a, v, x, z.

$v = \sin(\sqrt[3]{2z+1}) - 2.34,\ z = \left|2 - x \cdot \cos(a^2)\right|$ при $a = 2,\ x = 0.5$.

```
#include<stdio.h>
#include <math.h>
#include <windows.h>
```

```
// Задання значення константи a
#define a 2.0
// Задання значення константи x
#define x 0.5
void main()
{
double v, z;
SetConsoleOutputCP(1251);
SetConsoleCP(1251);
//обчислення значення z
z=fabs(2-x*cos(a*a));
//обчислення значення v
v=sin(pow(2*z-1,2/3.0));
printf("z=%f\t\tv=%e\n",z,v);
printf("a=%f\t\tx=%e\n",a,x);
}
```

Результат роботи програми:



Зверніть увагу, що для присвоєння константам a та x початкового значення використано дерективу препроцесора **`#define`**. Ця деректива використовується для заміни всіх наступних входжень констант (ключових слів, операторів, виразів) заданим значенням. Тип константи визначається компілятором за типом значення, заданого дерективою. Аналогічний результат отримається в при виконанні наступної програми:

```
#include<stdio.h>
#include <math.h>
#include <windows.h>
void main()
{
double v, z;
// Задання значення константи a
const double a=2.0;
// Задання значення константи x
const double x=0.5;
SetConsoleOutputCP(1251);
```

```
SetConsoleCP(1251);
//обчислення значення z
z=fabs(2-x*cos(a*a));
//обчислення значення v
v=sin(pow(2*z-1,2/3.0));
printf("z=%f\t\tv=%e\n",z,v);
printf("a=%f\t\tx=%e\n",a,x);
}
```

Приклад №2

Написати програму обчислення арифметичного виразу та виводу результату на екран за допомогою потокових операцій вводу-виводу.

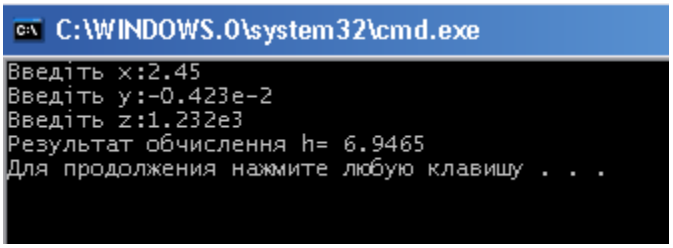
$$h=\frac{x^{2y}+e^{y-1}}{1+x\left|y-tg(z)\right|}+10\cdot\sqrt[3]{x}-\ln(z)$$

При $x=2.45,\ y=-0.423\cdot10^{-2},\ z=1.232\cdot10^{3}$ результат обчислення рівний $h=6.9465$.

```
#include <iostream>
#include <math.h>
#include <windows.h>
using namespace std;
int main()
{
double x,y,z,a,b,c,h;
SetConsoleOutputCP(1251);
SetConsoleCP(1251);
cout << "Введіть x:";
cin >>  x;
cout << "Введіть y:";
cin >>  y;
cout << "Введіть z:";
cin >>  z;
a=pow(x,2*y)+exp(y-1);
b=1+x*fabs(y-tan(z));
c=10*pow(x,1/3.)-log(z);
h=a/b+c;
cout << "Результат обчислення h= "<<h<<endl;
return 0;
}
```

Примітка: *для простоти та усунення можливих помилок обчислення* **h** *розбито на два етапи: вихідний вираз розбивається на три підвирази (***a, b,c***), а потім обчислюється результуюче значення* **h=a/b+c**.

Результат роботи програми:



Завдання для виконання

Завдання №1

При написанні програми використати форматний вивід та пояснювальну текстову інформацію для зручного візуального сприйняття результатів.

1. Обчислити $b=\sin^3(x)-a, z=\left|1-\sqrt{a}\cdot\cos(b)\right|$ при $a=2,\ x=0.5$. На екран вивести a, x, b, z.
2. Обчислити $y=x^4-\sqrt[3]{c}, d=2y+\cos(c)$ при $x=3, c=2.5$. На екран вивести x, c, y, d.
3. Обчислити $f=\ln(3x)-h, e=\sqrt[5]{f^3}$ при $x=1.6,\ h=1.6$. На екран вивести x, h, f, e.
4. Обчислити $d=\dfrac{1}{(z-2a)^2}-z^2,\ f=\sqrt[3]{d^2}$ при $z=3.6, a=2.6$. На екран вивести z, a, d, f.
5. Обчислити $h=\dfrac{1}{n}-e^n\sin(p)\quad y=\dfrac{h}{|h+1|}$ при $n=2,\ p=0.5$. На екран вивести n, p, h, y.
6. Обчислити $r=\sin(x)+\ln(s), r_1=\sqrt[2]{r^3}$ при $x=0.6, s=5.6$. На екран вивести x, s, r, r_1.
7. Обчислити $m=\sin(x)+\cos(x), n=\sqrt[2]{m}+\sin(x)$ при $x=0.3$. На екран вивести x, m, n.

8. Обчислити $p = 3.62\ln(x+2.3)$, $k = p^3$ при $x = 1.6$. На екран вивести x, p, k.

9. Обчислити $t = \dfrac{\sqrt{x}+\sqrt{y}}{e^x}$, $z = 1+t$ при $x = 0.5, y = 4.4$. На екран вивести x, y, t, z.

10. Обчислити $w = \left|1-\sin(2x)+n\right|$, $t = 2w + \dfrac{1}{7^3}$ при $x = 0.9, n = 1.6$. На екран вивести n, x, w, t.

11. Обчислити $y = \sqrt{t^3 - 1} + 3.31$, $k = \sin(2y + 1.21)$ при $t = 2.7$. На екран вивести t, y, k.

12. Обчислити $t = \sqrt[3]{x} - e^{(x+1)} - 10.4$, $r = 3t - \dfrac{1}{2}$ при $x = 1.24$. На екран вивести x, t, r.

13. Обчислити $n = \dfrac{1}{\left(x^3 - e^{1/x}\right)}$, $m = n^{\frac{3}{2}}$ при $x = 0.8$. На екран вивести x, n, m.

14. Обчислити $f = 0.45z^5 + \dfrac{1}{z}, z = 1 + \dfrac{1}{x^2}$ при $x = 1.1$. На екран вивести x, f, z.

15. Обчислити $d = \dfrac{3\sin(x) - 1.1w}{2\cos(x+2.14)}$, $l = 1 + \sin(d)$ при $x = 2.6$, $w = pi/2$, $pi = 3.1415$. На екран вивести x, w, d, l.

16. Обчислити $y = \dfrac{a_1}{b_1{}^2 + \sqrt{a_1}}$, $a_1 = arctg(b_1)$ при $b_1 = 0.72$. На екран вивести b_1, a_1, y.

17. Обчислити $y = xtg\left(\dfrac{x}{2.65}\right)$, $z = 1 + \dfrac{\ln(|y|)}{3}$ при $x = 6.145$. На екран вивести x, y, z.

18. Обчислити $l=\dfrac{e^{2x-1.3}}{\sqrt[3]{|x|+2}}$, $a=1+\dfrac{l}{3}$ при $x=-2.1$. На екран вивести x,l,a.

19. Обчислити $c=\sin(\cos(x)-1)e^2$, $y=1+\ln(3+|y|)$ при $x=-0.61$. На екран вивести x,c,y.

20. Обчислити $d=\dfrac{\sin(x)+\cos(y)}{\ln(|x|)}$, $c_1=tg(d)$ при $x=5.1, y=-2.25$. На екран вивести x,y,d,c_1.

21. Обчислити $t=\sqrt{\left|ab^2-\dfrac{1}{b}\right|e^{a+b}}$, $y=1+t+\ln(3)$ при $a=0.76, b=-2.2$. На екран вивести a,b,t,y.

22. Обчислити $t=\dfrac{x\sin^5(x)-\cos(x)}{\cos(x+3.23)+7}$, $y=1+\dfrac{1}{t^2+1}$ при $x=1.78$. На екран вивести x,t,y.

23. Обчислити $s=\dfrac{x^2+\sqrt[3]{x+1}}{x^3+3.43}$, $r=2.45s^2+s$ при $x=3.5$. На екран вивести x,s,r.

24. Обчислити $k=\dfrac{3(\sin(l)+2.345)}{4+\ln(5+l)}$, $r_a=k^2+2$ при $l=4$. На екран вивести l,k,r_a.

25. Обчислити $p=\dfrac{arctg(a)}{\sqrt{a^2+2x}}$, $y=1+\dfrac{p}{p^2+1}$ при $x=2.891, a=4.1$. На екран вивести x,a,p,y.

26. Обчислити $f=0.4x^3\ln(|x|)+\sqrt{1+\cos(2x)}$, $g=\dfrac{\sin(f+3.1415)}{2+0.1}$ при $x=4.44$. На екран вивести x, f, g.

27. Обчислити $c=\sin(\sin(x))$, $y=1+\dfrac{e^{2c}}{c^2+1}$ при $x=4.13$. На екран вивести x,c,y.

28. Обчислити $y = tg(\sin(bx)) + \frac{1}{x + \sqrt[4]{b}}$, $s = 1 + \frac{1}{y^2 + 3.12}$ при $x = 2.1,\ b = 3.23$. На екран вивести x, b, y, s.

29. Обчислити $k = \frac{b^2 + \sin(a^2)}{a^2 + b^4}$, $w = 1.1b^2 + \frac{1}{8^2} - k$ при $b = 0.47,\ a = 3$. На екран вивести a, b, k, w.

30. Обчислити $f = t^3 \ln(t) + 2.5 \cdot 6.83, t = 1.1z^2 + \frac{1}{8^2}$ при $z = 1.4$. На екран вивести z, f, t.

Завдання №2

Обчислити значення виразу при заданих вихідних даних, які в програмі задати як константи. Отриманий результат порівняти із вказаним правильним. При написанні програми використати потокові операції вводу-виводу та пояснювальну текстову інформацію для зручного візуального сприйняття результатів.

1. $s = \frac{2\cos\left(x - \frac{2}{3}\right)}{\frac{1}{2} + \sin^2(y)} \left(1 + \frac{z^2}{3 - \frac{z^2}{5}}\right)$ при $x = 14.26,\ y = -1.22,\ z = 3.5 \cdot 10^{-2}$.

   Результат обчислення $s = 0.749155$.

2. $x = \frac{\sqrt{\sin(wt + \varepsilon)} - e^{-wt}}{\sqrt[3]{\ln(2k + d) + d^{3k}}}$ при $t = 1.6,\ w = -3.4,\ \varepsilon = -1.5 \cdot 10^{-3},\ k = 2.1,$ $d = 3.2$. Результат обчислення $x = -229.579$.

3. $s = \frac{\sqrt[3]{9 + (x - y)^2}}{x^2 + y^2 + 2} - e^{|x-y|} tg^3(z)$ при $x = -4.5,\ y = 0.75 \cdot 10^{-4},$ $z = -0.845 \cdot 10^2$. Результат обчислення $s = -3.23765$.

4. $y = \frac{(arctgx^3 + \cos\sqrt{x})^{2x}}{h^x + \ln\left|2{,}4x^3\right|} - a$ при $x = 1.45,\ a = -5.89,\ h = 4.1$. Результат обчислення y=6.30082.

5. $q=\dfrac{1+\sin^2(x+y)}{\left|x-\dfrac{2y}{1+x^2y^2}\right|}x^{|y|}+\cos^2\left(arctg\left(\dfrac{1}{z}\right)\right)$ при $x=3.74\cdot10^{-2}$, $y=-0.825$, $z=0.16\cdot10^{2}$. Результат обчислення $q=1.05534$.

6. $y=\left(\sqrt{\dfrac{ax+b}{c+dx}}+\sqrt{arctgx}\right)^{2/3}-e^{2x}$ при $x=1.6$, $a=3.3$, $b=7.245$, $c=6.4$, $d=-1.45\cdot10^{-1}$. Результат обчислення $y=-22.7215$.

7. $w=\left|\cos(x)-\cos(y)\right|^{\left(1+2\sin^2(y)\right)}\left(1+z+\dfrac{z^2}{2}+\dfrac{z^3}{3}+\dfrac{z^4}{4}\right)$ при $x=0.4\cdot10^{4}$, $y=-0.875$, $z=-0.475\cdot10^{-3}$. Результат обчислення $w=1.98727$.

8. $y=\sqrt[3]{\left(\dfrac{ax}{b+cx}+tgx\right)^2}-e^{x^2}$ при $x=4.22\cdot10^{-1}$, $a=1.1$, $b=2.45$, $c=1.3$. Результат обчислення $y=-2.80621$.

9. $k=\ln\left(y^{-\sqrt{|x|}}\right)\left(x-\dfrac{y}{2}\right)+\sin^2\left(arctg(z)\right)$ при $x=-15.246$, $y=4.642\cdot10^{-2}$, $z=21$. Результат обчислення $k=-182.038$.

10. $z=\dfrac{1}{1-\dfrac{1+e^{tx}}{\dfrac{r}{x}(j+x^2)}}$ при $x=3.004$, $t=0.89$, $r=2.05$, $j=3.0$. Результат обчислення $z=1.00111$.

11. $v=\sqrt{10\left(\sqrt[3]{x}+x^{y+2}\right)}\left(\arcsin^2(z)-|x-y|\right)$ при $x=16.55\cdot10^{-3}$, $y=-2.75$, $z=0.15$. Результат обчислення $v=-40.6307$.

12. $f=\dfrac{\sqrt{8+|x-6|^2+b}}{\ln x+2}+e^{x^k}(\ln x+2)$ при $x=1.7$, $b=0.5$, $k=2.0$. Результат обчислення $f=47.5873$.

13. $r=5arctg(x)-\dfrac{1}{4}\arccos(x)\dfrac{x+3|x-y|+x^2}{|x-y|z}+x^2$ при $x=0.1722$, $y=6.33$, $z=3.25\cdot10^{-4}$. Результат обчислення $r=-205.306$.

14. $y=\frac{1}{2}\cdot\ln\left|\frac{a+\sin x^2}{b-\cos^2 x}\right|$ при $x=3.7$, $a=33.01$, $b=1.25\cdot10^2$. Результат обчислення $y=-0.649392$.

15. $h=\frac{e^{|x-y|}|x-y|^{x+y}}{arctg(x)-arctg(z)}+\sqrt[3]{x^6+\ln^2(y)}$ при $x=-2.235\cdot10^{-2}$ $y=2.23$, $z=15.221$. Результат обчислення $h=39.3741$.

16. $f=\frac{e^a+\ln|bx|+5}{1+\ln\sqrt{|dx|}+e^{x^2}}$ при $x=1.34$, $a=1.1$, $b=-2.55\cdot10^1$, $d=3.98$. Результат обчислення $f=1.46761$.

17. $q=\left|x^{\frac{y}{x}}-\sqrt[3]{\frac{y}{x}}\right|+(y-x)\frac{\cos(y)-\frac{z}{(y-x)}}{1+(y-x)^2}$ при $x=1.825\cdot10^2$ $y=18.225$, $z=-3.298\cdot10^{-2}$. Результат обчислення $q=1.21308$.

18. $b=x+\frac{\sin^2|y|}{3/5+\cos^2 z^2}$ при $x=8.5\cdot10^{-1}$ $y=-15.2$, $z=-1.4\cdot10^{-1}$. Результат обчислення $b=0.9979$.

19. $s=2^{-x}\sqrt{x+\sqrt[4]{|y|}}\sqrt[3]{e^{x-1/\sin(z)}}$ при $x=3.981\cdot10^{-2}$ $y=-1.625\cdot10^3$, $z=0.512$. Результат обчислення $s=1.26185$.

20. $y=\frac{3x^2+25e^{x^2}}{|x^7|+\sqrt{ax^2+2}}+\ln|x+k|$ при $x=2.2$ $a=1.2\cdot10^1$, $k=1.76$. Результат обчислення $y=13.7262$.

21. $k=y^{\sqrt[3]{|x|}}+\cos^3(y)\frac{|x-y|\left(1+\frac{\sin^2(z)}{\sqrt{x+y}}\right)}{e^{|x-y|}+\frac{x}{2}}$ при $x=6.251$ $y=0.827$, $z=25.001$. Результат обчислення $k=0.712122$.

22. $y=\frac{\sqrt{a_0+a_1x+a_2x^2}}{\sqrt[3]{a_2|\sin x|}}$ при $x=5$ $a_0=2$, $a_1=5.0$, $a_2=3$. Результат обчислення $y=1.1012$.

23. $s = 2^{(y^x)} + (3^x)^y - \dfrac{y\left(arctg(z) - \dfrac{1}{3}\right)}{|x| + \dfrac{1}{y^2+1}}$ при $x = 3.251$ $y = 0.325$, $z = 0.466 \cdot 10^{-4}$. Результат обчислення $s = 4.23655$.

24. $f = \sqrt[5]{\dfrac{(a+b)^2}{c+d} + e^{\sqrt{x+1}}}$ при $x = 2$, $a = 3.2$, $b = -4.1 \cdot 10^{-4}$, $c = 1.46$, $d = -1.3 \cdot 10^{-1}$. Результат обчислення $f = 1.62996$.

25. $s = \dfrac{\sqrt[4]{y + \sqrt[3]{x-1}}}{|x-y|\left(\sin^2(z) + tg(z)\right)}$ при $x = 17.421$ $y = 10.365 \cdot 10^{-3}$, $z = 0.828 \cdot 10^{5}$. Результат обчислення $s = 0.330564$.

26. $h = \dfrac{1}{a}\left(\dfrac{1+x^2}{b-x} + \dfrac{c}{2} tg\, x\right)$ при $x = 3.7$ $a = -1.5$, $b = 2.52 \cdot 10^{1}$, $c = -2.8$. Результат обчислення $h = 0.12758$.

27. $a = \dfrac{y^{x+1}}{\sqrt[3]{|y-2|} + 3} + \dfrac{x + \dfrac{y}{2}}{2|x+y|}(x+1)^{-1/\sin(z)}$ при $x = 12.3 \cdot 10^{-1}$ $y = 15.4$, $z = 0.252 \cdot 10^{3}$. Результат обчислення $a = 82.8256$.

28. $f = e^{x \cos \frac{\pi}{4}} \cos(z \sin \dfrac{\pi}{4})$ при $x = 2.3$ $\pi = 3.14$, $z = -1.21$. Результат обчислення $f = 3.33821$.

29. $s = \dfrac{x^{y+1} + e^{y-1}}{1 + x|y - tg(z)|}(1 + |y-x|) + \dfrac{|y-x|^2}{2} - \dfrac{|y-x|^3}{3}$ при $x = 2.444$, $y = 0.869 \cdot 10^{-2}$, $z = -0.13 \cdot 10^{3}$. Результат обчислення $s = -0.498707$.

30. $f = a_0 \cdot \sqrt{|\ln(b - 2x\cos\dfrac{\pi}{3} + x^2 \ln(d+x))|}$ при $x = 3$ $a_0 = 3.1 \cdot 10^{1}$, $b = -1.2$, $\pi = 3.1415$, $d = 4.22$. Результат обчислення $f = 48.4613$.

Лабораторна робота №4

Тема роботи: Оператори розгалуження

Мета роботи: Формування навичок та умінь розробки розгалужених програм для організації альтернативних обчислень

Для виконання роботи необхідно знати:

- умовний оператор ***if*** … ***else***;
- умовний оператор ***if*** ;
- оператор множинного вибору ***switch***;
- логічні вирази та логічні операції;
- умовна операція?;
- складений оператор.

Теоретичні відомості

Оператор умовного переходу служить для спрямування ходу обчислення по одній із можливих віток обчислення в залежності від істинності або хибності деякої умови. Він має дві форми запису. Синтаксис повного умовного оператора переходу:

if(<умова>) <оператор1>
else <оператор2>

Якщо ***<умова>*** істина (***true***), то виконується ***<оператор1>***, якщо хибна (***false***), виконується ***<оператор2>***. В якості ***<умови>*** можуть використовуватися операції відношення або логічні операції (див. додаток №6), арифметичний вираз чи просто ідентифікатор. Якщо як умова використовується арифметичний вираз, то коли він не рівний нулю (***true***) виконується ***<оператор1>***, а коли рівний нулю (***false***) - ***<оператор2>***.

Неповна форма запису умовного оператора, коли альтернативна вітка ***else*** відсутня:

if(<умова>) <оператор>

Якщо ***<умова>*** істинна, то виконується ***<оператор>***, а якщо хибна, то ***<оператор>*** пропускається і виконується наступний оператор.

В якості ***<оператора>*** може бути як простий, так і складений оператори. Складеним називають два чи більше простих операторів, які виконуються як єдине ціле. Для створення складеного оператора використовують операторні дужки ***{…}***. Транслятор сприймає складений оператор як єдине ціле. Наприклад:

```
{
pi=3.1415;
y=fabs(log(b-2*x*cos(pi/3.0)+x*x*log(d+x)));
y=a*sqrt(y);
cout << "Результат h= "<<y;
}
```

При необхідності перевірки виконання декількох умов можна скористатися вкладеними логічними операторами. Наприклад для перевірки одночасного виконання двох умов (***a<b*** та ***b<c***) можна скористатися наступним вкладенням логічних операторів:

```
if(a<b) if(b<c)
```

Для об'єднання двох і більше логічних виразів простіше використовувати логічні операції (див. додаток №6), оскільки вкладення логічних операторів може приводити до логічних помилок та поганої «читабельності» програми. Наприклад:

(a<b && b<c) – одночасне виконання двох умов;

(a<b || b<c) – виконання хоча б однієї із умов;

(a<b ! b<c) – виконання першої та невиконання другої умови;

(a<b && b<c ! b<c) – одночасне виконання двох перших умов та невиконання третьої.

У випадку, коли варіантів вибору є декілька, оптимальніше використовувати оператор вибору ***switch***. Він має наступний синтаксис запису:

```
switch (<вираз>)
{
case n1: <оператор 1>;
        break;
case n2: <оператор 2>;
        break;
     .  .  .  .
case nn: <оператор n>;
        break;
default: <оператор n+1>;
}
```

<Вираз> обчислює значення цілого типу і може приймати одне із значень міток ***case***. Якщо отримане значення не співпадає із жодним значенням міток, то виконується мітка ***default***. Щоби в кожному конкретному випадку виконувався тільки оператор відповідної мітки обов'язковим є використання команди ***break,*** яка передає управління в

кінець оператора ***switch*** (виконуються тільки оператори відповідної відки).

Для реалізації альтернативних обчислень в залежності від виконання певної умови може також використовуватися умовна операція **?** :

<Умова> ? <Вираз1> : <Вираз2>

Спочатку обчислюється значення виразу ***<Умова>***. Якщо вона істина, то обчислюється значення ***<Виразу1>***, а якщо ні - ***<Виразу2>***. Наступний приклад ілюструє використання умовної операції для знаходження більшого із двох чисел ***x*** та ***y***:

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

Приклади виконання завдання

Приклад №1 (використання оператора умовного переходу)

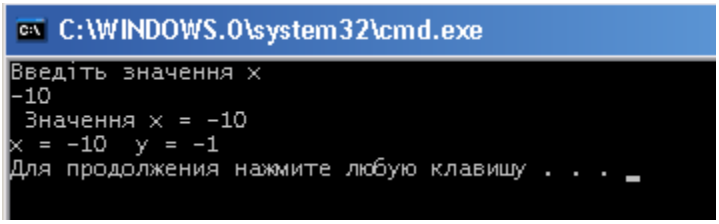
Програма обчислення функції, значення якої рівне:

$$y(x)=\begin{cases}-1 \ при \ x<0\\ 0 \ при \ x=0\\ +1 \ при \ x>0\end{cases}$$

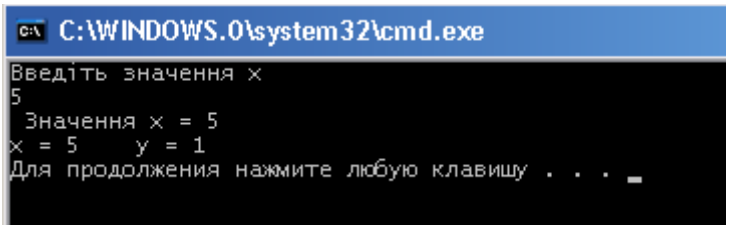
```
#include <iostream>
#include <windows.h>
using namespace std;
void main ( )
{
int x,y;
SetConsoleOutputCP(1251);
SetConsoleCP(1251);
cout << "Введіть значення x  \n";
cin >>x;
cout << " Значення x = "  << x;
if (x < 0) y = -1; else if (x==0) y=0; else y = 1;
cout << "\nx = " << x << "\t " << "y = " << y<<"\n";
}
```

Результати роботи програми:

Введено значення ***x=-10***



Введено значення ***x=5***



Приклад №2 (використання оператора вибору ***switch***)

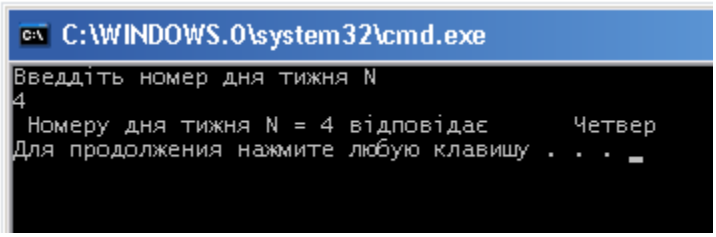
За введеним із клавіатури номером дня тижня програма виводить його назву. Якщо введеному номеру не відповідає жоден із днів тижня, то видається відповідне повідомлення.

```
#include <iostream>
#include <windows.h>
using namespace std;
void main ( )
{
int i;
SetConsoleOutputCP(1251);
SetConsoleCP(1251);
cout << "Введдіть номер дня тижня N \n";
cin >>i;
cout << " Номеру дня тижня N = "  << i<< "
відповідає ";
switch (i)
{
case 1: cout << "\tПонеділок \n";
break;
case 2:cout << "\tВівторок \n";
break;
case 3:cout << "\tСереда \n";
break;
case 4:cout << "\tЧетвер \n";
break;
case 5:cout << "\tПятниця \n";
break;
case 6:cout << "\tСубота \n";
break;
case 7:cout << "\tНеділя \n";
break;
```

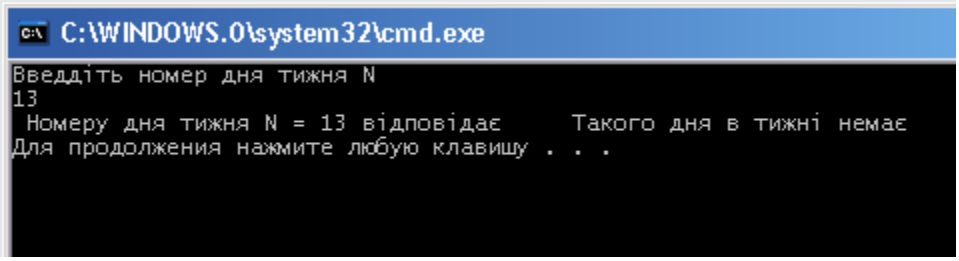
```
default: cout << "\tТакого дня в тижні немає \n";
    }
    }
```

Результати роботи програми:

Введено значення ***N=4***



Введено значення ***N=13***



Завдання для виконання

Завдання №1

Написати програму обчислення значення функції. Значення аргументу вводиться із клавіатури у діалоговому режимі. Значення інших змінних задати у вигляді констант. Передбачити вивід на друк вихідних даних та результатів у зручній для сприйняття формі з використанням форматного виводу.

1. $$y(x)=\begin{cases}\cos^2(bx), & x<a\\ \sqrt{(x-a)^3}, & a\le x\le b\\ \left|\dfrac{ax-2b}{b^2+a^2}\right|, & x>b\end{cases}$$

2. $$y(x)=\begin{cases} e^{|x+a|}\cdot\sin(x), & x<a \\ x^{a+b}, & a<x<b^2 \\ (x-1)^2\cos(x)^2, & x\geq a^2 \end{cases}$$

3. $$y(x)=\begin{cases} \cos\ (ax), & x<a \\ \sqrt{(x^4+a)}, & a\leq x\leq b \\ \left|\frac{1}{b^2+a^2}\right|, & x>b \end{cases}$$

4. $$d(\alpha)=\begin{cases} a^2\sin(\alpha+\pi/3), & \alpha<b \\ 1, & b\leq\alpha\leq a+1 \\ b\cos^2(\alpha^2), & \alpha>a+1 \end{cases}$$

5. $$y(x)=\begin{cases} \sqrt[5]{\cos^4(x-a)}, & x\leq a \\ x^2 tg(bx), & a<x\leq b \\ \left|\frac{\sin(a+b)x}{x^2-b^2}\right|, & x>b \end{cases}$$

6. $$f(k)=\begin{cases} \frac{k+b}{2+\sin^2(a)}, & k<2a \\ k+\sin(k^2), & 2a\leq k<2a+1 \\ \sqrt{a+4}\cdot\sin(a+b), & k\geq 2a+1 \end{cases}$$

7. $$y(x)=\begin{cases} \sqrt[3]{e^{ax}}, & x\leq a \\ a\cos(bx), & a<x<b \\ \frac{a+bx}{x^2-b^2}, & x\geq b \end{cases}$$

8. $$g(h)=\begin{cases} 2\cos^2(h+a)^2, & b+1<h<a \\ tg(a), & a\leq h\leq a+b \\ \ln(b+3\sin(h^2)), & h>a+b \end{cases}$$

9. $$y(x)=\begin{cases} a\sin(x^3), & x<a \\ e^{a+b}(x+1), & a<x\le b \\ \sqrt{(x^2+b^2)}, & x>b \end{cases}$$

10. $$o(q)=\begin{cases} \ln^2(a^2+2q), & q>a+b \\ \ln(q+2a), & q=a+b \\ \dfrac{b^2+2}{2}+\sin^3(q), & q<a+b \end{cases}$$

11. $$y(x)=\begin{cases} \cos^2(bx), & x<a \\ \sqrt{(x-a)}, & a\le x\le b \\ \left|\dfrac{ax-2b}{(x^2+b^2)}\right|, & x>b \end{cases}$$

12. $$p(x)=\begin{cases} e^{\frac{x}{2}+\sqrt{x}}, & a\le x \\ e^{\frac{x}{2}}, & a<x\le a+2b \\ \ln^2(b+x)+tg\left(x+\dfrac{\pi}{3}\right), & x>a+2b \end{cases}$$

13. $$s(p)=\begin{cases} \dfrac{a^2-p}{\log_2(b+p)}, & p<3 \\ \sqrt{p}, & 3<p\le 5 \\ \dfrac{\sqrt{p^3}}{a-1}, & p>5 \end{cases}$$

14. $$t(r)=\begin{cases} \dfrac{a+\sqrt[3]{r}}{\ln^2(r+b)}, & r\ge a \\ \sqrt{ae^r}, & b<r<a \\ \dfrac{\sqrt{a+e^r}}{r}, & r\le b \end{cases}$$

15. $k(t)=\begin{cases}\sqrt[5]{b\sin(t+\pi/6)}, & t<a\\ \cos\left(t+\dfrac{\pi}{2}\right), & a\le t<b\\ \sqrt{\dfrac{a}{2}\cos\left(t+\dfrac{\pi}{4}\right)}, & t\ge b\end{cases}$

16. $\varphi(c)=\begin{cases}\lg^2(a), & b\ge c>a\\ \lg\left(a^{|c-b|}\right), & b<c\\ \ln\left(b+c^2a\right), & c\le a\end{cases}$

17. $\gamma(x)=\begin{cases}\sqrt[4]{|x-2a|}+b, & x\le 0\\ |x-a|, & 0<x\le|ab|\\ \dfrac{|x-a|^3}{3}, & x>|ab|\end{cases}$

18. $c(\varphi)=\begin{cases}atg(\varphi-b/4), & \varphi>a\\ 0, & a\ge\varphi\ge a+b\\ btg\left(\dfrac{\varphi+b}{3}\right), & \varphi<a+b\end{cases}$

19. $d(a)=\begin{cases}e^{\sqrt{\alpha+b}}+b\sqrt{a}, & \alpha>2b\\ b\sqrt{a}+a\ln(b), & a\le\alpha\le 2b\\ e^{\sqrt{\alpha+b}}+a\ln(b), & \alpha<a\end{cases}$

20. $y(i)=\begin{cases}b+\ln^2|i+\pi|, & i<\text{-}b\\ \sin(i), & 0>i\ge\text{-}b\\ i^2+\ln^2|i+2\pi|, & 0\le i\end{cases}$

21. $y(k)=\begin{cases} ak^2+b, & k<0 \\ \dfrac{k-a}{k-c}, & a\le k\le 2|a+c| \\ \left|\dfrac{ak-2}{b^2+a^2}\right|, & k>2|a+c| \end{cases}$

22. $g(x)=\begin{cases} \dfrac{1}{bx}+b, & x+5<b \\ \dfrac{x-a}{x}, & b\le x+5\le |a+c| \\ \left|\dfrac{2x}{c^2-4}\right|, & x+5>|a+c| \end{cases}$

23. $f(s)=\begin{cases} as^2+bs+c, & s<b \\ \dfrac{-a}{s+c}, & b\le s\le a+c \\ a(s+c), & s>a+c \end{cases}$

24. $f(s)=\begin{cases} as^2+bs^3-c, & s<2b \\ \dfrac{a}{2s+c}, & 2b\le s\le (a+c)^2 \\ a^2(2s+4c), & s>(a+c)^2 \end{cases}$

25. $y(x)=\begin{cases} a-\dfrac{x}{10+b}, & x<0 \\ \dfrac{x-a}{2x+c}, & 0\le x\le |a+c+b| \\ 3x+4(a+b-c), & x>|a+c+b| \end{cases}$

26. $v(k)=\begin{cases} ak^2+b^2x-c, & x<-a \\ \dfrac{x-a}{x+c}, & -a\le x<0 \\ 3x-c, & x\ge 0 \end{cases}$

27. $f(x)=\begin{cases} -ax^2-bx+c, & x<-|ab| \\ \dfrac{x-a}{xc}, & -|ab|\le x\le 0 \\ 3x-c, & x>0 \end{cases}$

28. $y(x)=\begin{cases}\cos(cx), & x<2a\\ \sqrt{(x-a)^3}, & a\le x<3b\\ \left|\dfrac{a+x-2b}{(x+b^2)}\right|, & x\ge b\end{cases}$

29. $d(r)=\begin{cases}\dfrac{a+\sqrt[3]{r^2}}{\ln^2(r+cb)}, & r\ge a+c\\ \sqrt{ae^r}, & a+b+c\le r<a+c\\ \dfrac{\sqrt{a+e^r}}{r}, & r<b+a+c\end{cases}$

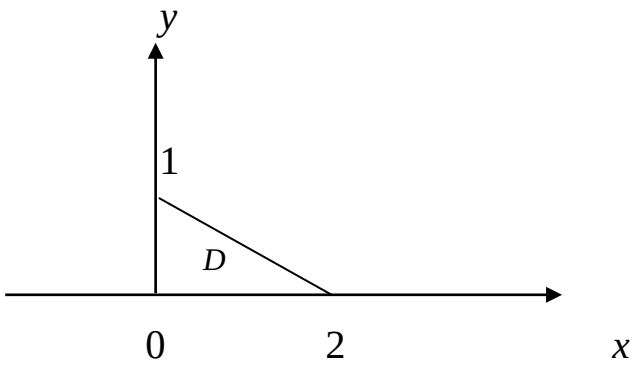
30. $s(h)=\begin{cases}\cos^2(bh^2), & x\le 0\\ \sqrt{(4x+3a)}, & 0\le x<ba\\ \left|\dfrac{ax-2}{b+a}\right|, & x\ge ba\end{cases}$

Завдання №2

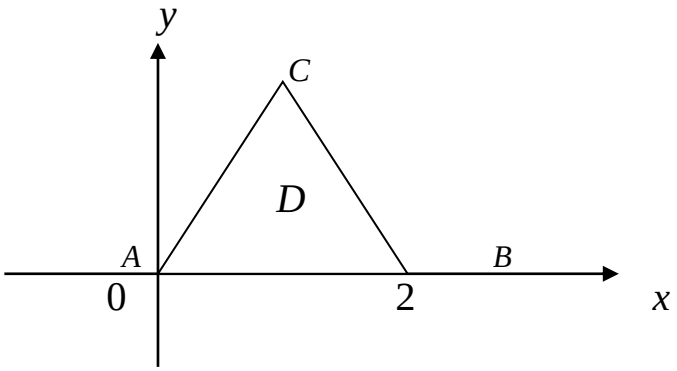
Розв'язати приведену нижче задачу. Вхідні дані вводиться із клавіатури у діалоговому режимі. Інші дані, необхідні для розв'язку задачі, задаються у вигляді констант. Передбачити вивід на друк результатів у зручній для сприйняття формі, використовуючи форматний вивід.

1. Написати програму, яка вводить вік користувача і, якщо йому більше 18 років, повідомляє, що він має право голосу. В іншому разі вона обчислює, через скільки років користувач буде мати право голосу.
2. Дано координати двох точок. Визначити, чи розміщені вони на одному колі із центром в початку координат.
3. Із клавіатури вводиться номер місяця в році. Виводиться повідомлення про назву місяця та пору року. Якщо місяця із таким номером не існує – виводиться відповідне повідомлення.

4. Дано радіус кола та довжина сторони квадрата. Перевірити, чи пройде квадрат крізь коло.
5. Ввести із клавіатури три числа та знайти найменше та найбільше серед цих чисел.
6. Дано радіус кола та довжина сторони квадрата. Визначити, чи пройде круг через квадрат.
7. На площині задано коло радіусу R із центром в початку координат. Ввести із клавіатури координати точки та визначити, чи знаходиться вона на колі, чи в середині кола, чи за його межами.
8. Дано координат и точки $M(x, y)$. Визначити, чи належить вона замкнутій області D.



9. Дано координати точки $M(x, y)$. Визначити, чи належить точка замкнутій області D. ΔABC – рівносторонній.



10. Дано координати точки $M(x, y)$. Визначити, чи належить точка замкнутій області D, заданій системою обмежень:

$$\begin{cases} x + y \le 1, \\ y \ge 0, \\ 2x - y \le 1 \end{cases}$$

11. Відома два кути трикутника φ_1 та φ_2. Визначити тип трикутника: гострокутний, прямокутний, тупокутний.

12. Дано довжини трьох відрізків. Визначити, чи можна із цих відрізків побудувати трикутник.
13. Дано довжини трьох сторін трикутника. Визначити, якого типу це є трикутник (прямокутний, гострокутний, рівносторонній, рівнобедрений …).
14. Знайти найбільше із трьох значень $f(x)$, $f(2x)$ та $f(3x)$, де $f(x)=\sin(x)$. Значення аргументу вводиться із клавіатури у градусах, тому перед обчисленням функції перевести їх у радіани.
15. Прямокутник задано координатами його вершин. Визначити, чи належить введена точка області прямокутника.
16. Прямокутник задано координатами його вершин. Визначити, чи коло із заданим радіусом та координатами центру повністю належить області прямокутника.
17. Ввести значення x та розмістити в порядку спадання результати обчислення $x\sin(x)$, $\ln(\sqrt{x})$, $\sqrt[5]{x^2}$.
18. Трикутник задано координатами вершин. Визначити, в яких квадрантах координатної площини він знаходиться.
19. Визначити, чи перетинаються два відрізки, задані координатами своїх кінців.
20. Два кола задано їхніми радіусами та координатами центрів. Визначити кількість спільних точок у цих кіл (одна, дві, чи жодної).
21. Із клавіатури вводиться порядковий номер місяця. Програма виводить назву пори року. Передбачити перевірку коректності вводу номера місяці.
22. Дано координати точки $M(x,y)$. Визначити, якому квадранту системи координат належить дана точка.
23. Дано параболу виду $y=ax^2+bx+c$ та координати двох точок. Визначити, як знаходяться ці точки відносно віток параболи (обидві між вітками, обидві зовні віток чи інші варіанти).
24. Дано параболу виду $y=ax^2+bx+c$ та координати двох точок. Визначити, яка із точок знаходиться ближче до параболи.
25. Дано дві прямі лінії $ax+by+c=0$ та $a_1x+b_1y+c_1=0$. Визначити, чи перетинаються вони.
26. Із клавіатури вводиться п’ять цілих чисел. Визначити кількість парник чисел та кількість чисел, кратних 3.

27. Коло задано радіусом та координатами центру. Визначити, як лежить початок координат відносно кола – в середині, на колі чи за його межами.
28. Із клавіатури вводиться оцінка в п’ятибальній системі. На екран виводиться відповідна оцінка в університетській системі: відмінно, добре, задовільно або незадовільно.
29. Дано три числа (в межах від 1 до 9 як константи). Із клавіатури вводяться три довільні числа, а на екран виводиться повідомлення про кількість вгаданих чисел.
30. Дано пряма $ax+by+c=0$ та парабола $y=a_1x^2+b_1x+c_1$. Визначити кількість точок перетину прямої з параболою: дві, одна, жодної.

Лабораторна робота №5

Тема роботи: Прості цикли із відомим числом повторів

Мета роботи: Формування навиків в розробці простих циклічних програм із відомим числом повторень для реалізації типових алгоритмів обробки даних.

Для виконання роботи необхідно знати:

- поняття циклічного обчислення, призначення та порядок його використання;
- оператор циклу ***for***: правила та особливості його використання;
- організацію циклічних обчислень за допомогою логічного оператора;
- оператор ***continue*** та ***break***: призначення та способи використання;
- оператор безумовного переходу;
- базові алгоритми обробки даних з використанням циклічних обчислень.

Теоретичні відомості

Оператор циклу ***for*** забезпечує циклічне (повторне) виконання деякого простого або складеного оператора певне число разів. Синтаксис оператора ***for*** є наступним:

```
for(<ініціалізація>; <умова>; <вираз>)
    <оператор>
```

де:

<ініціалізація> – змінна або вираз, що управляє виконанням циклу;

<умова> – перевірка умови досягнення кінця циклічних обчислень;

<вираз> – міняє значення лічильника циклу і може бути будь яким арифметичним виразом допустимого типу;

<оператор> – тіло циклу – будь який простий чи складений оператор.

У заголовку циклу ***for*** одна чи декілька компонент може бути відсутня, але при цьому обов’язково мусять бути збережені всі крапки з

комою (*;*). В ініціалізації початкового значення може стояти або один вираз, а декілька, розділених комою. При перевірці умови на закінчення циклічного обчислення можна використати декілька умов, об'єднаних логічними операціями. ***<Вираз>*** також може містити не одну змінну, а декілька, розділені між собою за допомогою коми.

Алгоритм роботи оператора ***for*** є наступний:

1. виконується початкова ініціалізація;
2. перевіряється умова, і якщо вона ***false***, то виконання циклу завершується, інакше управління передається пункту 3;
3. виконується тіло циклу - ***<оператор>***;
4. обчислюється ***<вираз>*** та управління передається пункту 2.

For може мати і наступну форму запису (нескінчений цикл):

for (; ;) < оператор >

Для виходу із такого нескінченого циклу обов'язковим є перевірка в тілі циклу виконання певної умови та завершення циклічного обчислення за допомогою ***break***. Він виконує негайний вихід із циклу та передачу управління наступному оператору. Наприклад:

```
/* Вивід парних чисел на проміжку від 0 до 100 */
    #include <stdio.h>
    void main()
    {
    int i=0;
    for(;;)
    {
    i++;
    if (i%2 == 1)
    continue;
    else
    printf("\n %ld", i );
    /* Перевірка умови завершення циклічного
    обчислення */
    if (i == 100) break;
    }
    printf("\n");
    }
```

Для передачі управління (виконання циклу) на наступну ітерацію в межах циклічного обчислення служить оператор ***continue***. Він перериває виконання даної ітерації циклічного обчислення та передає управління наступній ітерації. Наприклад:

```
    /* Вивід парних чисел на проміжку від 0 до
100 */
    #include <stdio.h>
    void main()
    {
    int i;
    for(i=1;i<=100;i++)
    if (i%2 == 1)
    continue;
    else
    printf("\n %1d", i );
    printf("\n");
    }
```

Для безумовної передачі управління використовується оператор переходу ***goto*** :

goto < мітка>

Керування передається оператору з даною міткою ***<мітка>: оператор***, а сама мітка в програмі не декларується. В якості мітки може виступати будь-який звичайний ідентифікатор. Область дії мітки обмежується функцією, в якій вона визначена, тому неможливо передати управління у іншу функцію.

Приклади виконання завдання

Приклад №1 (використання оператора циклу)

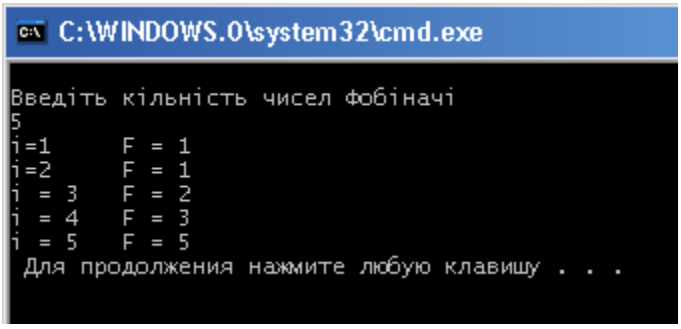
Написати програму обчислення n чисел Фобіначі, що визначаються рекурентним співвідношенням $a_1 = 1,\ a_2 = 1,\ a_i = a_{i-1} + a_{i-2}$.

```
    #include <iostream>
    #include <windows.h>
    using namespace std;
    void main ( )
    { int n,a1,a2,a;
    SetConsoleOutputCP(1251);
    SetConsoleCP(1251);
    a1=1;
    a2=1;
cout << "\nВведіть кільність чисел фобіначі  \n";
    cin >>n;
```

```
    cout << "i=1" << "" << "\t" << "F = " << a1;
cout << "\ni=2" << "" << "\t" << "F = " << a2;
    for(int i=3;i<=n;i++)
    {
    a=a2+a1;
    a2=a1;
    a1=a;
    cout << "\ni = " << i << "\t" << "F = " << a;
    }
    cout << "\n ";
    }
```

Результат роботи програми:

Введено значення ***N=5***



Приклад №2 (розв'язок попередньої задачі з використанням оператора умови та безумовного переходу)

```
    #include <iostream>
    #include <windows.h>
    using namespace std;
    void main ( )
    {
    int i,n,a1,a2,a;
    SetConsoleOutputCP(1251);
    SetConsoleCP(1251);
    i=2;
    a1=1;
    a2=1;
cout << "\nВведіть кільність чисел фобінаві  \n";
    cin >>n;
    cout << "i=1" << "" << "\t" << "F = " << a1;
cout << "\ni=2" << "" << "\t" << "F = " << a2;
    q:i=i+1;
```

```
a=a2+a1;
a2=a1;
a1=a;
cout << "\ni = " << i << "\t" << "F = " << a;
if (i<n) goto q;
cout << "\n ";
}
```

Завдання для виконання

Написати програму для розв'язку приведеної нижче задачі. В програмі передбачити використання форматного виводу та пояснювальної текстової інформації. Розв'язок реалізувати двома способами: у першому використати оператор циклу **`for`**, *а у другому організувати циклічне обчислення за допомогою оператора* **`if`**.

1. Обчислити суму $S = \sum_{i=1}^{m} 1/i^3$. Значення m вводиться із клавіатури.
2. Обчислити суму $S = \sum_{n=1}^{m} (-1)^n n^2$, де $m-$ ціле число, що вводиться із клавіатури.
3. Обчислити суму $S = \sum_{n=1}^{k} 2^n$, де k – ціле число, що вводиться із клавіатури.
4. Обчислити приблизно площу фігури, обмеженої графіками функцій $y(x) = x^2$, $x = 2$ та $y = 0$, розбивши інтервал зміни x на 100 частин та сумуючи площі прямокутників із основою, рівною інтервалу зміни x та висотою, що визначається значенням функції в середині основи.
5. Для введеного з клавіатури натурального числа n обчислити добуток $1 \cdot 3 \cdot ... \cdot n$, якщо n – непарне та $2 \cdot 4 \cdot ... \cdot n$, якщо n - парне.
6. Обчислити суму $S = \sum_{k=1}^{m} 1/k!$. Значення m вводиться із клавіатури.
7. Обчислити $n-$ний член ряду, елементи якого обчислюються за формулами: $a_1 = 2; a_2 = 1; a_i = a_{i-1} + 2a_{i-2}$ для $i > 2$.
8. Обчислити кількість розміщень із n по m $A_n^m = n(n-1)...(n-m-1)$.
9. Написати програму обчислення для заданого n суми $1^2 + 2^2 + 3^2 + ... + n^2$ та перевірити правильність отриманого результату (сума рівна $n(n+1)(2n+1)/6$).

10. Обчислити суму $S=\sum_{i=1}^{n}(-1)^i\cdot 2^i$. Значення n вводиться із клавіатури.

11. Обчислити суму $S=\sum_{i=1}^{n}(-1)^i\cdot(2i)$. Значення i вводиться із клавіатури.

12. Для заданого n обчислити добуток $P=1\cdot 3\cdot 5\cdot 7\cdot ...\cdot(2n+1)$.

13. Обчислити суму $S=1\cdot 3+3\cdot 5+5\cdot 7+...+(2n-1)(2n+1)$ для заданого n.

14. Обчислити добуток $P=(1+3)\cdot(5+7)\cdot ...\cdot((2n-1)+(2n+1))$ для заданого n.

15. Обчислити суму ряду $S=1+(1+2)+(2+3)+(3+4)+...+((n-1)+n)$ для введеного із клавіатури значення n.

16. Протабулювати функцію $y=x^3$ при зміні x на проміжку від a до b (вводяться із клавіатури), розбивши його на 25 частин.

17. Обчислити значення інтегралу $\int_0^{\pi/2}\sin(x)dx$ за формулою трапецій із заданим числом розбиттів n.

18. Протабулювати функцію $y=\sin^2(x)$ на проміжку від a до b, розбивши його на 30 частин.

19. Обчислити значення інтегралу $\int_0^5\sqrt{(x^2+1)}dx$ за формулою трапецій із числом розбиттів $n=100$.

20. Обчислити суму $S=\sum_{i=1}^{n}\frac{3^i}{i!}$. Значення n вводиться із клавіатури.

21. Обчислити значення інтегралу $\int_1^4\sqrt{(x^3+4)}dx$ за формулою трапецій із заданим числом розбиттів n.

22. Обчислити суму 120 членів ряду, $i-$тий член якого визначається формулою $a_i=\frac{x^{2i+1}}{(2i+1)!}$. Значення x вводиться із клавіатури.

23. Обчислити суму 100 членів ряду, $i-$тий член якого визначається формулою $a_i=(-1)^{i+1}\frac{x^{2i}}{2i(2i-1)}$. Значення x вводиться із клавіатури.

24. Із клавіатури вводиться послідовність $n=15$ цілих чисел. Знайти суму парних елементів цієї послідовності.

25. Із клавіатури вводиться послідовність $n=25$ цілих чисел. Знайти

добуток максимального та мінімального елементів цієї послідовності.

26. Із клавіатури вводиться послідовність $n = 20$ цілих чисел. Знайти кількість непарних елементів цієї послідовності.
27. Із клавіатури вводиться послідовність $n = 24$ цілих чисел. Знайти кількість елементів цієї послідовності, кратних 3.
28. Із клавіатури вводиться послідовність $n = 20$ цілих чисел: як додатних, так і від'ємних. Визначити, яких чисел у послідовності більше – додатних чи від'ємних та вивести відповідне повідомлення.
29. Обчислити суму 80 членів ряду, $i-$тий член якого визначається формулою $a_i = \frac{1}{(2i+1)!}$.
30. Обчислити суму 60 членів ряду, $i-$тий член якого визначається формулою $a_i = \left(\frac{1}{(2i+1)}\right)^i$.

Лабораторна робота №6

Тема роботи: Прості цикли із невідомим числом повторів

Мета роботи: Формування навиків в розробці простих циклічних програм із невідомим числом повторень для реалізації основних алгоритмів обробки даних

Для виконання роботи необхідно знати:

- поняття циклічного обчислення, його призначення;
- оператори циклу ***while*** та ***do…while*** та особливості їх використання;
- базові алгоритми обробки даних з використанням операторів циклу.

Теоретичні відомості

Оператор циклу ***while*** використовується для організації циклічних обчислень, якщо кількість повторів не відома наперед. Він виконується, поки виконується певна умова. Його синтаксис є наступним:

```
while (<логічний вираз>)
    <оператор>;
```

<Логічний вираз> - арифметичний або логічний вираз, що управляє роботою циклу. Поки вираз істинний – робота циклу продовжується. Якщо значення стає хибним – виконання циклу припиняється і управління передається наступному після циклу оператору. Цикл ***while*** не виконається жодного разу, якщо на початку його виконання ***<Логічний вираз>*** буде хибним.

<Оператор> - простий або складений оператор, сформований з використанням операторних дужок ***{}***. Вони становлять тіло циклу. Для формування тіла циклу можна також використати операцію слідування **,** (кома).

Оператора циклу ***do … while*** служить для виконання циклічних обчислень (тіла циклу) до тих пір, поки логічний вираз не стане хибним. Його синтаксис є наступним:

```
do  (<логічний вираз>)
    while  <оператор>;
```

Тіло циклу - ***<оператор>*** - може бути або простим, або складеним оператором. Тіло циклу виконується до тих пір, поки ***<логічний***

вираз> істинний. Коли значення логічного виразу стає хибним, проходить завершення циклічного обчислення та передача управління наступному оператору. Якщо на початку виконання циклу ***while <Логічний вираз>*** є хибним, то тіло циклу виконається один раз.

Для негайного виходу із циклу можна використати оператора ***break***, а для передачі управління на наступну ітерацію циклу – оператор ***continue*** (див. попередню роботу).

Приклади виконання завдання

Приклад №1 (використання оператора циклу ***while*** та складеного оператора)

Написати програму знаходження значення функції $y = \sin(x)$ із точністю ε (задати як константу) за допомогою її розкладу в ряд Маклорена:

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \ldots + (-1)^{n-1}\frac{x^{2n-1}}{(2n-1)!} + \ldots$$

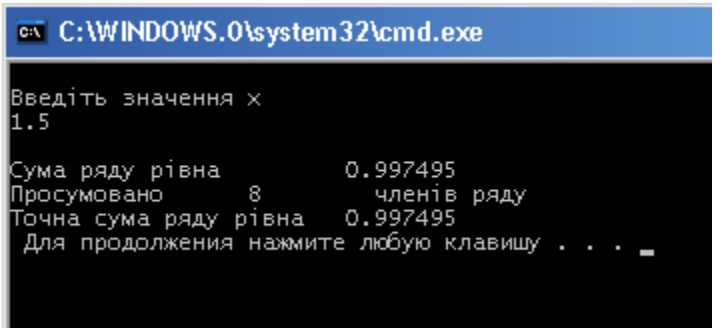
Примітка: *сумування проводиться до тих пір, поки* ***модуль*** *чергового члену ряду не стане меншим заданої точності ε.*

```
#include <iostream>
#include <math.h>
#include <windows.h>
#define  eps 1e-9
using namespace std;
void main ( )
{
int i;
double x,ai,s;
SetConsoleOutputCP(1251);
SetConsoleCP(1251);
cout << "\nВведіть значення x  \n";
cin >>x;
ai=x;
s=0;
i=1;
while (fabs(ai)>eps)
// тіло циклу
{    // початок складеного оператора
s=s+ai;
```

```
        i=i+1;
        ai=ai*(-x*x)/((2.0*i-1.0)*(2.0*i-2.0));
        } // кінець складеного оператора
        cout << "\nСума ряду рівна\t\t" << s;
cout<< "\nПросумовано \t " <<i<< "\t  членів ряду " ;
        s=sin(x);
        cout << "\nТочна сума ряду рівна\t" << s;
        cout << "\n ";
        }
```

Результат роботи програми:

Введено значення ***x=1.5***



Приклад №2 (використання оператора циклу ***do…while*** та операції слідування)

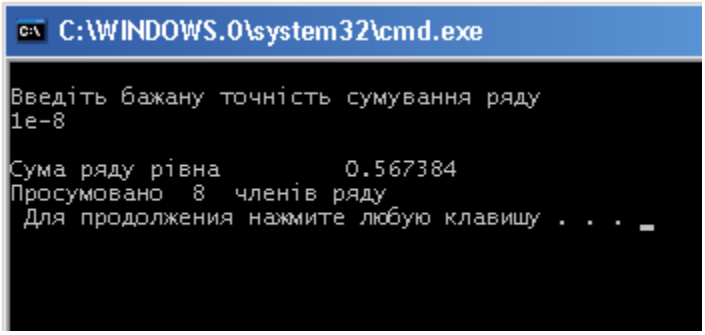
Написати програму знаходження суми ряду, $i-$тий член якого визначається формулою $a_i = \left(\frac{1}{2i}\right)^i$ із точністю ε (вводиться із клавіатури) з використанням оператора циклу ***do…while***.

```
#include <iostream>
#include <math.h>
#include <windows.h>
using namespace std;
void main ( )
{
int i;
SetConsoleOutputCP(1251);
SetConsoleCP(1251);
double ai,s;
```

```
        double eps;
        cout << "\nВведіть бажану точність сумування
    ряду  \n";
        cin >>eps;
        s=0;
        i=1;
        do
// тіло циклу з використанням оператора слідування
        ai=pow(1.0/(2*i),i),s=s+ai,i=i+1;
        while(ai>eps);
        cout << "\nСума ряду рівна\t\t" << s;
    cout<< "\nПросумовано  " <<i<< "  членів ряду " ;
        cout << "\n ";
        }
```

Результат роботи програми:

Введенно значення бажаної точності сумування ряду ***eps=1e-8***



Приклад №3

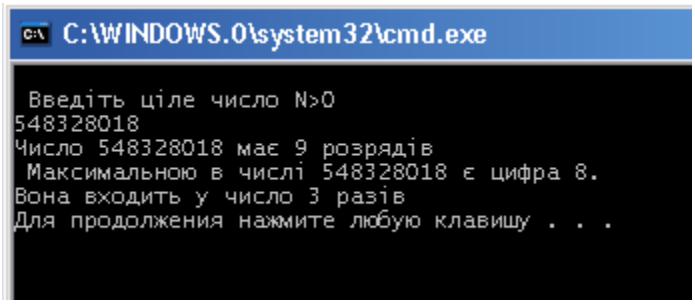
Із клавіатури вводиться довільне ціле число, більше нуля. Визначити розрядність числа, максимальну цифру та кількість її входжень у число.

```
        #include <stdio.h>
        #include <math.h>
        #include <windows.h>
        void main ( )
        {
        long int N,M,kilk=1;
        int max,poz;
        int i;
        SetConsoleOutputCP(1251);
```

```
        SetConsoleCP(1251);
        printf("\n Введіть ціле число N>0\n");
        scanf("%ld",&N);
        M=N;
        while(M/10>0)
        {
        kilk++;
        M/=10;
        }
    printf("Число %ld має %ld розрядів \n ", N,kilk);
        for (M=N, max=-1,poz=1,i=kilk;i>1;i--)
        {
        if(M%10>max)
        {
        max=M%10;
        poz=i;
        }
        M/=10;
        }
    printf("Максимальною в числі %ld є цифра %d.\n",
N, max);
        M=N;
        int j=0;
        for (i=kilk; i>=1; i--)
        {
        if(M%10==max) j++;
        M/=10;
        }
        printf("Вона входить у число %d разів\n",j);
        }
```

Результат роботи програми:

Введено число ***548328018***



Завдання для виконання

Написати програму розв'язку завдання двома способами – із використанням оператора циклу **`while`** *(перший спосіб) та за допомогою оператора циклу* **`do`**...**`while`** *(другий спосіб). Перевірити правильність виконання програми, використавши контрольні приклади.*

1. Обчислити значення квадратного кореня $y=\sqrt{x}$ із точністю ε, використавши ітераційну формулу Ньютона: $y_0=1$, $y_i=0.5(y_{i-1}+x/y_{i-1})$ $(i=1,2,3,...)$. Підрахувати кількість ітерацій, за які досягається бажана точність. Точність обчислення ε визначається як модуль різниці двох послідовних ітерацій.
2. Обчислювати суму членів ряду, поки черговий член не стане меншим ε.

   $$S=1 + 1/2 + 1/4 + 1/8 + ...$$

3. Обчислити значення кореня кубічного $y=\sqrt[3]{x}$ з точністю ε, використавши ітераційну формулу Ньютона $y_0=1$, $y_i=\frac{1}{3}(2y_{i-1}+x/y_{i-1}^2)$ $(i=1,2,3,...)$. Підрахувати кількість ітерацій, за які досягається бажана точність. Точність обчислення ε визначається як модуль різниці двох послідовних ітерацій.
4. Значення функції $\ln(x)$ можна обчислити її розкладом в ряд Маклорена:

   $$\ln(x)=x-\frac{x^2}{2}+\frac{x^3}{3}-\frac{x^4}{4}+...$$

   Обчислити $\ln(x)$ з точністю ε (обчислення продовжується до моменту, поки черговий член ряду по модулю не буде меншим ε). Підрахувати кількість членів ряду, які просумовано для досягнення заданої точності.
5. Для двох натуральних чисел m та n $(1<m<n)$ знайти найменше значення k, при якому $m^k>n$.
6. Значення функції $\cos(x)$ можна обчислити її розкладом в ряд Маклорена

   $$\cos(x)=1-\frac{x^2}{2!}+\frac{x^4}{4!}-\frac{x^6}{6!}+\frac{x^8}{8!}-...$$

Обчислити $\cos(x)$ з точністю ε (обчислення продовжується до моменту, поки абсолютне значення чергового члену ряду не буде меншим ε). Підрахувати кількість членів ряду, які необхідні просумувати для досягнення заданої точності.

8. Знайти наближено з точністю 0.01 найбільше значення функції $y=(ax^2+bx+c)/(d\cdot x+e)$ на відрізку $[x_1,x_2]$ (спосіб вибрати самостійно). Значення a, b, c, d, e, x_1, x_2 вводяться з клавіатури.
9. Для цілого числа $m>10$ отримати найбільше ціле k, для якого $4^k<m$.
10. Для натурального числа n знайти найменше натуральне число виду m^2, що перевищує n.
11. Значення функції $\sin^2(x)$ можна обчислити за допомогою її розкладу в ряд Маклорена

$$\sin^2(x)=x^2-2x^4/3-\ldots+(-1)^{n-1}2^{2n-1}x^{2n}/(2n)!$$

Обчислити $\sin^2(x)$ з точністю ε (обчислення продовжується до моменту, поки черговий член ряду по модулю не буде меншим ε). Підрахувати кількість членів ряду, які необхідно просумувати для досягнення заданої точності.

12. Міняючи x від початкового значення a з кроком h визначити, при якому значенні x добуток $x\cdot\sin(x)$ стане більшим $\cos(x)$. Вхідні дані вводяться із клавіатури.
13. Знайти корінь рівняння $e^x-10x=0$ з точністю ε методом простої ітерації.
14. Знайти корінь рівняння $tg(1.5773x)-2.3041x=0$ з точністю ε методом простої ітерації.
15. Знайти корінь рівняння $\ln(7.622x)-8.59x+0.5=0$ з точністю ε методом простої ітерації.
16. Знайти корінь рівняння $9.33\sin(6.977x)-7.25x=0$ з точністю ε методом простої ітерації.
17. Вивести значення функції $y=\sin(ax)$ для x, що міняються від 0 з кроком 0.05 до тих пір, поки функція зростає. Значення a вводиться із клавіатури.
18. Уточнити корінь рівняння $e^x-10x=0$ на проміжку $[a,b]$ методом половинного ділення. Ітераційний процес продовжується до тих пір,

поки величина проміжку не стане меншою заданої величини ε (0.001) (задається як константа).

19. Обчислити значення числа π, використавши приведену нижче формулу, із заданою точністю eps (0.001), введеною із клавіатури.

$$\frac{\pi}{4}=1-\frac{1}{3}+\frac{1}{5}-\frac{1}{7}+\frac{1}{9}-...$$

20. Обчислити значення числа π, використавши приведену нижче формулу, із заданою точністю ε (0.0001), введеною із клавіатури.

$$\frac{\pi}{8}=\frac{1}{1\times 3}+\frac{1}{5\times 7}-\frac{1}{9\times 11}+...$$

21. Написати програму обчислення суми членів послідовності, яка задається формулою

$$a_n=\frac{n}{n^2+1}$$

Сумування членів ряду проводити до тих пір, поки $|a_n|\geq eps$, $eps=0.0001$.

22. Написати програму обчислення суми членів послідовності, яка задається формулою

$$a_n=\frac{(-1)^n}{2n!}$$

Сумування членів ряду проводити до тих пір, поки $|a_n|\geq eps$, $eps=0.0005$.

23. Написати програму обчислення суми членів послідовності, яка задається формулою

$$a_n=\sin\left(\frac{1}{n}\right)$$

Сумування членів ряду проводити до тих пір, поки $|a_n|\geq eps$, $eps=0.0025$.

24. Написати програму обчислення суми членів послідовності, яка задається формулою

$$a_n=\frac{1}{n!}$$

Сумування членів ряду проводити до тих пір, поки $|a_n| \geq eps$, $eps = 0.00001$.

25. Знайти корінь рівняння $\sin(x) + x - 0.5 = 0$ на проміжку $[0;1]$ з точністю ε=0.001 методом половинного ділення.
26. Із клавіатури вводиться довільне ціле число. Підрахувати суму його цифр.
27. Знайти корінь рівняння $e^x - \ln(x) - 20 = 0$ на проміжку $[3;3.2]$ з точністю ε=0.005 методом половинного ділення.
28. Із клавіатури вводиться довільне ціле число. Отримати із нього нове число, в якому цифри слідують у зворотному порядку.
29. Написати програму обчислення суми членів ряду, n-ний член якого задається формулою

$$a_n = \frac{1}{(n+1)^3}$$

Сумування членів ряду проводити до тих пір, поки $|a_n| \geq eps$, $eps = 0.0001$. Підрахувати кількість просумованих членів ряду.

30. Знайти корінь рівняння $x^3 - e^x - 5.5 = 0$ на проміжку $[2;3]$ з точністю ε=0.0001 методом половинного ділення.

Лабораторна робота №7

Тема роботи: Програмування з використанням функцій

Мета роботи: Формування навиків в розробці та використанні функцій для структуризації програми при розробці складних алгоритмів обробки даних.

Для виконання роботи необхідно знати:

- поняття функції та синтаксис її опис;
- рекурсивні функції;
- перевантажені функції;
- фактичні та формальні параметри;
- вказівники і адреси змінних, їх використання та основні операції з ними.

Теоретичні відомості

Програма на мові С++ є сукупністю функцій. Функції використовують для структурування складної програми, особливо коли одну і ту ж саму послідовність операцій необхідно виконати кілька разів.

Функція – це закінчений модуль, що містить деяку послідовність операторів, оформлених таким чином, що її можна викликати в будь-якому місці програми та служить для розв'язку певної локальної задачі. Особливістю програми на мові С++ є те, що вона обов'язково містить одну головно функцію ***main*** (головна). Інші функції є рівноправними по відношенню одна до одної, але головною завжди залишається функція ***main***.

Загальний синтаксис опису функції є наступним:

```
<тип> <імя_функції>(<список_аргументів>)
    {
<тіло функції, що складається із набору операторів>
     return <значення,що повертається в головну функцію>;
    }
```

де:

<тип> – задає тип значення, що повертається, та може бути будь-яким допустимим в мові С++;

<імя_функції> – утворюється аналогічно до правил побудови ідентифікаторів змінних;

<список_аргументів> – список формальних параметрів, розділених комами;

<тіло функції> – набір операторів, які реалізують алгоритм роботи функції;

<значення,що повертається> – результат роботи функції – значення, яке повертається в головну програму за допомогою оператора ***return***. За типом воно повинно відповідати типу самої функції. У головну програму також жодне значення може і не повертатися – тоді використовується ключове слово ***void*** (ніякий).

При виклику функції на місці формальних параметрів повинні стояти фактичні параметри відповідного типу: при цьому передаються тільки значення відповідних параметрів, які змінені бути не можуть. Значення формальних параметрів можна задавати і по замовчуванню: якщо такий параметр буде відсутній при виклику функції, то для нього буде взято значення по замовчуванню. Всі такі параметри повинні стояти в кінці списку формальних параметрів. Наприклад (функція для піднесення числа ***a*** до степені ***n***. При відсутності степеня (***n***) число підноситься до квадрату):

```
float stepin(float a, int n=2)
stepin(float a,)/* виклик функції для піднесення числа a до квадрату */
```

Глобальні змінні може використовувати будь-яка функція. Змінні, що ініціалізовані в функції є локальними і видимі тільки в її межах.

У програмі на мові С++ допускається рекурсивне використання функцій. Функція називається рекурсивною, якщо під час її виконання здійснюється її повторний виклик або напряму (пряма рекурсія), або через інші функції (непряма рекурсія). Структурна схема прямої рекурсії має наступний вигляд:

```
void fact(int k);
{
    . . .
void fact(k);
    . . .
}
```

Щоби функція не створила нескінчену послідовність викликів самої себе, обов’язковою є перевірка досягнення вибраним параметром граничного значення (див приклад №2).

Функція в головну програму може повернути тільки одне значення. Для повернення в головну програму більше одного значення в якості аргументів функції необхідно використовувати вказівники або адреси змінних.

Кожна змінна у програмі – це об'єкт, який володіє ім'ям і значенням. Після компіляції програми всі імена змінних заміняються адресами комірок пам'яті, в яких містяться відповідні дані. На машинному рівні імена змінних у командах не використовуються, а тільки відповідні адреси.

Вказівник – це змінна або константа стандартного типу для збереження змінної визначеної типу. Значення вказівника – це беззнакове ціле, що вказує на адресу розміщення і ніяким чином не говорить про саму змінну. Змінна типу вказівника оголошується аналогічно до змінних іншого типу, тільки із використанням унарного символу *. Вказівник містить адресу нульового байту змінної, а тип змінної визначає, скільки байтів, починаючи із цієї адреси, займає це значення. Синтаксис оголошення вказівника є наступним:

<тип> * <вказіввник>

де:

<тип> - тип змінної, адресу якої буде містити покажчик.

Наприклад:

```
int *i;     //Вказівник - змінна на дані типу int
double *Fx;/* Вказівник - змінна на дані типу double */
float *x;   /*Вказівник - змінна на дані типу float  */
```

С++ для роботи зі змінними дозволяє використовувати дві основні операцій: & та *. Ці операції служать для:

&<імя змінної> – визначення адреси розміщення значення змінної;

**** <вказівник>*** – отримання значення визначеного типу за вказаною адресою.

Оператор присвоєння значення адреси покажчику має наступний синтаксис:

<змінна - вказівник>=&<імя змінної>

і при цьому операндом операції ***&*** повинно бути імя змінної того ж типу, що і вказівник лівої частини оператора. Наприклад:

```
int k, *vk; //Опис змінної - вказівника *vk
vk=&i;      /*Вказівнику присвоюється значення адреси змінної i  */
```

Непряма адресація змінної за допомогою вказівника дозволяє отримати доступ до значення змінної. Синтаксис такої операції є наступним:

```
<ім'я змінної>=* <вказівник>
```

Наприклад:

```
i=*vk;  /*  Змінна i отримує значення,
розташоване за заданою
вказівником *vk адресою   */
```

В програмі для звертання до значення за допомогою вказівника його можна використовувати в операторі присвоєння скрізь, де може бути ім'я самої змінної. Фактично вказівник є «синонімом» змінної, на яку він вказує. До вказівників можуть бути застосовані операції присвоєння, цілочисельної арифметики та порівняння.

С++ допускає створення декількох функцій із одним і тим же ім'ям, але різним набором формальних параметрів (різним типом, кількістю чи порядком слідування). Такі функції називають перевантаженими. Вони створюються у випадках, коли одна функція над різними наборами даних має виконувати обчислення за різними алгоритмами.

Приклади виконання завдання лабораторної роботи

Приклад №1

Програма знаходження числа комбінацій із n елементів по m за формулою

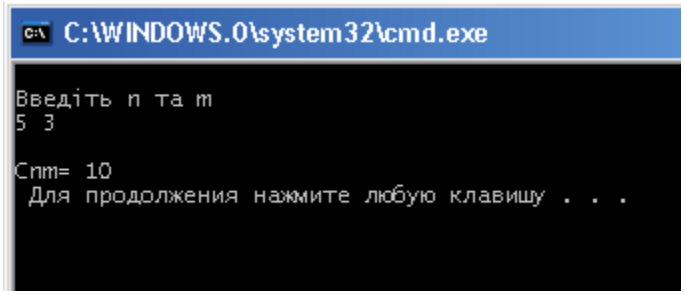
$$C_n^m = \frac{n!}{m!(n-m)!}.$$

```
#include <iostream>
#include <windows.h>
using namespace std;
int fact(int n)// Функція, що обчислює n!
{
int p=1;
for (int i=1;i<=n;i++)
p=p*i;
return p;
}
void dryk(int Cnm) /* Функція, що виводить на
екран значення Cnm   */
```

```
{
cout << "\nCnm= " << Cnm;
cout << "\n ";
}
void main ( )
{
int n,m;
int Cnm;
SetConsoleOutputCP(1251);
SetConsoleCP(1251);
cout << "\nВведіть n та m \n";
cin >>n;
cin >>m;
// Обчислення Cnm
Cnm=fact(n)/(fact(m)*fact(n-m));
dryk(Cnm);
}
```

Результат роботи програми:

Введено значення ***n=5*** та ***m=3***



Примітка: *зверніть увагу, що для виводу результату обчислення описана окрема функція* ***dryk****.*

Приклад №2 (рекурсивне використання функції)

Програма знаходження *n*! з використанням рекурсії.

```
#include <iostream>
#include <windows.h>
#include<stdio.h>
using namespace std;
int Factorial(int N) /*Рекурсивна функція,що
обчислює n! */
```

```
    {
    int F;
    if (N<=1) F=1; else F=Factorial(N-1)*N;
/*Перевірка умови
завершення рекурсивного процесу*/
    return F;
    }
    void main()
    {
    int k;
    SetConsoleOutputCP(1251);
    printf(" Введіть N  \n" );
    cin >>k;
    printf("Факторіал(%i)=%i\n",k,Factorial(k));
    }
```

Приклад №3 (використання адресації змінних та вказівників для повернення із функції значення двох змінних)

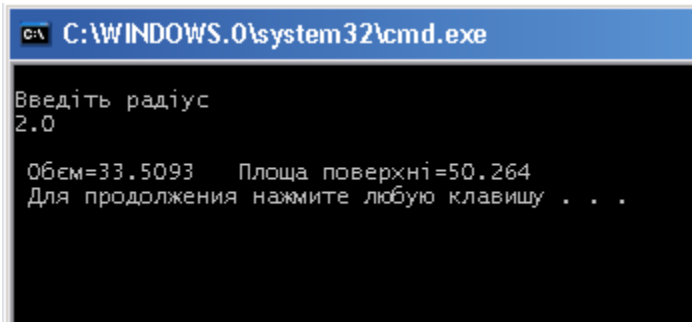
Програма знаходження за введеним із клавіатури радіусом об'єму кулі та площі її поверхні.

```
#include <iostream>
#include <math.h>
#include <windows.h>
using namespace std;
void Vs(double &V, double &s, double r)
{
double Pi=3.1415;
V=4./3.*Pi*pow(r,3);
s=4.*Pi*pow(r,2);
}
void main ( )
{ double v,s;
double r;
SetConsoleOutputCP(1251);
SetConsoleCP(1251);
cout << "\nВведіть радіус \n";
cin >>r;
v=0;
s=0;
Vs(v,s,r);
```

```
        cout << "\n Обєм=" << v<<"\t"<<"Площа
    поверхні="<<s;
        cout << "\n ";
        }
```

Результат роботи програми:

Введено значення ***r=2.0***



Приклад №4 (використання вказівників для повернення із функції значення двох змінних)

Програма знаходження за введеним із клавіатури радіусом об'єму кулі та площі її поверхні.

```
#include <iostream>
#include <math.h>
#include <windows.h>
using namespace std;
void Vs(double *v, double *s, double r)
{
double Pi=3.1415;
*v=4./3.*Pi*pow(r,3);
*s=4.*Pi*pow(r,2);
}
void main ( )
{
double v,s;
double r;
SetConsoleOutputCP(1251);
SetConsoleCP(1251);
cout << "\nВведіть радіус \n";
cin >>r;
v=0;
```

```
    s=0;
    Vs(&v,&s,r);
    cout << "\n Обєм=" << v<<"\t"<<"Площа
поверхні="<<s;
    cout << "\n ";
    }
```

Приклад №5 (використання перевантажених функцій та формальних параметрів по замовчуванню)

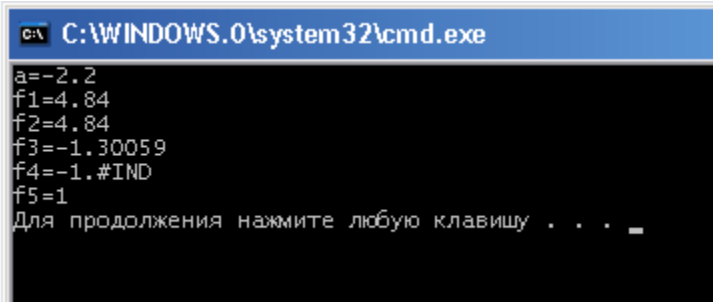
Обчислення степені дійсного числа ***a***. Ціла степінь числа обчислюється за допомогою рекурсії. Дійсна степінь числа (степінь задається у вигляді дробу ***k/m***) обчислюється за формулою $a^n = e^{(\ln(a)*n)}$

```
    #include <iostream>
    #include <math.h>
    #include <windows.h>
    using namespace std;
    /*Обчислення цілої степені числа a за
допомогою рекурсії*/
    /* По замовчуванню число підноситься до
другого степеня */
    double stepin(float a, int n=2)
    {
    if (n==0) return(1.);
    else
    if (n<0) return(1./stepin(a,-n));
    else
    return(a*stepin(a,n-1));
    }
    //Обчислення значення числа a в степені k/m
    float stepin( float a, int k, int m)
    // степінь k/m.
    {
    if (a==0) return (0);
    else
    if (k==0) return(1);
    else
    if (a>0)
    return(exp(float(k)/float(m)*log(a)));
    else
    if (m%2!=0)
```

```
return (-(exp(float(k)/float(m)*log(-a))));
if (m%2==0) cout<<"обчислити неможливо\n";
}
void main()
{
SetConsoleOutputCP(1251);
SetConsoleCP(1251);
float a;
double f;
cout<<"a=";
cin>>a;
f=stepin(a,2);
cout<<"f1="<<f<<endl;
//Обчислення цілої степені числа a
//Степінь 2 задається по замовчуванню
f=stepin(a);
cout<<"f2="<<f<<endl;
//Обчислення числа a в степені k/m
int k=1;
int m=3;
f=stepin(a,k,m);
cout<<"f3="<<f<<endl;
f=pow(a,(float(k)/float(m)));
cout<<"f4="<<f<<endl;
f=pow(a,k/m);
cout<<"f5="<<f<<endl;
}
```

Результат роботи програми:

Введено значення **a=-2.2**



Примітка: *зверніть увагу на результат обчислення від'ємного числа* **a=-2.2** *в степені* $\frac{1}{3}$. *Результат рівний* **f3=-1.30059.** *При*

спробі піднесення до степеня $1/3$ *числа за* **a** *допомогою функції* **pow** *виводиться або повідомлення про помилку* **f4=-1.#IND** (**f=pow(a,(float(k)/float(m))**), *або отримується неправильний результат* **f5=1** (**f=pow(a, k/m)**).

Завдання для виконання

Написати програму обчислення арифметичних виразів завдань лабораторної роботи №3. Завдання №1 реалізувати з використанням звичайної функції та можливості задання значень формальних параметрів по замовчуванню. Завдання №2 розв'язати або з використанням вказівників, або з використанням адресації змінних. Вивід результатів обчислення на екран організувати за допомогою окремих функцій (див. Приклад №1).

Лабораторна робота №8

Тема роботи: Вкладені цикли та задачі обробка масивів

Мета роботи: Формування навиків розробки циклічних алгоритмів обробки масиву в цілому та його окремих елементів.

Для виконання роботи необхідно знати:

- поняття масиву та його опис;
- вкладені цикли та їх використання;
- використання вказівників при роботі з масивами;
- базові алгоритми обробки масивів.

Теоретичні відомості

Масивом називають набір однотипних даних, для збереження та обробки яких служить одне ім'я. Будь-який масив попередньо повинен бути описаний. Одномірний масив описується наступним чином:

<тип> <імя_масиву> [розмір]=<ініціалізація>

де:

<тип> – тип елементів масиву;

<імя_масиву> – імя, що будується згідно прави побудови імен змінних;

[розмір] – кількість елементів масиву;

<ініціалізація> – задання початкового значення елементам масив. Вона може бути відсутньою.

Наприклад:

float mas[5]– масив 5 дійсних чисел;

int mas[7]={2,3,24,10,9,6,4}– масив 7 цілих чисел;

double mas[]={4,3.1,4,10.2,9,7.3}– масив 6 дійсних чисел подвійної точності. При такому описі розмірність масиву можна не вказувати.

Для звернення до елементів масиву необхідно вказати ім'я масиву та номер елементу в квадратних дужках. Необхідно пам'ятати, що індекси елементів масивів у С++ починаються з 0. Наприклад:

mas[2]=2 – присвоєння ***3***-тьому елементу значення ***2***;

mas[i+1]=4.2 - присвоєння ***i+2***-му елементу масиву значення ***4.2***;

mas[2i-1]=2x*sin(s) – присвоєння ***2i***-му елементу масиву значення виразу ***2x*sin(s)***.

Багатомірний масив у С++ розглядається як сукупність масивів меншої розмірності. Так, двомірний масив розглядається як сукупність одномірних масивів – його рядків. Тривимірний – як сукупність матриць, які в свою чергу є сукупністю одномірних масивів – рядків. Опис ***n*** – мірного масиву матиме наступний синтаксис:

<тип><імя_масиву>[розмір 1][розмір 2]…[розмір n]

Наприклад:

int mas[5][7]– масив цілих чисел розміром 5×7;

float x[5][6][10]– масив дійсних чисел розміром $5\times6\times7$;

int s[2][2]={1,2,5,6}– масив цілих чисел розміром 2×2 з ініціалізацією.

Оскільки ділянка у оперативній пам'яті під масив задається на етапі компіляції і її розмір визначається типом елементів масиву та їх кількістю, розмір масиву повинен бути визначений у тексті програми, а не підчас її виконання.

До елементів масиву можна звертатися двома способами:

1. За допомогою імені масиву: до 5-го елементу одномірного масиву ***mas*** можна звернутися як ***mas[4]*** ;
2. За допомогою вказівника масиву ****(mas+4.*** (Ім'я масиву є одночасно вказівником на його нульовий елемент).

Для доступу до всіх елементів масиву та їх опрацювання за тим чи іншим алгоритмом необхідно використовувати циклічні оператори. Наприклад для вводу всіх елементів двомірного масиву ***a[i0][j0]*** із клавіатури служить наступний фрагмент програми (із виводом додаткової пояснювальної інформації на екран):

```
for(i=0;i<i0;i++)
for(j=0;j<j0;j++)
{
printf(" Введіть A[%i,%i]=  \n",i,j );
cin >>a[i][j];
}
```

Примітка: *перевірка виходу індексів елементів за допустимі межі масиву не виконується і повідомлення в ході виконання програми не видається.*

При використанні у функції в якості формального параметру масиву вона отримує тільки адресу його початку (першого елементу), а кількість елементів (розмірність масиву) при цьому має бути передана окремо за

допомогою додаткового формального параметру. Наприклад, функція ***sum*** сумує елементи масиву ***p***:
перший варіант (робота як з масивом):

```
/*формальні параметри - вказівник масиву та
кількість елементів*/
    int sum(int *p, int n)
    {
    int result = 0;
    for (int i = 0 ; i < n; i++)
    {
    result += p[i]; //Сумування елементів масиву
    }
    return result;
    }
```

другий варіант (робота як з вказівником):

```
    int sum(int *p, int n)
    {
    int result = 0;
    for (int i = 0 ; i < n; i++)
    {
    result += *p; //Сумування елементів масиву
    p++;
    }
    return result;
    }
```

Тут ****p*** отримує значення елементу масиву, на який вказує ***p*** , а ***p++*** виконує зсув вправо на один елемент масиву.

Приклади виконання завдання

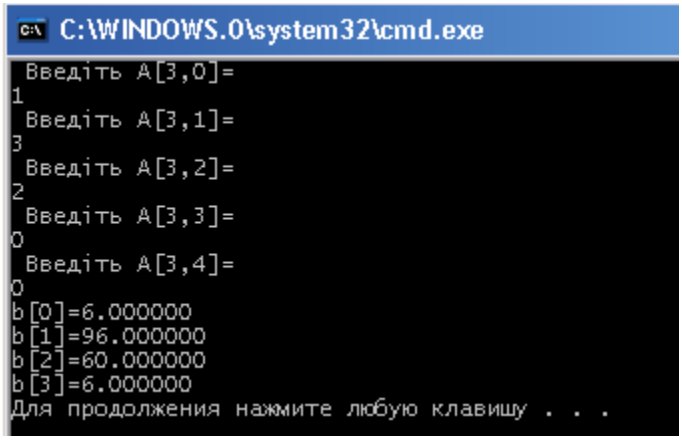
Приклад №1

Програма, яка для матриці дійсних чисел $A_{4\times6}$ обчислює добуток ненульових елементів кожного стовпця, а результат виводить у одномірний масив B та на екран.

```
#include <iostream>
#include<stdio.h>
```

```
#include <windows.h>
#define i0 4
#define j0 5
using namespace std;
  void main()
{
float a[i0][j0];
float b[j0];
int i,j;
SetConsoleOutputCP(1251);
SetConsoleCP(1251);
cout << "\nВведіть елементи масиву A\n" ;
for(i=0;i<i0;i++)
for(j=0;j<j0;j++)
{
printf(" Введіть A[%i,%i]=  \n",i,j );/* Ввід
  елементів матриці із клавіатури    */
cin >>a[i][j];
}
for(i=0;i<i0;i++)
{
b[i]=1.0;
for(j=0;j<j0;j++)
{
/* Перевірка,чи поточний елемент не дорівнює нулю та
сумування у однорідному масиві b */
if (a[i][j]!=0) b[i]=b[i]*a[i][j];
}
printf("b[%i]=%f\n",i,b[i]);
}
}
```

Кінцевий фрагмент роботи програми:



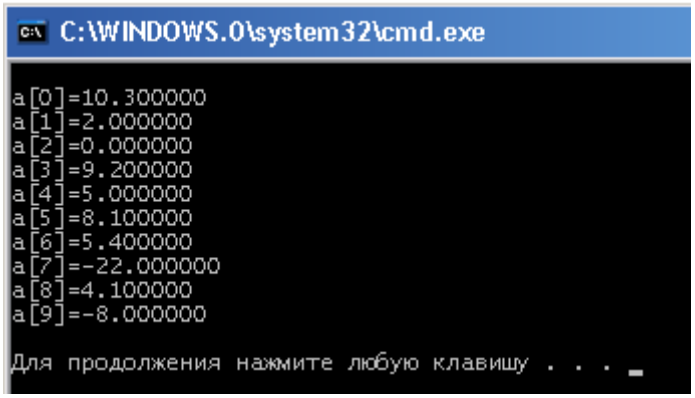
Приклад №2

Написати програму, яка у векторі дійсних чисел A_{10} знаходить мінімальний та максимальний елементи та міняє їх місцями. Значення елементів вектора ініціалізуються при його описі. Вивести вектор A на екран після перестановки елементів.

```
        #include <iostream>
        #include<stdio.h>
        #define i0 10
        using namespace std;
        void main()
        {
        // опис масиву та його ініціалізація
    double a[i0]={- 22,2.0,0.0,9.2,
    5.0,8.1,5.4,10.3,4.1,-8.0};
        double a_min,a_max;
        int i;
        int i_min,i_max;
        SetConsoleOutputCP(1251);
        SetConsoleCP(1251);
        a_min=a_max=a[1];
        i_min=i_max=1;
        printf("\n");
        for(i=0;i<i0;i++)
        {
if (a_min<a[i]) {/*Пошук мінімального значення та
його індексу */
        a_min=a[i];
        i_min=i;
        }
if (a_max>a[i]) {/*Пошук максимального значення та
його індексу */
        a_max=a[i];
        i_max=i;
        }
        }
/*Обмін місцями максимального та мінімального
елементів */
        a[i_min]=a_max;
        a[i_max]=a_min;
        for(i=0;i<i0;i++)
```

```
printf("a[%i]=%f\n",i,a[i]);
printf("\n");
}
```

Результат роботи програми:



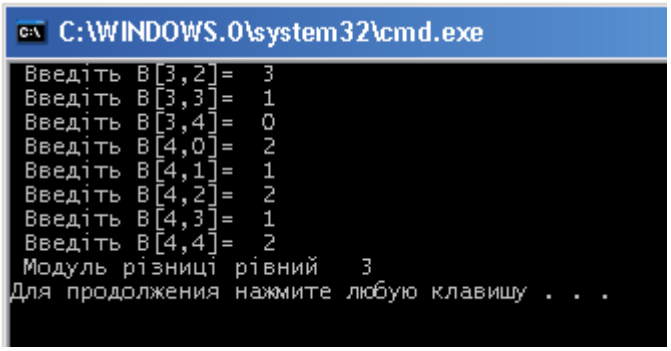
Приклад №3

У квадратній матриці $B_{5\times5}$ цілих чисел обчислити модуль різниці між числом нульових елементів, що стоять нижче головної діагоналі та числом нульових елементів, що стоять над головною діагоналлю.

```
#include <iostream>
#include<stdio.h>
#include <math.h>
#include <windows.h>
#define i0 5
#define j0 5
using namespace std;
void main()
{
SetConsoleOutputCP(1251);
SetConsoleCP(1251);
float b[i0][j0];
int i,j;
int i_1=0,i_2=0;
cout << "\nВведіть елементи матриці B\n" ;
for(i=0;i<i0;i++)
for(j=0;j<j0;j++)
{
printf(" Введіть B[%i,%i]=  ",i,j );
cin >>b[i][j];
}
```

```
        for(i=0;i<i0;i++)
        for(j=0;j<j0;j++)
        {
if ((i>j) && (b[i][j]==0) ) i_1++;/*Підрахунок
нульових елементів вище головної діагоналі*/
if ((i<j) && (b[i][j]==0))  i_2++;/*Підрахунок
нульових елементів нижче головної діагоналі*/
        }
printf("Модуль різниці рівний %i\n",abs(i_1-i_2) );
        }
```

Кінцевий фрагмент роботи програми:



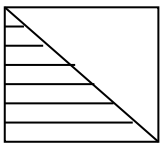
Завдання для виконання

Написати програму розв'язку задачі. Перевірити правильність роботи програми за допомогою контрольних прикладів. Результат вивести на екран з використанням форматного виводу у зручній для сприйняття формі.

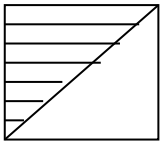
1. У квадратній матриці $A_{6\times6}$ цілих чисел обчислити модуль різниці між кількістю нульових елементів, що стоять нижче головної діагоналі та кількістю ненульових елементів, що стоять на головній діагоналі.
2. У матриці $B_{10\times6}$ обчислити середнє арифметичне кожного стовпця, а результат сформувати у вигляді одномірного масиву S.
3. У квадратній матриці $A_{5\times5}$ обчислити суму додатних елементів, що стоять над головною діагоналлю, та суму від'ємних елементів, що стоять на головній діагоналі.
4. Перевірити, чи є матриця, введена із клавіатури, симетричною відносно головної діагоналі.

5. В кожному рядку матриці $A_{5\times5}$ замінити елемент головної діагоналі (якщо він є простим числом) сумою вище розміщених елементів стовпця.
6. У квадратній матриці $C_{6\times6}$ знайти суму елементів, що знаходяться по її периметру.
7. У квадратній матриці $C_{6\times6}$ знайти добуток ненульових елементів, що знаходяться на головній та допоміжній діагоналях. Підрахувати загальну кількість нульових елементів матриці.
8. Повернути матрицю $B_{6\times6}$ на 90° за годинниковою стрілкою (рядки повинні стати стовпцями) та підрахувати кількість елементів, для яких виконується рівність $B_{i,j}=i+j$.
9. Матрицю $A_{5\times6}$ ввести із клавіатури. Якщо всі елементи або рядка, або стовпця рівні між собою, то замінити їх одиничками.
10. Обчислити найбільшу суму модулів елементів по кожному із рядків матриці $A_{6\times10}$.
11. Для двох матриць однакової розмірності обчислити або їх суму, або їх різницю в залежності від значення числа ***flag***, введеного із клавіатури (0 або 1).
12. Підрахувати кількість рядків у заданій квадратній матриці, які містять тільки додатні елементи.
13. У квадратній матриці дійсних чисел $A_{8\times8}$ знайти кількість рядків, сума модулів елементів яких не перевищує введеного із клавіатури числа.
14. У квадратній матриці дійсних чисел $A_{7\times7}$ обчислити середнє арифметичне додатних елементів кожного стовпця та сформувати із них вектор X.
15. Перевірити, чи у квадратній матриці $A_{5\times5}$ суми елементів по рядках рівні між собою.
16. Із квадратної матриці $A_{10\times10}$ побудувати вектор, елементи якого є найменші числа кожного зі стовпців матриці.
17. Із квадратної матриці $A_{3\times7}$ побудувати нову матрицю B, в якій максимальні елементи по стовпцях поміняні місцями із елементами головної діагоналі.

18. Дано координати n точок x_i та y_i, що лежать на площині (сформовані у вигляді двомірного масиву). Підрахувати кількість точок, що віддалені від початку координат не більше ніж на введену із клавіатури відстань.
19. Дано координати n точок x_i та y_i , що лежать на площині (сформовані у вигляді двомірного масиву). Визначити та видрукувати номер координатної чверті, у якій знаходиться максимальна кількість точок.
20. Сформувати квадратну матрицю цілих чисел $B_{7\times7}$, кожен елемент якої є добутком номера рядка на номер стовпця, в якому він знаходиться. Вивести створену матрицю на екран та обчислити середнє арифметичне елементів головної діагоналі.
21. Впорядкувати елементи вектора A, що містить n натуральних чисел, у порядку зростання.
22. У масиві $B_{5\times5}$ визначити та вивести елементи, які зустрічаються більше одного разу.
23. У матриці $A_{7\times6}$ визначити та вивести номер рядка, який містить максимальну кількість нульових елементів.
24. У матриці $A_{6\times6}$ підрахувати кількість рядків, які містять хоча б один нульовий елемент.
25. В матриці $A_{3\times6}$ поміняти місцями елементи першого стовпця із елементами останнього стовпця.
26. В квадратній матриці $A_{4\times4}$ провести циклічний зсув стовпців на 2 позиції – перший стовпець стане третім, другий – четвертим, і. т.д.
27. В квадратній матриці $A_{6\times6}$ обчислити суму елементів, що знаходяться в заштрихованій області.



28. В квадратній матриці $A_{7\times7}$ обчислити суму елементів, що знаходяться в заштрихованій області.



29. У квадратній матриці дійсних чисел $A_{8\times 8}$ обчислити різницю середніх арифметичне елементів головної та допоміжної діагоналей.
30. У квадратній матриці дійсних чисел $A_{5\times 5}$ замінити всі нульові елементи середнім геометричним ненульових елементів за виключенням елементів головної діагоналі.

Лабораторна робота №9

Тема роботи: Робота із файлами

Мета роботи: Формування навиків та умінь при роботі із файлами засобами С++.

Для виконання роботи необхідно знати:

- правила опису файлів;
- правила доступу до файлових даних;
- ввід-вивід інформації з використанням файлових змінних;
- алгоритми роботи із файлами.

Теоретичні відомості

В С++ можлива робота як з текстовими, так і з двійковими (бінарними) файлами. Про роботу із двійковими файлами в рамках даного практикуму мова йти не буде. Текстовий файл – файл, в якому кожний символ зберігається у вигляді окремого байту. Текстовий файл розбивається на рядки спеціальним символом «кінець рядка **\n**» та завершується символом «кінець файлу ***EOF***». ***EOF*** (end of file) - відємним числом, як правило -1, що гарантує неспівпадіння із кодом будь-якого іншого символу. Якщо в текстовому файлі знаходяться числові дані, то після останнього числа не можуть стояти ні пробіли, ні символ кінця рядка. Два числа в текстовому файлі вважаються окремими, якщо вони розділені між собою хоча-б одним із наступних символів: пробіл; символ табуляції; символ кінці рядка.

Операції з файлами в С++ можна здійснювати за допомогою файлових вказівників та потокових операцій.

Стиль мови програмування С (файлові вказівники).

Для роботи із файлом необхідно:

1. – описати вказівник на файлову змінну
   ```
   FILE *<імя файлової змінної>
   ```
2. - відкрити файл за наступним синтаксисом:
   ```
   FILE *fopen(filename, mode)
   ```

де:

filename – ім’я файлу, що відкривається. Файл повинен знаходитися у папці, в якій міститься сама програма на С++, інакше необхідно задавати повне ім’я файлу;

mode – режим відкриття файлу (див. додаток №7).

При роботі із текстовими файлами в кінці специфікатора ***mode*** додається символ ***t*** – текстовий файл. Наприклад:

```
FILE *f;
if ((f=fopen("dani.in", "rt"))==NULL)
{
printf("Не можливо відкрити файл"\n);
return;
}
```

Тут файлова змінна ***f*** зв'язується із текстовим файлом ***dani.in***, який відкривається тільки для читання. В логічному операторі ***if*** виконується перевірка успішності відкриття файлу. Якщо файл відкрити не вдалося (***==NULL***), то на екран видається відповідне повідомлення і робота програми завершується. По завершенню роботи із файлом він повинен бути закритий функцією ***fclose()***. Так, файл, відкритий у попередньому прикладі, буде закритий наступною командою:

```
fclose(f)
```

Інформацію із текстового файлу можна читати декількома способами:

1. Посимвольно за допомогою функції ***putc()*** за наступним синтаксисом (див приклад №1):

   ```
   fgetc(<ім'я файлової змінної>)
   ```

2. По рядках за допомогою функції ***fgets()*** за наступним синтаксисом (див приклад №2.):

   ```
   fgets(char s, int n, <ім'я файлової змінної>)
   ```

де:

char s – рядкова змінна, якій передається ***n*** зчитаних символів. Зчитується або ***n-1*** символ (n-тий символ – нульовий символ ***\0***), або до появи кінця рядка (***\n***).

3. Читання текстового файлу за форматом здійснюється за допомогою функції ***fscanf()*** (див приклад №3.):

   ```
   fscanf(<ім'я файлової змінної>, format,
   address)
   ```

де:

format – перелік форматів аргументів;

address char – перелік ідентифікаторів змінних, заданих своїми адресами.

При успішному читанні із файлу функція ***fscanf()*** повертає кількість прочитаних полів, тому організація перевірки правильності зчитування може виглядати наступним чином (при читанні двох даних із файлу ***f***):

```
if (fscanf(f,"%i%d",&i,&x)!=2)
{
printf("Помилка читання із файлу!");
};
```

Для перевірки умови досягнення кінця файлу служить функція ***feof()***. Якщо зчитано символ кінця файлу (***EOF***), то ***feof()*** повертає ненульове значення.

Для запису даних у файл використовуються функції ***fprintf()***,***fputc()*** та ***fputs()*** (див приклад №4):

fprintf() аналогічна функції ***printf()***, тільки першим аргументом у ній є посилання на файл, у який виконується запис;

fputc() та ***fputs()*** аналогічні функціям ***putc()*** та ***puts()*** відповідно, тільки першими аргументами у них є посилання на файл, у який виконується запис.

Стиль мови програмування С++ (потокові операції).

В С++ робота із файлом здійснюється за допомогою функцій, що знаходяться у заготовочному файлі ***<fstream.h>***:

Для запису даних у файл необхідно:

1. описати змінну типу ***ofstream*** за наступним синтаксисом:
   ofstream <ім'я файлової змінної>
2. відкрити файл за допомогою функції ***open***:
   <ім'я файлової змінної>.open ("<ім'я файлу>", mode)
   тут ***mode*** – задає режим відкриття файлу.
3. Записати інформацію у файл за наступним синтаксисом:
   <ім'я файлової змінної> << <змінна>
   тут ***<змінна> -*** ідентифікатор змінної, яка записується у файл.
4. Закрити файл командою ***<ім'я файлової змінної>.close()***.

Для читання даних з файлу необхідно:

1. описати змінну типу ***ifstream*** за наступним синтаксисом:
   ifstream <ім'я файлової змінної>
2. відкрити файл за допомогою функції ***open***:
   <ім'я файлової змінної>.open ("<ім'я файлу>", mode)
   тут ***mode*** – задає режим відкриття файлу.
3. Зчитати інформацію з файл за наступним синтаксисом:
   <ім'я файлової змінної> >> <змінна>

тут ***<змінна>*** – ідентифікатор змінної, якій присвоюється зчитане із файлу значення.

4. Закрити файл командою ***<імя файлової змінної>.close()***.

Основні режими відкриття файлу ***mode***:

ios::out – файл відкрито для запису даних (по замовчуванню для ***ofstream***);

ios::in – файл відкрито для читання даних (по замовчуванню для ***ifstream***);

ios::app – файл відкрито для запису даних в кінець файлу.

Для перевірки досягнення кінця файлу при вводі даних служить функція ***eof()*** із наступним синтаксисом:

<імя файлової змінної>.eof()

Якщо кінець файлу не досягнуто, то вона повертає ***false***, а якщо досягнуто – ***true***. Наприклад, цикл зчитування даних (до досягнення кінця файлу) та їх сумування матиме наступний вигляд:

```
int s=0;
while (!F.eof())
{
//Дані читаються із файлу F та присвоюються змінній a
F>>a;
S=s+a;
}
```

Інші додаткові можливості потокових операцій роботи із файлами розглянуто в прикладах №5 та №6).

Приклади виконання завдання

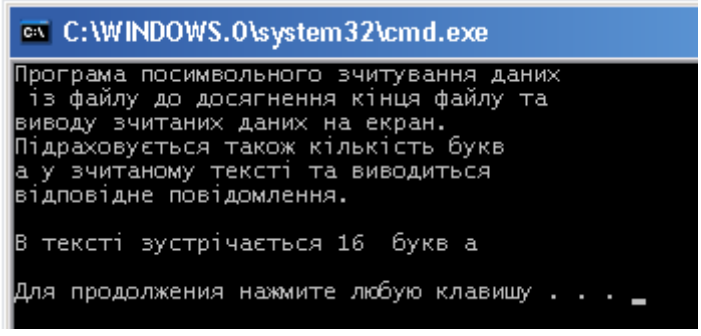
Приклад №1

Програма посимвольного зчитування даних із файлу до досягнення кінця файлу та виводу зчитаних даних на екран. Підраховується також кількість букв ***a*** у зчитаному тексті та виводиться відповідне повідомлення.

```
#include<stdio.h>
#include<string.h>
#include <windows.h>
void main()
{
```

```
        SetConsoleOutputCP(1251);
        SetConsoleCP(1251);
        char ch;
        int n=0;
        FILE *f1;
    /* Перевірка, чи вдалося успішно відкрити файл
Text1.in */
        if ((f1=fopen("Text1.in", "rt"))==NULL) {
        printf("Файл відкрити не вдалося\n");
        return;
        }
        int i=0;
        // Перевірка, чи вже досягнуто кінець файлу
        while (!feof(f1))
        {
        i++;
        // Читання символу із файлу
        ch=fgetc(f1);
/*Якщо досягнуто кінець файлу, то символ кінця файлу
не виводиться */
        if (!feof(f1))
        {
        printf("%c",ch);
        if (ch=='a') n++;
        }
        };
        // Закриття файлу
        fclose(f1);
        printf("\n");
printf("\nВ тексті зустрічається %i  букв a\n",n);
        printf("\n");
        }
```

Результат роботи програми:

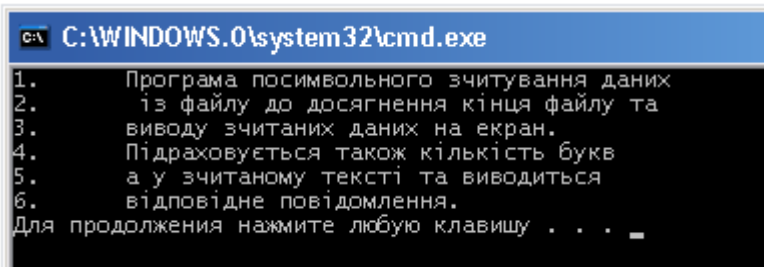


Приклад №2

Програма порядкового зчитування із файлу текстових даних до досягнення кінця файлу та їх виводу на екран (перед кожним зчитаним рядком також виводиться його номер):

```
        #include<stdio.h>
        #include<string.h>
        #include <windows.h>
        void main()
        {
        SetConsoleOutputCP(1251);
        SetConsoleCP(1251);
        char ch[80];
        FILE *f1;
/* Перевірка, чи вдалося успішно відкрити файл
Text1.in */
        if ((f1=fopen("Text1.in", "rt"))==NULL)
          {
        printf("Файл відкрити не вдалося\n");
        return;
        }
        int i=0;
        do {
        i++;
// Зчитування рядка довжиною 79 символів
        fgets(ch,80,f1);
        printf("%i.\t%s",i,ch);
// Перевірка, чи вже досягнуто кінець файлу
        } while (!feof(f1));
// Закриття файлу
        fclose(f1);
        printf("\n");
        }
```

Результат роботи програми:

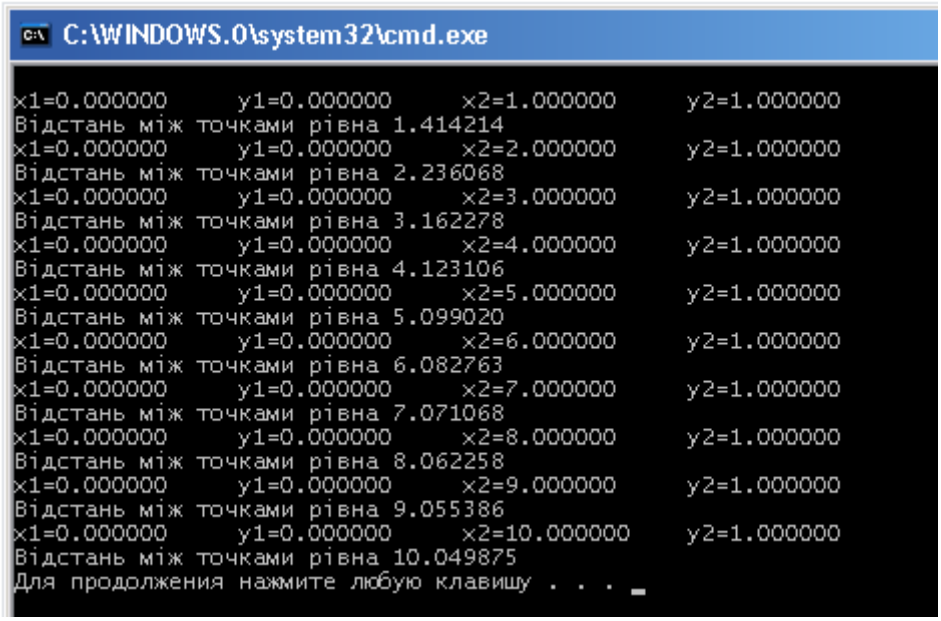


Приклад №3

Програма зчитування із файлу 10 координат точок на площині, обчислення відстані між ними та виводу результату на екран:

```
#include<stdio.h>
#include <windows.h>
#include <math.h>
void main()
{
SetConsoleOutputCP(1251);
SetConsoleCP(1251);
float x1,x2,y1,y2;
float r;
FILE *f1;
/* Перевірка, чи вдалося успішно відкрити файл
Text1.in */
if ((f1=fopen("Text1.in", "rt"))==NULL)
{
printf("Файл відкрити не вдалося\n");
return;
}
for(int i=0;i<10;i++)
{
/* Перевірка, чи вдалося успішно зчитати 4 дані із
файлу */
if (fscanf(f1,"%f%f%f%f",&x1,&y1,&x2,&y2)!=4)
{
printf("\nПомилка читання із файлу!");
break;
};
printf("\nx1=%f\ty1=%f\tx2=%f\ty2=%f\t",x1,y1,x2,y2);
r=sqrt(pow(x1-x2,2)+pow(y1-y2,2));
    printf("\nВідстань між точками рівна %f\t",r);
};
// Закриття файлу
fclose(f1);
printf("\n");
}
```

Результат роботи програми:



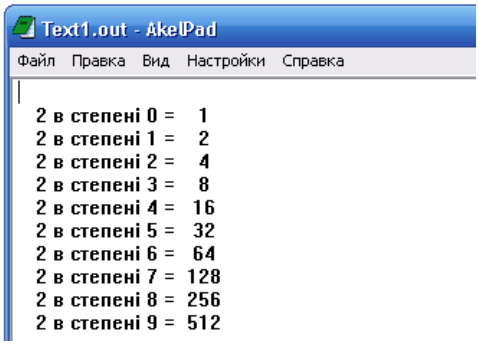
Приклад №4

Програма обчислення степені двійки від 2^0 до 2^9 та виводу результату у файл ***Text1.out***

```
        #include<stdio.h>
        #include <windows.h>
        #include <math.h>
        void main()
        {
        SetConsoleOutputCP(1251);
        SetConsoleCP(1251);
        FILE *f1;
/* Перевірка, чи вдалося успішно відкрити файл
Text1.out*/
        if ((f1=fopen("Text1.out", "wt"))==NULL)
        {
        printf("Файл відкрити не вдалося\n");
        return;
        }
        for(int i=0;i<10;i++)
        {
// Вивід даних у файл Text1.out
        fprintf(f1,"\n   2 в степені %i =
    %4.0f",i,pow(2.,i));
        }
```

```
        printf("\n");
// Закриття файлу Text1.out
        fclose(f1);
        }
```

В результаті роботи програми буде створено текстовий файл ***Text1.out***:



Приклад №5

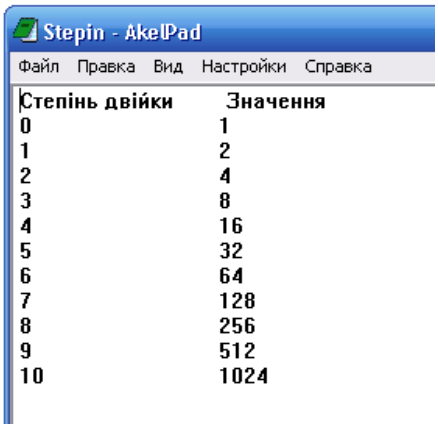
Програма обчислення степенів двійки та трійки та виводу результатів у файли ***Stepin*** та ***Stepin1*** відповідно (з використанням потокових операцій):

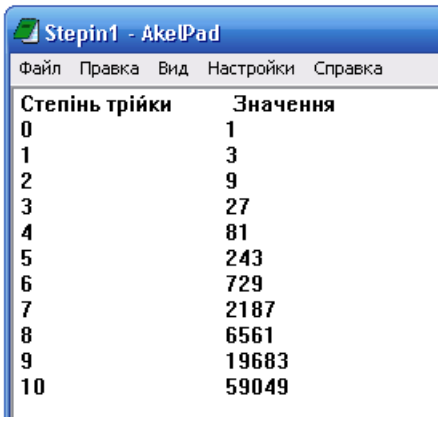
```
        #include<iostream>
        #include<fstream>
        #include <windows.h>
        #include<stdlib.h>
        #include <math.h>
        using namespace std;
        void main ()
        {
        SetConsoleOutputCP(1251);
        SetConsoleCP(1251);
        ofstream out;
        out.open ("Stepin",ios::out );
// Перевірка, чи вдалося успішно відкрити файл Stepin
        if ( ! out )
        {
        cout<< " Неможливо відкрити  файл \n ";
        exit (1);
        }
        ofstream out1;
```

```
// Відкриття файлу Stepin1
        out1.open("Stepin1");
// Перевірка, чи вдалося успішно відкрити файл
        if ( ! out1 )
        {
        cout<< " Неможливо відкрити  файл \n ";
        exit (2);
        }
        out<<"Степінь двійки\t "<<"Значення\n";
        for(int i=0;i<11;i++)
        out<<i<<"\t\t"<<pow(2.,i)<<"\n";
        out.close();
        out1<<"Степінь трійки\t "<<"Значення\n";
        for(int i=0;i<11;i++)
        out1<<i<<"\t\t"<<pow(3.,i)<<"\n";
        out1.close();
        }
```

В результаті роботи програми буде створено два текстові файли:

Stepin ***Stepin1***





Приклад №6

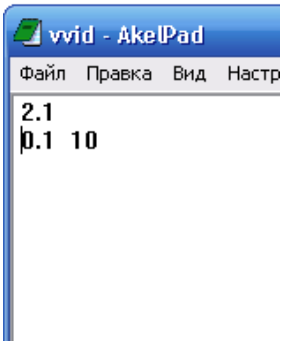
Програма із одного файлу зчитує інформацію про початкову точку, крок та кількість точок табуляції, а в другий виводить результат табулювання функції $f(x)=\sin(x)+\cos(x)$.

```
#include<iostream>
#include<fstream>
#include<stdlib.h>
#include <math.h>
```

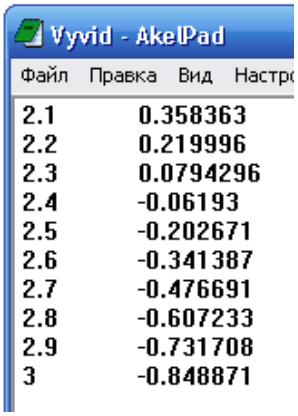
```
#include <windows.h>
using namespace std;
void main ()
{
float x0;
float h;
int n;
SetConsoleOutputCP(1251);
SetConsoleCP(1251);
ifstream in( "vvid" );
ofstream out( "Vyvid" );
// Перевірка, чи вдалося успішно відкрити файл Vyvid
if ( ! out )
{
cout<< " Неможливо відкрити  файл \n ";
exit (1);
}
// Перевірка, чи вдалося успішно відкрити файл vvid
if ( ! in ) {
cout<< " Неможливо відкрити  файл \n ";
exit (1);
}
in>>x0;
in>>h>>n;
for(int i=0;i<n;i++,x0=x0+h)
// Вивід даних у файл Vyvid
out<<x0<<"\t"<<sin(x0)+cos(x0)<<"\n";
out.close();
in.close();
}
```

Результат роботи програми:

Вхідний файл ***vvid***



Результат роботи програми – файл ***Vyvid***



Завдання для виконання

Написати програму розв'язку задач двома способами: із використанням файлових вказівників та використовуючи потокові операції. Перевірити правильність роботи програми за допомогою контрольних прикладів. Результат вивести з використанням пояснювальної текстової інформації.

1. Файл містить 100 цілих чисел. Підрахувати, скільки серед них є трійок та скільки чисел більші десяти.
2. Файл містить 50 дійсних чисел. Підрахувати, скільки серед них є від'ємних чисел (вивести на екран). Вивести у інший файл номери позицій, в яких знаходяться від'ємні числа та їх значення.
3. Написати програму, що зчитує із файлу результати деякого експерименту, обчислює середнє значення, абсолютну та відносну похибки вимірювання і результат дописує у даний файл.
4. Написати програму, яка із даного файлі із дійсними числами формує два нові. Один із них містить всі від'ємні числа та їх кількість, другий - всі додатні та їх кількість.
5. Написати програму, яка із даного файлу із цілими числами формує два нові. Один із них містить позиції всіх чисел, більших 3, другий – позиції всіх нульових елементів.
6. Написати програму, яка обчислює середнє арифметичне чисел, що записані у файлі. Якщо середнє арифметичне більше нуля, то отриманий результат записується у файл plus.rez, а якщо менше нуля – minus.rez.
7. Написати програму, яка із даного файлу дійсних чисел видаляє всі, які більші деякого числа A, але менші B та підраховує кількість таких елементів.
8. Написати програму, що створює копію файлу, але записує числові дані задом наперед (першим стоїть останній елемент і т. д.).
9. Написати програму, що створює копію файлу, який містить дійсні числа, але спочатку ідуть додатні елементи, потім нульові, а в кінці від'ємні.
10. Написати програму, що створює копію файлу із даними цілого типу, вирізавши із нього всі елементи, рівні нулю.
11. Написати програму, яка сортує дійсні числа, зчитані із файлу по зростанню та виводить їх в інший файл.

12. Дано масив 100 цілих чисел. Перевірити, чи впорядкований даний масив по зростанню (спаданню, невпорядкований) та вивести відповідне повідомлення на екран та у файл.
13. Написати програму, яка із даного файлу, що містить дійсні числа формує новий, в якому сусідні елементи поміняно місцями.
14. Файл s.dani містить 25 цілих чисел, записаних у рядок. Вивести ці числа у новий файл, де вони будуть записані у стовпчик.
15. Створити файл, який містить табличку множення чисел від одного до 9, сформовану у вигляді матриці.
16. У файлі ds.dat записано матрицю цілих чисел розміром 5×5 елементів. Транспонувати дану матрицю та вивести її у новий файл.
17. Написати програму табулювання функції $sh(x)=\frac{e^x-e^{-x}}{2}$ із виводом результату у файл. Всі дані для табуляції вводяться із клавіатури (задати самостійно).
18. Сформувати файл, який містить таблицю значень функції $n!$ з використанням форматного виводу (значення n та відповідне йому значення $n!$). Значення n читається із попередньо підготовленого файлу.
19. У файл rjad.rez виводити у вигляді таблички (значення і – відповідне йому значення члену ряду) результат обчислення членів ряду $\sum_{i=1}^{\infty} i^{-3}$ до тих пір, поки член ряду не стане меншим 0.00001. У інший файл вивести кількість члені ряду, яка була обчислена із пояснювальною текстовою інформацією.
20. У одному файлі записано інформацію про координати 10 точок у просторі, а в іншому координати центу кулі та її радіус. У новий файл вивести інформацію про координати точок, що лежать в середині кулі. На екран вивести кількість таких точок
21. Із клавіатури вводиться цілі числа. Числа, кратні 3, записуються у один файл, а кратні 7 – у інший. Якщо сумарна кількість записаних у файл чисел стане рівною 20 – завершити роботу програми.
22. Файл містить дані про результати 25 експериментальних вимірювань. Переписати їх у новий файл, відкинувши найбільший та найменший результат.
23. Файл містить результати 50 вимірювань. Вивести у новий файл найбільший та найменший результат та їх порядкові номери.

24. Катети прямокутного трикутника вводяться із клавіатури. У файл вивести гіпотенузу, площу трикутника та його кути із пояснювальною текстовою інформацією. Передбачити можливість проведення обчислень для кількох різних значень (задається із клавіатури) довжин катетів.
25. Файл містить 100 цілих чисел. Дописати у вихідний файл загальну суму чисел. Знайти суми цифр цих чисел, а результати записати у інший файл.
26. У файлі записана матриця $B_{7\times7}$. Обчислити добуток ненульових елементів та дописати у вихідний файл. У новий файл вивести транспоновану матрицю.
27. У двох файлах записано матриці $A_{7\times7}$ та $C_{7\times7}$. Обчислити суму цих матриць та записати сумарну матрицю у новий файл та вивести її на екран.
28. У файлі записано інформацію про координати вершин 20 прямокутників на площині. У новий файл вивести кординати найбільшого та найменшого за площею прямокутників та їх порядкові номери у вихідному файлі.
29. У файлі записано інформацію про 10 варіантів довжин 3 відрізків. У новий файл вивести інформацію про те, чи можна із даних відрізків побудувати трикутник, чи ні.
30. У файлі записано інформацію про координати 100 точок на площині. У новий файл вивести інформацію про відстань від точки до початку координат та те, в якому квадранті системи координат вони знаходяться.

Лабораторна робота №10

Тема роботи: Робота із рядковими та символьними змінними

Мета роботи: Формування навиків роботи із рядковими змінними та розробки алгоритмів їх обробки.

Для виконання роботи необхідно знати:

- опис рядкових та символьних змінних;
- функції обробки символьних змінних;
- функції обробки рядкових змінних;
- алгоритми обробки рядкових змінних.

Теоретичні відомості

Рядок - це одномірний масив символів в коді ASCII, що закінчується нульовим символом. Кожний символ рядка можна вибрати по значенню індексу з допомогою оператора циклу, а кінець рядка визначається по значенню ***'\0'***- нульовий стмвол. Наявність нульового символу означає, що кількість комірок масиву повинна бути принаймі на 1 більше, ніж число символів, які необхідно розмістити в пам'яті.

Необхідно розрізняти одиничний символ (символьна змінна), який записується як ***'a'***, ***'b'*** і т.д., та рядок ***"a"***, ***"b"*** (рядкова змінна). Рядкова змінна може бути ініціалізована декількома способами:

1. ```
   char st[40]
      st= "Це рядок";
   ```
2. ```
   char st1[]= " Це рядок";
   ```
3. ```
   char st3[100]
      gets(st3);  // ввід з клавіатури
   ```
4. ```
   char str4[100]
      scanf("%s",str4); // ввід з клавіатури
   ```

Для вводу-виводу символьних рядків служать наступні функції (заголовочний файл ***stdio.h***) :

gets()– ввід рядка символів із клавіатури;

scanf()– ввід рядка символів із клавіатури (із форматом ***%s***);

puts()– вивід символьного рядка на екран;

printf()– вивід символьного рядка на екран (із форматом ***%s***).

Для роботи із рядковими змінними служить клас функцій, що знаходяться в заголовочному файлі ***string.h*** (всі вони починаються

префіксом ***str***) (див. додаток №8). Для роботи із символьними даними служить клас функцій, що знаходяться в заголовочному файлі ***ctupe.h*** (всі вони починаються префіксом ***is***) (див. додаток №9).

Більшість задач обробки рядкових зміних зводяться до аналізу масивів даних з використанням операторів циклу.

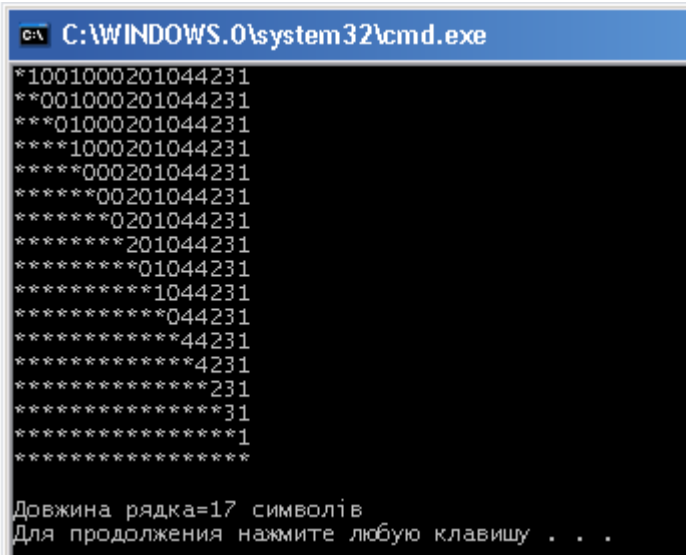
Приклади виконання завдання

Приклад №1

Програма послідовно, в циклі, міняє символи рядка на *******, а в кінці роботи виводить довжину символьного рядка.

```
#include <stdio.h>
#include <conio.h>
#include <windows.h>
void main()
{
SetConsoleOutputCP(1251);
SetConsoleCP(1251);
// Ініціалізація символьної змінної
char s[]="1100100020104423 1";
int i=0;
for (i=0;i<strlen(s);i++)
{s[i]='*';printf("%s\n" , s);}
printf("\nДовжина рядка=%i символів",i);
printf("\n");
}
```

Результат роботи програми:

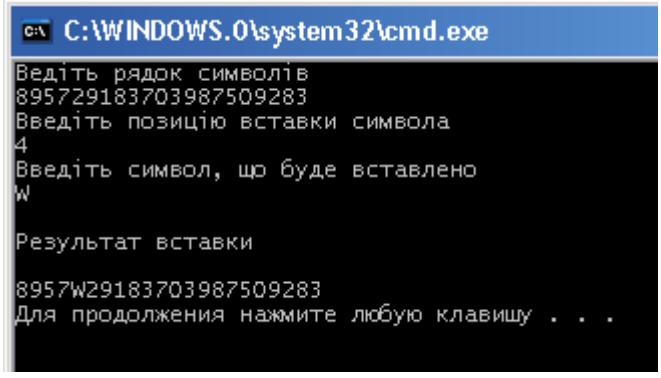


Приклад №2

Програма вставляє заданий із клавіатури символ у вказане місце, зсуваючи всі символи, що стоять лівіше нього на одну позицію. Для вставки використовується окрема функція (***insert***).

```
#include <stdio.h>
#include <conio.h>
#include <string.h>
#include <windows.h>
void insert (char *str, int p, char c)
{
int i;
for (i=strlen(str);i>=p;i--)
str[i+1]=str[i];
str[p]=c;
}
void main()
{char c,s[100];
int n;
SetConsoleOutputCP(1251);
SetConsoleCP(1251);
puts("Ведіть рядок символів");
gets(s);
puts("Введіть позицію вставки символа");
scanf("%i",&n);
puts("Введіть символ, що буде вставлено");
c=getch();
printf("%c\n",c);
insert(s,n,c);
puts("\nРезультат вставки\n");
puts(s);
}
```

Результат роботи програми:



Приклад №3

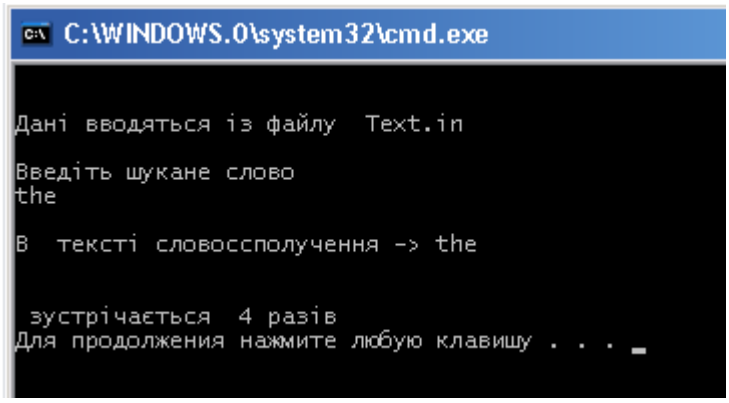
Програма підраховує, скільки разів дане слово зустрічається у тексті. Слово вводиться із клавіатури, а текст читається із файлу ***Text.in***.

Примітка: *оскільки посимвольна обробка українського алфавіту в консольному режимі може приводити до помилок на етапі виконання програми, використовуємо або український текст, набраний англійськими буквами, або текст на англійській мові.*

```
#include<stdio.h>
#include<string.h>
#include <windows.h>
#include <ctype.h>
#include<iostream>
using namespace std;
void main()
{
SetConsoleOutputCP(1251);
SetConsoleCP(1251);
char ch;
char word[20];/* опис рядкової змінної для
вводу слова*/
char line [10000];/* опис рядкової змінної
для вводу тексту */
int count=0;
cout << "\n";
cout << "\nДані вводяться із файлу  Text.in\n";
int n=0;
FILE *f1;
/*Перевірка, чи вдалося відкрити
файл Text.in */
if ((f1=fopen("Text.in", "rt"))==NULL) {
printf("Файл відкрити не вдалося\n");
return;
}
cout << "\nВведіть шукане слово \n";
scanf("%s",word);
int i=0;
while (!feof(f1))
/*Посиммольне зчитування тексту із файлу до
досягнення кінці файлу*/
```

```
        {
        ch=fgetc(f1);
        line[i]=ch;
        i++;
        };
        line[i-1]='\0'; /*в кінець рядка ставимо 0-
    символ */
        int len;
        //Визначення довжини введеного слова
        len=strlen(word);
        char *p=line;
        //пошук введеного слова у рядку
        while(p=strstr(p,word))
        {
        char * c=p;
        // Перехід вказівника p через слово
        p=p+len;
/* Чи є перед словом та після нього розділювачі
(пробіли або розділові знаки ). Інакше – перейти до
наступної ітерації */
        // Слово знаходиться на початку рядка
        if(c==line)
        {
        if (ispunct(*p)|| isspace(*p)|| (*p=='\0'))
    count ++;
        }
        else
        // Слово знаходиться в середині  рядка
        if ((ispunct(*p)|| isspace(*p)||
    (*p=='\0'))&& (!ispunct(*(c-1))&& isspace(*(c-
    1)))) count ++;
        }
        // Закриття файлу
        fclose(f1);
        printf("\n");
    printf("В  тексті словосполучення -> %s\n",
    word);
        printf("\n");
        printf("\n зустрічається  %i разів\n",count);
        }
```

Результат роботи програми:



Завдання для виконання

Написати програму розв'язку задачі. Розробити контрольні приклади та протестувати за їх допомогою програму.

1. Рядок складається із груп нулів та одиниць, розділених одним або декількома пробілами. Знайти кількість груп, що містять п'ять або шість символів.
2. Речення складається із деякої кількості слів, розділених одним або декількома пробілами. Видалити із речення лишні пробіли: там де їх більше одного.
3. Рядок складається із груп цифр, розділених довільною кількістю пробілів. Знайти та вивести на екран найкоротшу групу та кількість груп такої довжини.
4. Речення складається із деякої кількості слів, розділених пробілами. Знайти та вивести на екран найкоротше та найдовше слово.
5. Речення складається із деякої кількості слів, розділених пробілами. Знайти та вивести на екран найкоротше слово, що має парну кількість символів. Підрахувати та вивести на екран також кількість букв у реченні.
6. Речення складається із деякої кількості слів, розділених пробілами. Знайти та вивести на екран найдовше слово, що має непарну кількість символів, але не більше 5. Вивести також кількість слів у реченні.
7. Рядок складається із цифр, арифметичних операцій та деякої кількості відкриваючих та закриваючих дужок. Перевірити, чи кількість відкриваючих дужок рівна кількості закриваючих дужок.

8. Рядок символів складається із довільних десяткових цифр в межах від 0 до 9, розділених довільною кількістю пробілів. Знайти суму цих цифр.
9. Рядок символів складається із довільних десяткових цифр, розділених пробілами. Знайти та видалити із рядка всі парні цифри, а новий рядок вивести на екран.
10. Рядок символів складається із довільних чисел, розділених пробілами. Знайти найбільше та найменше число.
11. Рядок символів складається із чисел, розділених пробілами. Вивести на екран ці числа у порядку зростання.
12. Рядок символів складається із десяткових цифр, розділених пробілами. Підрахувати та вивести на екран кількість парних та кількість непарних цифр.
13. Рядок символів складається із десяткових цифр, розділених знаками додавання та віднімання. Обчислити значення арифметичного виразу а результат вивести на екран.
14. Речення складається із деякої кількості слів, розділені пробілами. Вивести на екран окремі слова, впорядкувавши їх у алфавітному порядку.
15. Речення складається із деякої кількості слів, розділених пробілами та певною кількістю ком. Підрахувати кількість ком у реченні та перевірити, чи закінчується воно крапкою.
16. Із клавіатури вводиться рядок символів − ім’я тригонометричної функції та деяке числове значення аргументу(наприклад sin(1)). Розпізнати тригонометричну функцію та обчислити її значення.
17. Речення складається із деякої кількості слів, розділених пробілами та певною кількістю ком. Знайти в реченні введене із клавіатури слово та замінити його іншим, заданим із клавіатури словом. Результат вивести на екран.
18. Дано рядок непарної кількості символів. Дзеркально відобразити рядок відносно центрального символу. Результат вивести на екран.
19. Дано рядок символів: два речення, розділені крапкою. Поміняти речення місцями та вивести результат на екран.
20. Дано довільний рядок символів. Вивести на екран монітора символи, які зустрічаються найменшу та найбільшу кількість раз.
21. Речення складається із деякої кількості слів, розділених пробілами та певною кількістю ком. Після другої коми добавити задану із кла-

віатури кількість додаткових пробілів. Результат вивести на екран.

22. Рядок символів складається із довільних десяткових цифр в межах від 1 до 9, розділених пробілами. Якщо перше число парне, то виконати циклічний зсув на дві позиції, а якщо непарне – на три.
23. Речення складається із деякої кількості слів, розділених пробілами та певною кількістю ком. Вивести на екран всі слова, що починаються заданою із клавіатури буквою та після яких стоїть кома.
24. Рядок символів складається із довільних чисел, розділених пробілами. Обчислити значення даного виразу, якщо із клавіатури вводиться по порядку арифметичні операції додавання та віднімання між даними числами.
25. Речення складається із деякої кількості слів, розділених довільною кількістю пробілів. Видалити всі пробіли а результат вивести на екран монітора. Підрахувати кількість видалених пробілів.
26. Речення складається із деякої кількості слів. Поміняти сусідні слова місцями. Результат вивести на екран. Вивести також кількість виконаних операцій обміну слів місцями.
27. Речення складається із деякої кількості слів. Замінити слова значеннями довжин слів, із яких воно складається. Результат вивести на екран.
28. Речення складається із деякої кількості слів, розділених пробілами та певною кількістю ком. Вивести на екран всі слова, що починаються заданим із клавіатури сполученням двох букв.
29. Рядок складається із деякої кількості слів – цифр в межах від 1 до 9. Замінити ці слова відповідними числовими значеннями та підрахувати їх суму.
30. Рядок є набором деякої кількості десяткових цифр в межах від 0 до 9, розділених словами «додати та «відняти»». Замінити слова відповідними арифметичними операціями та обчислити значення виразу. Результат вивести на екран.

Лабораторна робота 11

Тема роботи: Програмування з використанням структур

Мета роботи: Формування навиків роботи із структурами та розробка алгоритмів їх обробки

Для виконання роботи необхідно знати:

- оголошення структур;
- використання структурованих даних;
- алгоритми обробки структурованих даних;
- доступ до полів структури;
- масиви структур.

Теоретичні відомості

Структура – це складений тип даних, в якому під одним іменем об'єднано дані різного типу. Окремі дані структури називають полями. Оголошення структурного типу даних виконують за допомогою ключового слова ***struct*** за наступним синтаксисом:

```
struct <ім'я>
{
Тип_елементу_1     ім'я_елементу_1;
Тип_елементу_2     ім'я_елементу_2;
                .   .   .
Тип_елементу_n     ім'я_елементу_n;
};
```

де:

<ім'я> - ім'я структурного типу даних.

Оголошення структури є оператором, тому в кінці її опису обов'язково повинна стояти крапка з комою.

На відміну від інших типів даних опис структури не виділяє місця у пам'яті під збереження елементів структури. Опис визначає лише шаблон, що задає характеристики змінних структури. Для введення змінних типу структура необхідно або після фігурної дужки, що завершує опис структури, вказати список відповідних ідентифікаторів структури, або окремо оголосити змінну як це звичайно робиться. При цьому структурою може бути як звичайна змінна, так і будь-який масив (масив структур) Наприклад:

1. Оголошення структури ***date*** та опис структурних змінних:

```
// Оголошення структури
struct date {
int day ;
int month ;
int year;
char day_name[15];
char mon_name[14];
}
// Опис структурних змінних
den,dat[100],*sp;
```

2. Оголошення структури ***date*** та окреме оголошення структурних змінних:

```
// Оголошення структури
struct date {
int day ;
int month ;
int year;
char day_name[15];
char mon_name[14];
}
. . . .
. . . .
// Опис структурних змінних
date den,dat[100],*sp;
```

Правила роботи із полями структури аналогічні роботі із змінними відповідного типу. До полів структури можна звертатися через складене ім'я згідно наступного формату:

прямий селектор :

Імя_структури.імя_поля.

Наприклад: доступ до другого поля (***month***) другого елементу масиву структури ***date***:

```
dat[1].month=9
```

доступ до третього поля структури ***den***:

```
den.year =2011
```

непрямий селектор:

```
Вказівник_на_структуру ->імя_поля
```

Наприклад доступ до другого поля (***month***):

```
Sp-> month=9
```

Для змінних одного і того ж самого структурного типу визначено операцію присвоєння. При цьому виконується копіювання значень відповідних полів: наприклад (для описаної вище структури):

```
den = dat[1]
```

Для порівняння двох структур необхідно порівняти відповідні поля цих структур. Обробка структурних даних в цілому аналогічна обробці масивів даних. При цьому тільки необхідно враховувати, що у різних полях структури можуть знаходяться дані різного типу.

Змінна структурного типу може бути як глобальною, так і локальною. Допустимим є використання структури чи її окремих полів як параметра функції. Можна також використовувати вказівники на структуру.

Приклади виконання завдання

Приклад №1

Створити масив структур, що мітить наступну інформацію про студента: прізвище; ім’я; по батькові; номер групи; чотири оцінки здачі сесійних екзаменів; середній бал за результатами сесії. Дані структури зчитуються із файлу (створеного за допомогою Блокнота). Вивести на екран інформацію про середній бал студентів групи М21 у порядку спадання.

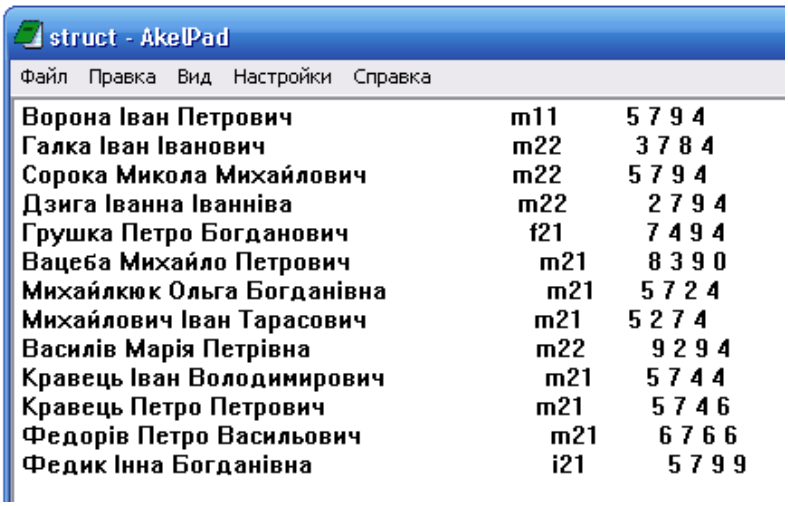
```
#include <iostream>
#include<stdio.h>
#include <fstream>
#include <string.h>
#include <windows.h>
using namespace std;
void main()
{
struct strc{
char f[15];
char i[15];
char o[15];
char ngr[7];
int otc[4];
double sb;
}
ms[1000];
int nst,i,j;
```

```
        SetConsoleOutputCP(1251);
        SetConsoleCP(1251);
        FILE *f1;
/* Відкриття файлу для вводу даних та перевірка, чи
вдалося його відкрити */
        if ((f1=fopen("struct", "rt"))==NULL)
        {
        printf("Файл відкрити не вдалося\n");
        return;
        }
        cout<<"\n";
        i=0;
// Читання даних з файлу до досягнення його кінця
        while (!feof(f1))
        {
        i++;
// Форматний ввід даних з файлу
    fscanf(f1,"%s%s%s%s%i%i%i%i",ms[i].f,ms[i].i,ms[i
    ].o,ms[i].ngr
    ,&ms[i].otc[0],&ms[i].otc[1],&ms[i].otc[2],&ms[i]
    .otc[3]);
// Обчислення середнього балу
    ms[i].sb=(ms[i].otc[0]+ms[i].otc[1]+ms[i].otc[2]+
    ms[i].otc[3])/4.;
        }
        nst=i;
cout<<"\n Загальна кількість студентів ="<<nst<<"\n";
        cout<<"\n";
cout<<"\nСтуденти групи m21 у порядку спадання
середнього балу\n";
/* Впорядкування студентів групи m21 у порядку
спадання */
        for(i=0;i<nst-1;i++)
        for(j=i+1;j<nst;j++)
        if(ms[i].sb<ms[j].sb
        && !strcmp(ms[i].ngr,"m21")
        && !strcmp(ms[j].ngr,"m21"))
        {
        ms[999]=ms[i];
        ms[i]=ms[j];
        ms[j]=ms[999];
        }
        int k=0;
```

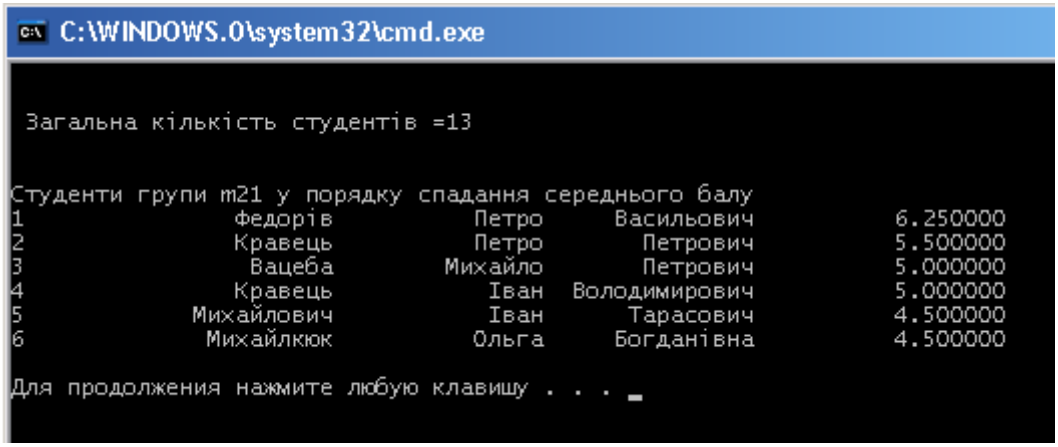
```
/* Перегляд масиву структури та вивід на екран тільки
студентів групи m21 */
        for(i=0;i<nst;i++)
        if(!strcmp(ms[i].ngr,"m21"))
        {k++;
        printf("%i\t%15s%15s%15s\t%15.6f\n",k,ms[i].f
    ,ms[i].i,ms[i].o,ms[i].sb);
        }
        cout<<"\n";
        fclose(f1);
        }
```

Результат роботи програми:

Вхідний файл ***struct*** для вводу даних:



Результат роботи програми:



Завдання для виконання

Написати програму розв'язку задачі. Розробити контрольні приклади та протестувати за їх допомогою програму. Вхідні дані оформити у вигляді окремого файлу.

1. Масив структур мітить інформацію про 23 студентів: прізвище, ім'я та по батькові; стать;номер групи; оцінка за здачу екзамену з інформатики в стобальній системі. Зчитати дані із файлу та вивести прізвища студентів чоловічої статі, що мають відмінну оцінку.
2. Масив структур мітить інформацію про 25 людей: прізвище; ім'я; по батькові; стать; сімейний стан; відомості про кількість дітей. Зчитати дані із файлу та вивести прізвища всіх жонатих чоловіків, що мають дітей.
3. Масив структур мітить інформацію про 25 клієнтів пункту прокату: прізвище; ім'я; домашня адреса; номер телефону; назва речі, взятої напрокат. Зчитати дані із файлу та вивести прізвища та телефони всіх клієнтів , що взяли на прокат телевізор.
4. Масив структур мітить інформацію 22 клієнтів пункту прокату: прізвище; ім'я; назва речі, взятої напрокат; домашня адреса; номер телефону. Зчитати дані із файлу та вивести прізвища всіх клієнтів, телефони яких починаються цифрою 7.
5. Масив структур мітить інформацію про 25 клієнтів пункту прокату: прізвище; ім'я; назва речі, взятої напрокат; домашня адреса; номер телефону. Зчитати дані із файлу та вивести прізвища всіх клієнтів, що проживають на введеній із клавіатури вулиці.
6. Масив структур мітить інформацію про 30 студентів: прізвище; ім'я; місце проживання (гуртожиток або дома); номер телефону (якщо він є). Зчитати дані із файлу та вивести прізвища тих студентів, що проживають дома та мають домашній телефон.
7. Масив структур мітить інформацію про 30 студентів: прізвище; ім'я; номер групи; місце проживання (гуртожиток або дома). Зчитати дані із файлу та вивести прізвища студентів заданої із клавіатури групи, що проживають у гуртожитку.
8. Масив структур мітить інформацію про 30 студентів: прізвище; ім'я; стать; номер групи; місце проживання (гуртожиток або дома). Зчитати дані із файлу та підрахувати загальну кількість студентів жіночої статі, що проживають дома.

9. Масив структур мітить інформацію про 30 студентів: прізвище; ім’я; номер групи; адреса прописки; стать. Зчитати дані із файлу та підрахувати кількість студентів, що прописані на заданій із клавіатури вулиці.
10. Масив структур мітить інформацію про 25 студентів: прізвище; ім’я; номер групи; адреса прописки, стать. Зчитати дані із файлу та підрахувати загальну кількість студентів чоловічої та жіночої статі.
11. Масив структур мітить інформацію про 10 автомобілів: модель; вартість; рік випуску, розхід пального на 100 кілометрів шляху. Зчитати дані із файлу та вивести автомобілі, які не старші 5 років та мають розхід пального, що не перевищує вказану із клавіатури величину.
12. Масив структур мітить інформацію про 12 автомобілів: модель; рік випуску;вартість; розхід пального на 100 кілометрів шляху. Зчитати дані із файлу та вивести автомобілі, вартість яких не перевищує вказану із клавіатури величину, а розхід пального менше 7 літрів на 100 кілометрів.
13. Масив структур мітить інформацію про 15 міст України: назва міста; кількість населення; рік заснування; середньорічна температура. Зчитати дані із файлу та вивести назви міст, старших 300 років, кількість населення в яких перевищує введену із клавіатури величину.
14. Масив структур мітить інформацію про 12 міст України: назва міста; кількість населення; середньорічна температура. Зчитати дані із файлу та впорядкувати дані по спаданню середньорічої температури. Вивести міста, в яких вона не перевищує введену із клавіатури величину.
15. Масив структур мітить інформацію про 20 людей: прізвище; ім’я; стать; вага; ріст. Зчитати дані із файлу і вивести на екран середній ріст людей чоловічої статі.
16. Масив структур мітить інформацію про 22 людей: прізвище; ім’я; стать; ріст; вага. Зчитати дані із файлу і вивести на екран прізвища осіб жіночої статі, вага яких перевищує середнє значення.
17. Масив структур мітить інформацію про 17 людей: прізвище; ім’я; стать; ріст; вага. Зчитати дані із файлу та вивести на екран прізвища всіх осіб, вага яких не перевищує норму (ріст – вага < 100).
18. Масив структур мітить інформацію про 26 людей: прізвище; ім’я; стать; ріст; вага. Зчитати дані із файлу та вивести на екран прізвища

всіх осіб чоловічої статі, вага та ріст яких перевищує введені із клавіатури значення.

19. Масив структур мітить інформацію про 15 працівників фірми: прізвище; ім’я; рік народження; рік, із якого працює на фірмі; оклад. Зчитати дані із файлу і вивести на екран прізвища всіх осіб жіночої статі, що працюють на фірмі не менше 2 років та мають оклад, що не перевищує середнього по фірмі.
20. Масив структур мітить інформацію про 15 працівників фірми: прізвище; ім’я; по батькові; оклад; рік народження;. Зчитати дані із файлу і вивести на екран прізвища всіх працівників, оклад яких перевищує середнє по фірмі значення.
21. Масив структур мітить інформацію про 25 жителів населеного пункту: прізвище; ім’я; адреса проживання; номер телефону. Зчитати дані із файлу та вивести номер телефону людини, прізвище якої введено із клавіатури. Якщо такої людини немає, то видати відповідне повідомлення.
22. Масив структур мітить інформацію про 23 жителів населеного пункту: адреса проживання; прізвище; ім’я; номер телефону. Зчитати дані із файлу та вивести на екран прізвище людини за введеним із клавіатури номером телефону. Якщо такого номера телефону немає, то видати відповідне повідомлення.
23. Масив структур мітить інформацію про 35 абітурієнтів: прізвище; ім’я; адреса проживання; три оцінки за зовнішнє незалежне оцінювання. Зчитати дані із файлу та вивести прізвища та адреси абітурієнтів, середній бал яких перевищує середній по університету.
24. Масив структур мітить інформацію про 35 абітурієнтів: спеціальність; прізвище; ім’я; адреса проживання; три оцінки за зовнішнє незалежне оцінювання. Зчитати дані із файлу та вивести прізвище та стать абітурієнтів, впорядкувавши їх по спаданню середнього балу – від найбільшого до найменшого.
25. Масив структур мітить інформацію про 35 абітурієнтів: прізвище; ім’я; три оцінки за зовнішнє незалежне оцінювання; стать. Зчитати дані із файлу та вивести прізвища та стать абітурієнтів, впорядкувавши їх по спаданню середнього балу – від найбільшого до найменшого. На початку розмістити всіх пільговиків.
26. Масив структур мітить інформацію про відправлення поїздів: номер поїзда; пункт призначення; час відправлення; кількість вільних місць. Зчитати дані із файлу та вивести номери тих поїздів, на які є не менше 25 вільних місць.

27. Масив структур мітить інформацію про виліт літаків: номер рейсу; пункт призначення; час вильоту; наявність вільних місць (є або нема). Зчитати дані із файлу та вивести всі номери рейсів та пунктів призначення, на які є вільні місця, впорядкувавши їх за зростанням часу вильоту.
28. Масив структур мітить інформацію про каталог книг: реєстраційний номер книги; назва; автор; видана, чи знаходиться у книгосховищі. Зчитати дані із файлу та вивести всі книги автора, прізвище якого введене із клавіатури якщо вона не видана.
29. Масив структур мітить інформацію про студентів: прізвище; ім’я; номер групи; середній дохід на члена сім’ї. Зчитати дані із файлу та вивести всіх студентів, у яких середній дохід не перевищує заданої з клавіатури величини, впорядкувавши їх у порядку його спадання.
30. Масив структур мітить інформацію про каталог книг: реєстраційний номер книги; видана, чи знаходиться у книгосховищі; якщо видана, то якого числа; прізвище студента, якому видана книга. Зчитати дані із файлу та вивести всіх студентів, яким книги видані більше терміну, заданого із клавіатури.

Лабораторна робота 12

Тема роботи: Програмування класів

Мета роботи: Формування навиків роботи із класами, алгоритмами їх оголошення та обробки.

Для виконання роботи необхідно знати:

- поняття класу та його призначення;
- оголошення класу;
- доступ до відкритих та закритих членів класу;
- визначення елементів-функцій класу;
- алгоритми роботи із класами.

Теоретичні відомості

Клас – визначений користувачем тип даних (змінні будь-якого типу, інші класи чи вказівники), який створюється для опису абстрактної множини об'єктів – членів класу. При цьому він інкапсулює в себе елементи-дані (члени) та елементи-функції (методи). Елементи-дані задають властивості об'єктів (описуються за допомогою полів класу). Елементи-функції задають операції, які можна виконувати над елементами-даними класу.

В цілому поняття класу нагадує поняття структури в C++ за виключенням ряду моментів:

- він містить в собі ряд специфікацій доступу до об'єктів класу;
- як правило він включає в себе елементи - функції, що задають правила обробки об'єктів класу;
- для створення-знищення об'єктів класу використовуються спеціальні функції – конструктор та деструктор.

На відміну від полів структури, які є доступними завжди, в класі члени та методи можуть мати різний рівень доступу. До відкритих полів (**`public`**) доступ дозволений будь-кому. До закритих (***`private`***) та захищених (***`protected`***) полів доступ ззовні класу є закритий, щоби не порушити цілісність даних. Доступ до таких полів мають тільки елементи-функції класу.

Оголошення класу виконується за наступним синтаксисом:

```
class <ім'я класу>
    {
```

```
private:
<закриті елементи-дані >
<закриті елементи-функції>
protected:
<захищені елементи-дані >
<захищені елементи-функції>
public:
<відкриті елементи-дані >
<відкриті елементи-функції>
}
```

<ім'я класу> утворюється за правилами утворення будь-яких інших ідентифікаторів. По замовчуванню для всіх елементів класу діє закритий режим доступу.

Для прикладу створимо клас **complex** для роботи із комплексними числами. В ньому будуть:

члени класу:

double x – дійсна частина комплексного числа;

double y – уявна частина комплексного числа.

методи класу:

double modul() – функція, що обчислює модуль комплексного числа;

double argument() – функція, що обчислює аргумент комплексного числа;

void show_complex() – функція, що виводить комплексне число на екран;

Оголошення класу матиме наступний вигляд:

```
class complex
{
public:
double x;
double y;
double modul()/*обчислення модуля
комплексного числа */
{
double temp;
temp=pow(x*x+y*y,1/2.0);
return temp;
}
double argument()/*обчислення аргумента
комплексного числа */
```

```
{
double temp;
temp=atan2(y,x)*180/pi;
return temp;
}
void show_complex() /*вивід комплекснеого
числа на екран */
{
if(y>=0) cout<<x<<"+"<<y<<"i"<<endl;
else cout<<x<<"-"<<y<<"i"<<endl;
}
};
```

Функції-члени можуть бути визначені як безпосередньо при описі класу, так і в будь якому місці за межами класу, так як це зроблено нижче. Визначення функції-члену починається із імені класу, після якого через операцію *::* ім'я функції та її параметри:

```
class complex
{
public:
double x;
double y;
double modul();
double argument();
void show_complex();
};
//обчислення модуля комплексного числа
double complex::modul()
{
double temp;
temp=pow(x*x+y*y,1/2.0);
return temp;
};
//обчислення аргумента комплексного числа
double complex::argument()
{
double temp;
temp=atan2(y,x)*180/pi;
return temp;
};
 //вивід комплекснеого числа на екран
```

```
void complex::show_complex()
{
if(y>=0) cout<<x<<"+"<<y<<"i"<<endl;
else cout<<x<<"-"<<y<<"i"<<endl;
};
```

В такому описі і елементи-члени, і елементи-функції є відкритими (**public**). Доступ до відкритих членів класу може бути здійснений прямим зверненням за допомогою оператора «.» згідно наступного синтаксису:

```
<ім'я класу>.<ім'я поля класу>
<ім'я класу>.<ім'я методу класу>
```

Наприклад:

```
void main()
{
complex chislo; //Опис екземпляру класу chislo
chislo.x=1;  // Присвоєння полю x значення 1
chislo.y=1;  // Присвоєння полю y значення 1
// Вивід комплексного числа
chislo.show_complex();
// Вивід модуля комплексного числа
cout<<"Модуль комплексного числа
="<<chislo.modul()<<endl;
// Вивід аргументу комплексного числа
cout<<"Аргумент комплексного числа =
"<<chislo.argument()<<endl;
}
```

Використання відкритих членів та методів дозволяє отримати повний доступ до елементів класу, але при беспосередньому зверненні до них виникає потенційна небезпека внести помилки у функціонування взаємозвязаних між собою елементів класу. При цьому обмеження доступу не є способом захисту даних, а тільки методом полегшення програмування та пошуку логічних помилок в роботі програми. Загальноприйнятим елементи - дані є закритими (**private**) , а елементи-функції – відкритими (**public**). Для доступу до закритих члені класу необхідно створити відповідні відкриті елементи-функції, бо доступ до них за допомогою оператора «.» в цьому випадку є неможливим. Наприклад:

```
class complex
{
private:
double x;
double y;
public:
double modul();
double argument();
void show_complex();
void vvid();
};
//обчислення модуля комплексного числа
double complex::modul()
{
double temp;
temp=pow(x*x+y*y,1/2.0);
return temp;
};
//обчислення аргумента комплексного числа
double complex::argument()
{
double temp;
temp=atan2(y,x)*180/pi;
return temp;
};
//вивід комплекснеого числа на екран
void complex::show_complex()
{
if(y>=0) cout<<x<<"+"<<y<<"i"<<endl;
else cout<<x<<"-"<<y<<"i"<<endl;
};
//функція вводу комплексного числа
void complex::vvid()
{
cout<<"Введіть x\t";
cin>>x;
cout<<"Введіть y\t";
cin>>y;
};
```

Фрагмент головної програми для вводу комплексного числа із клавіатури та виводу його на екран матиме наступний вигляд:

```
// Опис комплексного числа
complex chislo;
// Присвоєння значення комплексному числу
chislo.vvid();
// Вивід комплексного числа
chislo.show_complex();
```

Приклади виконання завдання

Приклад №1

Програма для роботи точкою в декартовій системі координат. Членами класу є просторові координати точки **(x,y,z)** (**private**). Методами класу (**public**) є: ввід координат точки із клавіатури; вивід координат точки на екран; обчислення та вивід відстані до початку координат ρ; обчислення та вивід азимутального кута φ (у сферичній системі координат); обчислення та вивід полярного кута θ (у сферичній системі координат). При розробці методів класу використовуються наступні формули:

$$\rho = \sqrt{x^2 + y^2 + z^2}\,; \quad \varphi = arctg\left(\frac{y}{x}\right); \quad \theta = arctg\left(\frac{\sqrt{x^2 + y^2}}{z}\right)$$

```
#include<iostream>
#include<math.h>
#include <windows.h>
#define pi 3.1415926
using namespace std;
class tochka
{
private:
double x;
double y;
double z;
public:
double modul();//Обчислення ρ
double fi();    //Обчислення φ
double teta();  //Обчислення θ
```

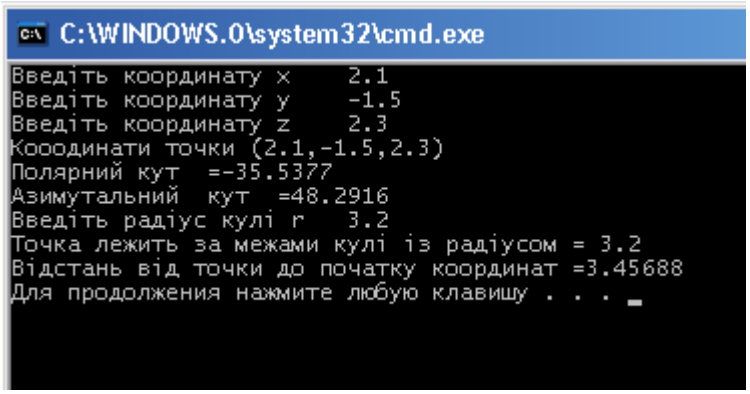
```
    void sfera(double); /*Визначення положення точки
    відносно кулі */
        void show_tochka();//Вивід координат точки
        void vvid();//Ввід координат точки
        };
        double tochka::modul()
        {
        double temp;
        temp=pow(x*x+y*y+z*z,1/2.0);
        return temp;
        };
        double tochka::fi()
        {
        double temp;
        temp=atan2(y,x)*180/pi;
        return temp;
        };
        double tochka::teta()
        {
        double temp;
        temp=atan2(sqrt(x*x+y*y),z)*180/pi;
        return temp;
        };
        void tochka::sfera(double r)
        {
if (pow(x*x+y*y+z*z,1/2.0)<r) cout<<"Точка лежить в
межах кулі із радіусом = "<<r<<endl;

else if (pow(x*x+y*y+z*z,1/2.0)==r) cout<<"Точка
лежить на сфері із радіусом = "<<r<<endl;

else cout<<"Точка лежить за межами кулі із радіусом =
"<<r<<endl;
        };
        void tochka::show_tochka()
        {
    cout<<"Koooдинати точки ("<<x<<","
<<y<<","<<z<<")"<<endl;
        };
        void tochka::vvid()
        {
        cout<<"Введіть координату x\t";
        cin>>x;
```

```
        cout<<"Введіть координату y\t";
        cin>>y;
        cout<<"Введіть координату z\t";
        cin>>z;
        };
        void main()
        {
        SetConsoleOutputCP(1251);
        SetConsoleCP(1251);
        tochka M;
        M.vvid();
        M.show_tochka();
        cout<<"Полярний кут  ="<<M.fi()<<endl;
        cout<<"Азимутальний  кут  ="<<M.teta()<<endl;
        double r;
        cout<<"Введіть радіус кулі r\t";
        cin>>r;
        M.sfera(r);
    cout<<"Відстань від точки до початку координат
    ="<<M.modul() <<endl;
        }
```

Результат роботи програми



Завдання для виконання

Написати програму розв'язку задачі. Розробити контрольні приклади та протестувати за їх допомогою програму.

1. Створити клас дробів. Членами класу є чисельник та знаменник. Методами класу є: ввід дробу з клавіатури; вивід дробу на екран; об-

числення та вивід значення дробу. Написати програму, що демонструє роботу з класом.

2. Створити клас векторів. Членами класу є декартові координати початку та кінця вектора в просторі. Методами класу є: ввід вектора з клавіатури; вивід вектора на екран; обчислення та вивід довжини вектора. Написати програму, що демонструє роботу з класом.
3. Створити клас матриць розміру 3×3. Членами класу є елементи матриці. Методами класу є: ввід матриці з клавіатури; вивід матриці на екран; обчислення та вивід визначника матриці. Написати програму, що демонструє роботу з класом.
4. Створити клас відрізків на площині. Членами класу є координати кінців відрізка. Методами класу є: ввід відрізка з клавіатури; вивід відрізка на екран; обчислення та вивід на екран довжини відрізка. Написати програму, що демонструє роботу з класом.
5. Створити клас прямих на площині. Членами класу є коефіцієнти рівняння прямої $ax+by+c=0$. Методами класу є: ввід прямої з клавіатури; вивід прямої на екран; обчислення та вивід на екран координат точок перетину із осями. Написати програму, що демонструє роботу з класом.
6. Створити клас для роботи із часом в межах доби. Членами класу є години, хвилини та секунди. Методами класу є: ввід часу з клавіатури; вивід часу на екран; обчислення та вивід на екран часу, що залишився до завершення доби. Написати програму, що демонструє роботу з класом.
7. Створити клас комплексних чисел, заданих в показниковій формі $\rho\cdot e^{i\varphi}$. Членами класу є модуль ρ та аргумент φ комплексного числа. Методами класу є: ввід числа з клавіатури; вивід числа на екран; перевід числа у алгебраїчну форму та вивід результату на екран. Написати програму, що демонструє роботу з класом.
8. Створити клас для роботи із колом. Членами класу є радіус кола та координати його центру. Методами класу є: ввід кола із клавіатури; вивід кола на екран; обчислення площі круга та довжини кола і вивід результату на екран. Написати програму, що демонструє роботу з класом.
9. Створити клас прямокутників. Членами класу є довжини сторін прямокутника. Методами класу є: ввід прямокутника із клавіатури; вивід прямокутника на екран; обчислення периметра та його площі і вивід результату на екран. Написати програму, що демонструє роботу з класом.

10. Створити клас трикутників. Членами класу є довжини сторін трикутника. Методами класу є: ввід трикутника із клавіатури; вивід трикутника на екран; обчислення периметра та площі і вивід результату на екран. Написати програму, що демонструє роботу з класом.
11. Створити клас прямокутників. Членами класу є координати вершин на площині. Методами класу є: ввід координат вершин із клавіатури; вивід прямокутника на екран; обчислення периметра та площі і вивід результату на екран. Написати програму, що демонструє роботу з класом.
12. Створити клас трикутників. Членами класу є координати вершин в просторі. Методами класу є: ввід координат вершин із клавіатури; вивід трикутника на екран; обчислення периметра та площі і вивід результату на екран. Написати програму, що демонструє роботу з класом.
13. Створити клас матриць розміру 3×3. Членами класу є елементи матриці. Методами класу є: ввід матриці з клавіатури; вивід матриці на екран; обчислення та вивід оберненої матриці. Написати програму, що демонструє роботу з класом.
14. Створити клас комплексних чисел. Членами класу є дійсна та уявна частина комплексного числа. Методами класу є: ввід комплексного числа з клавіатури; вивід комплексного числа на екран; обчислення та вивід комплексно - спряженого числа. Написати програму, що демонструє роботу з класом.
15. Створити клас для роботи із датою. Членами класу є рік, місяць та день місяця. Методами класу є: ввід дати з клавіатури; вивід дати на екран; обчислення та вивід на екран пори року, що відповідає даній даті. Написати програму, що демонструє роботу з класом.
16. Створити клас матриць розміру 4×4. Членами класу є її елементи. Методами класу є: ввід матриці; вивід матриці на екран; обчислення та вивід на екран максимального елементу матриці. Написати програму, що демонструє роботу з класом.
17. Створити клас для роботи із кільцем. Членами класу є внутрішній та зовнішній радіуси кільця. Методами класу є: ввід кільця із клавіатури; вивід кільця на екран; обчислення площі та товщини кільця і вивід результатів на екран. Написати програму, що демонструє роботу з класом.
18. Створити клас чисел, заданих своїми розрядами. Членами класу є кількість одиниць, десятків, сотень та тисяч. Методами класу є: ввід

числа із клавіатури; вивід числа на екран; обчислення числа та вивід результату на екран. Написати програму, що демонструє роботу з класом.

19. Створити клас векторів. Членами класу є полярні координати вектора в трьохмірному просторі. Методами класу є: ввід вектора з клавіатури; вивід вектора на екран; обчислення та вивід декартових координат вектора. Написати програму, що демонструє роботу з класом.
20. Створити клас точок. Членами класу є координати точки на площині. Методами класу є: ввід точки з клавіатури; вивід координат точки на екран; обчислення та вивід номера квадранта системи координат, в якому точка знаходиться. Написати програму, що демонструє роботу з класом.
21. Створити клас трикутників. Членами класу є координати вершин на площині. Методами класу є: ввід координат вершин трикутника із клавіатури; вивід трикутника на екран; обчислення радіусу кола, периметр якого рівний периметру трикутник і вивід результату на екран. Написати програму, що демонструє роботу з класом.
22. Створити клас точок в просторі. Членами класу є координати точки в циліндричній системі координат. Методами класу є: ввід координат точки з клавіатури; вивід координат точки на екран; обчислення та вивід відстані від точки до початку системи координат. Написати програму, що демонструє роботу з класом.
23. Створити клас матриць розміру 5×5. Членами класу є елементи матриці. Методами класу є: ввід матриці з клавіатури; вивід матриці на екран; обчислення та вивід суми елементів головної та допоміжної діагоналей. Написати програму, що демонструє роботу з класом.
24. Створити клас поліномів розмірності 4. Членами класу є коефіцієнти полінома. Методами класу є: ввід коефіцієнтів полінома; вивід полінома на екран; обчислення та вивід значення полінома для заданого значення x. Написати програму, що демонструє роботу з класом.
25. Створити клас тріад чисел. Членами класу є три дійсних числа. Методами класу є: ввід трьох чисел; вивід чисел на екран; обчислення та вивід найбільшого та найменшого числа. Написати програму, що демонструє роботу з класом.
26. Створити клас еліпсів. Членами класу є довжини півосей еліпса. Методами класу є: ввід еліпса із клавіатури; вивід еліпса на екран;

обчислення площі та периметра еліпса і вивід результату на екран. Написати програму, що демонструє роботу з класом.

27. Створити клас точок в просторі. Членами класу є декартові координати точки. Методами класу є: ввід точки з клавіатури; вивід координат точки на екран; обчислення та вивід полярних координат точки. Написати програму, що демонструє роботу з класом.
28. Створити клас п’ятірок чисел. Членами класу є значення п’яти дійсних чисел. Методами класу є: ввід чисел з клавіатури; вивід чисел на екран; обчислення та вивід на екран середнього арифметичного чисел. Написати програму, що демонструє роботу з класом.
29. Створити клас матриць розміру 5×5. Членами класу елементи матриці. Методами класу є: ввід матриці з клавіатури; вивід матриці на екран; обчислення та вивід максимального та мінімального елементів матриці. Написати програму, що демонструє роботу з класом.
30. Створити клас для роботи із колом та точкою. Членами класу є радіус кола, координати його центру та координати деякої точки. Методами класу є: ввід кола та точки із клавіатури; вивід точки та кола на екран; визначення положення точки відносно кола (в колі чи за його межами) і вивід результату на екран. Написати програму, що демонструє роботу з класом.

Лабораторна робота 13

Тема роботи: Програмування класів: конструктор, деструктор, перевантаження операторів

Мета роботи: Формування навиків роботи із класами з використанням перевантажених операцій, конструкторів та деструктора.

Для виконання роботи необхідно знати:

- опис класу;
- доступ до відкритих та закритих членів класу;
- поняття конструктора, його призначення та опис;
- поняття деструктора, його призначення та опис;
- перевантаження операцій та їх опис.

Теоретичні відомості

Конструктор – це функція-метод класу з таким самим ім'ям, як у класу. Він призначений для створення та ініціалізації об'єктів класу та присвоєння початкових значень їх полям. В класі може бути визначено кілька конструкторів, які відрізняються списком параметрів. При створенні об'єктів класу викликатися буде (перевантаження конструктора) конструктор із відповідною кількістю та типом вхідних параметрів. Конструктор може мати вхідні параметри, а може їх і не мати. Конструктор, який не має вхідних параметрів називається конструктором за замовчуванням.

Клас може і не мати конструктора у явному вигляді. Тоді розподіл пам'яті та ініціалізація об'єкту виконуються системою автоматично стандартним шляхом. Стандартна процедура не враховує особливостей класу, тому може бути некоректною і приводити до помилок в роботі програми.

Деструктор – це функція-метод класу, яка відповідає за коректне вивільнення пам'яті при знищенні об'єкту. Деструктор завжди має ім'я класу, перед яким ставиться символ „~". Якщо деструктор у програмі не визначено, то він генерується на етапі компіляції програми автоматично. Локальні об'єкти видаляються деструктором тоді, коли вони виходять за межі області видимості, о глобальні об'єкти – по завершенні роботи функції ***`main()`***. Деструктор, при необхідності, може виконувати і будь-які інші дії – вивід кінцевих значень елементів даних чи текстових повідомлень при налагодженні програми.

Конструктори і деструктори не можуть повертати значень і тому при їх описі відсутній тип результату (див. приклад №1).

В С++ визначено ряд операцій (+, - , * , / і. т.д.) при роботі зі змінними стандартних типів. Перевантаження операторів дозволяє призначити відповідні операції при їх використанні до елементів-членів власного класу. Компілятор відрізняє різні функції за числом і типом параметрів і операндів. Основними перевантаженими операціями є: ***+; -; *; /; %; =; <; >; +=; -=; *=; /=; <; >; &&; ||; ++; --.*** При цьому перевантажена операція є членом класу і може використовуватися тільки у виразах із об'єктами свого класу.

Для перевантаження операції в середині класу необхідно написати відповідні функції-методи класу. Для визначення нової, перевантаженої дії, операції використовується ключове слово ***operator*** згідно наступного синтаксису:

унарна операція:

void operator <символ оператора>() (префіксний оператор)

void operator <символ оператора>(int) (постфіксний оператор)

бінарна операція:

<тип> operator <символ оператора>(<тип1> <імя змінної>)

де:

<тип> **-** тип значення, що повертається операцією;

<символ оператора> **-** символ перевантаженої операції;

<тип1> **-** тип другого операнду (першим операндом є член класу);

<імя змінної> **-** ім'я змінної другого операнду.

Приклади унарних операцій:

префіксна операція збільшення дійсної частини комплексного числа на 1 може мати наступний вигляд:

```
void operator ++();
 .  .  .  .
void complex::operator ++()
{
x++;
 };
```

постфіксна операція зменшення уявної частини комплексного числа на 1 може мати наступний вигляд:

```
void operator ++(int);
 .  .  .  .
void complex::operator ++(int)
{
y--;
 };
```

Приклади бінарних операцій:
операція *додавання* двох комплексних чисел:

```
complex operator +(complex );
.  .  .  .
complex complex::operator +(complex a)
{
complex temp(0.);
temp.x=x+a.x;
temp.y=y+a.y;
return temp;
};
```

Операція додавання двох комплексних чисел з використанням вказівника **`this`** : вказівник на екземпляр об'єкту класу, для якого виконується метод. Він використовується для отримання доступу до самого об'єкту в тілі члена-функції без його додаткового оголошення. Наприклад:

```
complex operator +(complex );
.  .  .  .
complex complex::operator +(complex a)
{
x=x+a.x;
y=y+a.y;
return *this;
};
```

Приклади операцій присвоєння:
операція присвоєння **=**

```
complex operator =(complex );
        .  .  .  .
complex complex::operator =(complex a)
{
x=a.x;
```

```
y=a.y;
return *this;
};
```

***Примітка**: якщо операція присвоєння не перевантажена явно, то по замовчуванню у виразі **x=y** відбувається побайтове копіювання даних об'єкту **y** в дані об'єкту **x**.*
операція присвоєння **-=**

```
complex operator -=(complex );
        .  .  .  .
complex complex::operator -=(complex a)
{
x-=a.x;
y-=a.y;
return *this;
};
```

Приклади логічних операцій:
операція порівняння двох комплексних чисел:

```
bool operator ==(complex);
  .  .  .
bool complex::operator== (complex other)
{
if(m_re == other.m_re && m_im == other.m_im)
{
return true;
}
return false;
};
```

Приклади виконання завдання

Приклад №1

Програма для роботи із комплексними числами. Членами класу є дійсна та уявна частина комплексного числа. Методами класу є: вивід комплексного числа на екран; конструктор для ініціалізації комплексних чисел; деструктор; обчислення та вивід модуля комплексного числа; обчислення та вивід аргументу комплексного числа.

Для ініціалізації комплексного числа створено перевантажений конструктор:

complex(double) ініціалізація комплексного числа виду $a+i\cdot a$ (якщо параметр в дужках не задано, то по замовчуванню ініціалізується комплексне число $1+i$);

complex(double,double) ініціалізація комплексного числа виду $a+i\cdot b$.

Деструктор ***~complex()*** створено також, хоча для нашого класу задач його використання не є обов'язковим.

Для роботи із комплексними числами описано перевантажені операції:

bool operator ==(complex) порівняння двох комплексних чисел;

complex operator +(complex) додавання двох комплексних чисел;

complex operator -(complex) віднімання двох комплексних чисел;

complex operator *(complex) множення двох комплексних чисел;

complex operator /(complex) ділення двох комплексних чисел;

complex operator -=(complex) комплексне спряження;

complex operator =(complex) операція присвоєння для комплексних чисел;

void operator ++() префіксне збільшення на 1 дійсної частини;

void operator ++(int) постфіксне збільшення на 1 уявної частини;

void operator --() префіксне зменшення на 1 дійсної частини;

void operator --(int) постфіксне зменшення на 1 уявної частини.

```
#include<iostream>
#include<math.h>
#include <windows.h>
#define pi 3.1415926
using namespace std;
class complex
{
private:
```

```
    double x;
    double y;
    public:
    void operator ++();
    void operator ++(int);
    void operator --();
    void operator --(int);
    complex operator +(complex );
    complex operator -(complex );
    complex operator *(complex );
    complex operator /(complex );
    complex operator -=(complex );
    complex operator =(complex);
    bool operator ==(complex);
    double modul();
    double argument();
    void show_complex();
    complex(double);
    complex(double, double);
    ~complex();
    };
    // Порівняння двох комплексних чисел
    bool complex::operator== (complex chislo)
    {
    if(x == chislo.x && y == chislo.y)
    {
    return true;
    }
    return false;
    };
    complex complex::operator +(complex a)
//Додавання
    {
    complex temp(0.);
    temp.x=x+a.x;
    temp.y=y+a.y;
    return temp;
    };
    complex complex::operator -(complex a)
//Віднімання
    {
    x=x-a.x;
```

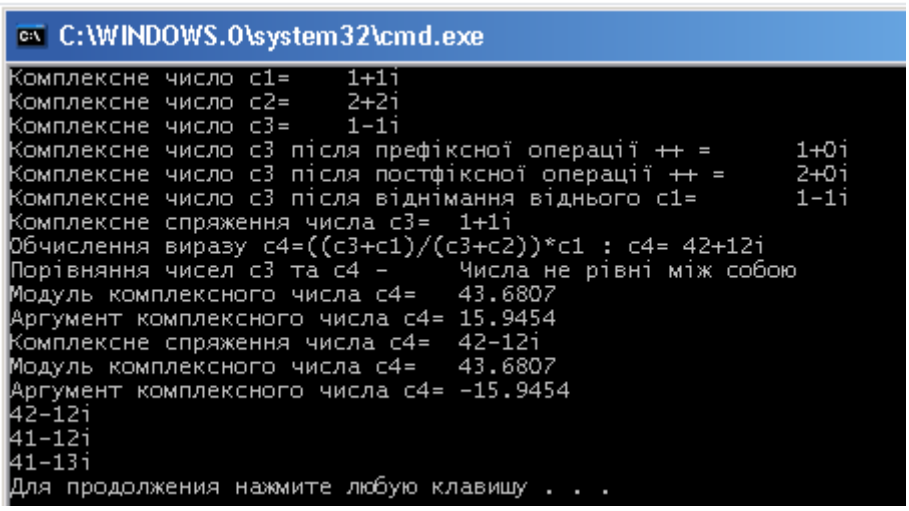
```
    y=y-a.y;
    return *this;
    };
    complex complex::operator *(complex a)
//Множення
    {
    x=x*a.x-y*a.y;
    y=y*a.x+x*a.y;
    return *this;
    };
    complex complex::operator /(complex a)
//Ділення
    {
    double temp;
    temp=a.x*a.x-a.y*a.y;
    x=x*a.x+y*a.y;
    y=y*a.x-x*a.y;
    return *this;
    };
    //Комплексне спряження
    complex complex::operator -=(complex a)
    {
    x=a.x;
    y=-a.y;
    return *this;
    };
    //Операція присвоєння
    complex complex::operator =(complex a)
    {
    x=a.x;
    y=a.y;
    return *this;
    };
    //Префіксне збільшення на 1 дійсної частини
    void complex::operator ++()
    {
    x++;
    };
    //Префіксне зменшення на 1 дійсної частини
    void complex::operator --()
    {
    x--;
    };
```

```
//Постфіксне збільшення на 1 уявної частини
void complex::operator ++(int)
{
y++;
};
//Постфіксне зменшення на 1 уявної частини
void complex::operator --(int)
{
y--;
};
double complex::modul()
{
return pow(x*x+y*y,1/2.0);
};
double complex::argument()
{
return atan2(y,x)*180/pi;
};
void complex::show_complex()
{
if(y>=0) cout<<x<<"+"<<y<<"i"<<endl;
else cout<<x<<"-"<<fabs(y)<<"i"<<endl;
};
//Конструктор №1
complex::complex(double x0=1.0)
{
x=x0;
y=x0;
};
//Конструктор №2
complex::complex(double x0, double y0)
{
x=x0;
y=y0;
};
//Деструктор
complex::~complex()
{
};
void main()
{
SetConsoleOutputCP(1251);
```

```
        SetConsoleCP(1251);
    /*Ввід комплексного числа c1 за допомогою
конструктора №1  */
        complex c1;
        cout<<"Комплексне число c1=\t";
        c1.show_complex();
    /*Ініціалізація комплексного числа c2 за
допомогою конструктора №1 */
        complex c2(2.);
        cout<<"Комплексне число c2=\t";
        c2.show_complex();
    /*Ініціалізація комплексного числа c3 за
допомогою конструктора №2 */
        complex c3(1.,-1.);
        cout<<"Комплексне число c3=\t";
        c3.show_complex();
        c3++;
    cout<<"Комплексне число c3 після префіксної
    операції ++ =\t";
        c3.show_complex();
        ++c3;
    cout<<"Комплексне число c3 після постфіксної
    операції ++ =\t";
        c3.show_complex();
        c3=c3-c1;
    cout<<"Комплексне число c3 після віднімання
віднього c1=\t";
        c3.show_complex();
        c3-=c3;
        cout<<"Комплексне спряження числа c3=\t";
        c3.show_complex();
        complex c4;
        c4=((c3+c1)/(c3+c2))*c1;
        cout<<"Обчислення виразу
    c4=((c3+c1)/(c3+c2))*c1 : c4=\t";
        c4.show_complex();
        cout<<"Порівняння чисел c3 та c4 -\t";
        if (c3==c4) cout<<"Числа рівні \n";
        else cout<<"Числа не рівні між собою\n";
        cout<<"Модуль комплексного числа
    c4=\t"<<c4.modul()<<endl;
        cout<<"Аргумент комплексного числа
    c4=\t"<<c4.argument()<<endl;
```

```
    cout<<"Комплексне спряження числа c4=\t";
    c4-=c4;
    c4.show_complex();
    cout<<"Модуль комплексного числа
c4=\t"<<c4.modul()<<endl;
    cout<<"Аргумент комплексного числа
c4=\t"<<c4.argument()<<endl;
    c4.show_complex();
    --c4;
    c4.show_complex();
    c4--;
    c4.show_complex();
    }
```

Результат роботи програми:



Завдання для виконання

1. Доповнити клас, створений в попередній роботі, перевантаженим конструктором, деструктором та перевантаженими операціями: +; -; *; /; префіксний ++ збільшує чисельник на 1, а постфіксний ++ збільшує знаменник на 1; префіксний -- зменшує чисельник на 1, а постфіксний -- зменшує знаменник на 1; логічною операцією порівняння дробів = =.
2. Доповнити клас, створений в попередній роботі, перевантаженим конструктором, деструктором та перевантаженими операціями: +; -; * (векторний добуток); / (ділення на число); префіксний ++ збільшує координати на 1; постфіксний -- зменшує координати на 1; логічною операцією перевірки колінеарності векторів ||.

3. Доповнити клас, створений в попередній роботі, перевантаженим конструктором, деструктором та перевантаженими операціями: +; -; *= (множення матриці на число); / (ділення матриці на число); префіксний ++ збільшує елементи головної діагоналі на 1, а постфіксний ++ збільшує елементи допоміжної діагоналі на 1; префіксний -- зменшує елементи головної діагоналі на 1, а постфіксний -- зменшує на 1 елементи допоміжної діагоналі; логічними операціями порівняння двох матриць <; = =; > (порівнюються їх визначники).
4. Доповнити клас, створений в попередній роботі, перевантаженим конструктором, деструктором та перевантаженими операціями: + (додавання відрізків згідно правил додавання векторів); - (віднімання відрізків згідно правил віднімання векторів); * (довжина відрізка помножується на число; координат кінця відрізка перераховуються, а координати початку залишаються незмінними); / (довжина відрізка ділиться на число; координат кінця відрізка перераховуються, а координати кінця залишаються незмінними); префіксний ++ збільшує координати початку відрізка на 1, а постфіксний ++ збільшує координати кінця відрізка на 1; префіксний -- зменшує координати початку відрізка на 1, а постфіксний -- зменшує координати кінця відрізка на 1; логічною операцією перевірки паралельності відрізків ||.
5. Доповнити клас, створений в попередній роботі, перевантаженим конструктором, деструктором та перевантаженими операціями: % (знаходження кута між двома прямими); += (паралельний зсув прямої на число); префіксний ++ поворот прямої (зміна кутового коефіцієнта) на 1 градус за годинниковою стрілкою; префіксний -- поворот прямої (зміна кутового коефіцієнта) на 1 градус проти годинникової стрілки; логічною операцією перевірки паралельності прямих ||.
6. Доповнити клас, створений в попередній роботі, перевантаженим конструктором, деструктором та перевантаженими операціями: +; -; префіксний ++ збільшує кількість хвилин на 1, а постфіксний ++ збільшу кількість секунд на 1; префіксний -- зменшує кількість хвилин на 1, а постфіксний -- зменшує кількість секунд на 1; логіч-ними операціями порівняння часів <; = =; >). При всіх операція передбачити перевірку виходу отриманого значення за допустимі межі.
7. Доповнити клас, створений в попередній роботі, перевантаженим конструктором, деструктором та перевантаженими операціями: +; -; *; /; префіксний ++ збільшує дійсну і уявну частину числа на 1; -=

(комплексне спряження числа); префіксний -- зменшує дійсну і уявну частину числа на 1; логічною операцією порівняння двох комплексних чисел = =.

8. Доповнити клас, створений в попередній роботі, перевантаженим конструктором, деструктором та перевантаженими операціями: + (в результаті додавання отримується коло, площа якого рівна сумі площ доданків з координатами центру першого доданку); - (в результаті віднімання отримується коло, площа якого рівна різниці площ доданків з координатами центру першого доданку); * (множення радіусу на число); / (ділення радіусу на число); префіксний ++ збільшує x - координату центру кола на 1, а постфіксний ++ збільшує y - координату центру кола на 1; префіксний -- зменшує x - координату центру кола на 1, а постфіксний -- зменшує y - координату центру кола на 1; логічними операціями порівняння двох кіл <; = =; > (порівнюються їх площі).
9. Доповнити клас, створений в попередній роботі, перевантаженим конструктором, деструктором та перевантаженими операціями: + (в результаті додавання отримується прямокутник, площа якого рівна сумі площ доданків за рахунок пропорційного збільшення довжин сторін першого доданку); - (в результаті віднімання отримується прямокутник, площа якого рівна різниці площ по модулю за рахунок пропорційної зміни довжин сторін першого доданку); префіксний ++ збільшує довжини сторін прямокутника на 1; префіксний -- зменшує довжини сторін прямокутника на 1; логічною операцією порівняння двох прямокутників = = (порівнюються їх площі).
10. Доповнити клас, створений в попередній роботі, перевантаженим конструктором, деструктором та перевантаженими операціями: + (в результаті додавання отримується трикутник, площа якого рівна сумі площ доданків за рахунок пропорційного збільшення довжин сторін першого доданку); - (в результаті віднімання отримується трикутник, площа якого рівна різниці площ по модулю за рахунок пропорційної зміни довжин сторін першого доданку); префіксний ++ збільшує всі сторони трикутника на 1; префіксний -- зменшує всі сторони трикутника на 1; *= (множення всіх довжин сторін трикутника на число); /= (ділення всіх довжин сторін трикутника на число); логічною операцією порівняння двох трикутників = = (порівнюються їх периметри).
11. Доповнити клас, створений в попередній роботі, перевантаженим конструктором, деструктором та перевантаженими операціями: + (в результаті додавання отримується прямокутник, площа якого рівна

сумі площ доданків за рахунок пропорційного збільшення довжин сторін першого доданку при незмінних координатах нижньої лівої точки першого прямокутника); - (в результаті віднімання отримується прямокутник, площа якого рівна різниці площ по модулю за рахунок пропорційної зміни довжин сторін першого доданку при незмінних координатах правої верхньої точки першого прямокутника); префіксний ++ збільшує x- координати лівих точок прямокутника на 1, а постфіксний ++ збільшує $x-$ координати правих точок прямокутника на 1; префіксний -- зменшує $x-$ координати лівих точок прямокутника на 1, а постфіксний -- зменшує x- координати правих точок прямокутника на 1; логічними операціями порівняння двох прямокутників <; = =; > (порівнюються їх площі).

12. Доповнити клас, створений в попередній роботі, перевантаженим конструктором, деструктором та перевантаженими операціями: + (в результаті додавання отримується трикутник, площа якого рівна сумі площ доданків за рахунок пропорційного збільшення довжин сторін першого доданку при незмінній координаті першої точки трикутника); - (в результаті віднімання отримується трикутник, площа якого рівна різниці площ по модулю за рахунок пропорційного зменшення довжин сторін першого доданку при незмінній координаті останньої точки трикутника); префіксний ++ збільшує всі x – координати трикутника на 1, а постфіксний ++ збільшує всі y – координати трикутника на 1; префіксний -- зменшує всі x – координати трикутника на 1, а постфіксний -- зменшує всі y – координати трикутника на 1; логічною операцією порівняння двох трикутників = = (порівнюються довжини їх сторін).
13. Доповнити клас, створений в попередній роботі, перевантаженим конструктором, деструктором та перевантаженими операціями: +; -; /= (ділення елементів головної діагоналі на число); */ (множення елементів допоміжної діагоналі на число); префіксний ++ збільшує всі елементи матриці на 1; префіксний -- зменшує всі елементи матриці на 1; логічною операцією порівняння двох матриць = = (порівнюються елементи матриць).
14. Доповнити клас, створений в попередній роботі, перевантаженим конструктором, деструктором та перевантаженими операціями: +; -; *= (множення дійсної та уявної частину на дане число); /= (ділення дійсної та уявної частини на дане число); префіксний ++ збільшує дійсну частину числа на 1; постфіксний ++ збільшує уявну частину числа на 1; префіксний -- зменшує дійсну частину числа на 1; постфіксний -- зменшує уявну частину числа на 1; логічною операцією порівняння двох комплексних чисел !=.

15. Доповнити клас, створений в попередній роботі, перевантаженим конструктором, деструктором та перевантаженими операціями: + (додавання відповідно років, місяців та днів) ; - (віднімання відповідно років, місяців та днів); префіксний ++ збільшує рік на 1, а постфіксний ++ збільшу місяць на 1; префіксний -- зменшує рік на 1, а постфіксний -- зменшує місяць на 1; += (збільшує день на 1); -= (зменшує день на 1); логічними операціями порівняння часів <; = =; >. При всіх операція передбачити перевірку виходу отриманого значення за допустимі межі.
16. Доповнити клас, створений в попередній роботі, перевантаженим конструктором, деструктором та перевантаженими операціями: +; -; *; /= (ділення елементів матриці на число); += (збільшує всі елементи матриці на 1); -= (зменшує всі елементи матриці на 1); логічною операцією порівняння двох матриць != (порівнюються елементи матриць).
17. Доповнити клас, створений в попередній роботі, перевантаженим конструктором, деструктором та перевантаженими операціями: + (в результаті додавання отримується кільце, площа якого рівна сумі площ доданків за рахунок збільшення зовнішнього радіусу першого доданку); - (в результаті віднімання отримується кільце, площа якого рівна різниці площ по модулю за рахунок зменшення зовнішнього радіусу першого доданку); префіксний ++ збільшує зовнішній радіус кільця на 1, а постфіксний ++ збільшу внутрішній радіус кільця на 1; префіксний -- зменшує зовнішній радіус кільця на 1, а постфіксний -- зменшує внутрішній радіус кільця на 1; логічною операцією порівняння = = (порівнюються внутрішній та зовнішній радіуси). При всіх операція передбачити перевірку виходу отриманого значення за допустимі межі.
18. Доповнити клас, створений в попередній роботі, перевантаженим конструктором, деструктором та перевантаженими операціями: + (додаються відповідні розряди); - (віднімаються відповідні розряди); префіксний ++ збільшує всі розряди числа на 1; префіксний -- зменшує всі розряди числа на 1; логічними операціями порівняння двох чисел, заданих своїми розрядами <; = =; >). При всіх операція передбачити перевірку виходу отриманого значення за допустимі межі.
19. Доповнити клас, створений в попередній роботі, перевантаженим конструктором, деструктором та перевантаженими операціями: +; -; += (збільш полярний кут на 1 градус); -= (зменшує полярний кут на 1 градус); префіксний ++ збільшує довжину вектора на 1;

постфіксний -- зменшує довжину вектора на 1; логічною операцією перевірки неколінеарності векторів !=.

20. Доповнити клас, створений в попередній роботі, перевантаженим конструктором, деструктором та перевантаженими операціями: + (додаються відповідні координати двох точок); - (віднімаються відповідні координати двох точок); префіксний ++ збільшує x – координату точки на 1, а постфіксний ++ збільшує y – координату точки на 1; префіксний -- зменшує x – координату точки на 1, а постфіксний -- зменшує y – координату точки на 1; логічною операцією порівняння двох точок = = (порівнюються відповідні кординати точок).
21. Доповнити клас, створений в попередній роботі, перевантаженим конструктором, деструктором та перевантаженими операціями: += (в результаті додавання отримується трикутник, площа якого рівна сумі площ доданків за рахунок пропорційного збільшення довжин сторін другого доданку при незмінній першій вершині трикутника); -= (в результаті віднімання отримується трикутник, площа якого рівна різниці площ по модулю за рахунок пропорційного зменшення довжин сторін першого трикутника при незмінних координатах першої вершини); префіксний ++ збільшує всі координати три кутника на 1; префіксний -- зменшує всі координати трикутника на 1; логічною операцією порівняння двох трикутників = = (порівнюються площі трикутників).
22. Доповнити клас, створений в попередній роботі, перевантаженим конструктором, деструктором та перевантаженими операціями: + (додаються z координати точок); - (віднімаються z координати точок); += (до радіальної відстані ρ додається число); -= (від радіальної відстані ρ віднімається число); префіксний ++ збільшує азимутальний кут на 1°; префіксний -- зменшує азимутальний кут на 1°; логічною операцією порівняння двох точок != (порівнюються відповідні циліндричні координати точок).
23. Доповнити клас, створений в попередній роботі, перевантаженим конструктором, деструктором та перевантаженими операціями: +; / (почленне ділення елементів однієї матриці на елементи другої матриці (при діленні на нуль результуючий елемент рівний 0)); */ (множення елементів матриці на число); префіксний ++ збільшує елементи лівого стовпця матриці на 1; постфіксний ++ збільшує елементи правого стовпця матриці на 1; префіксний -- зменшує елементи верхнього рядка матриці на 1; постфіксний -- зменшує елементи нижнього рядка матриці на 1; логічними операціями порів-

няння двох матриць <; = =; > (порівнюються суми елементів головної та допоміжної діагоналі).

24. Доповнити клас, створений в попередній роботі, перевантаженим конструктором, деструктором та перевантаженими операціями: + (почленне додавання поліномів); - (почленне віднімання поліномів); */ (множення всіх коефіцієнтів полінома на число); префіксний ++ збільшує всі коефіцієнти полінома на 1; префіксний -- зменшує всі коефіцієнти полінома на 1; логічною операцією порівняння двох поліномів = = (порівнюються відповідні коефіцієнти полінома).

25. Доповнити клас, створений в попередній роботі, перевантаженим конструктором, деструктором та перевантаженими операціями: + (почленне додавання тріад чисел); - (почленне віднімання тріад чисел); += (циклічний зсув чисел за годинниковою стрілкою на одну позицію); -= (циклічний зсув чисел проти годинникової стрілки на одну позицію); префіксний ++ збільшує перше число на 2, а постфіксний ++ збільшує останнє число на 2; префіксний -- зменшує перше число на 1, а постфіксний -- зменшує останнє число на 1; логічною операцією порівняння трійок чисел = = .

26. Доповнити клас, створений в попередній роботі, перевантаженим конструктором, деструктором та перевантаженими операціями: + (додаються відповідні півосі еліпсів); - (віднімаються відповідні півосі еліпсів (по модулю)); * (множення півосей еліпса на число); / (ділення півосей еліпса на число); префіксний ++ збільшує півосі на 1; префіксний -- зменшує півосі еліпса на 1; логічними операціями порівняння еліпсів <; = =; > (порівнюються площі еліпсів).

27. Доповнити клас, створений в попередній роботі, перевантаженим конструктором, деструктором та перевантаженими операціями: + (додаються координати точок); - (віднімаються координати точок); += (до координат додається число); -= (від координат віднімається число); префіксний ++ збільшує всі координати точки на 2; префіксний -- зменшує всі координати точки на 2; логічними операціями порівняння двох точок <; = = ; > (порівнюються відстані від точок до початку координат).

28. Доповнити клас, створений в попередній роботі, перевантаженим конструктором, деструктором та перевантаженими операціями: + (почленне додавання п’ятірок чисел); - (почленне віднімання пятірок чисел); += (додає до всіх чисел дане число); -= (віднімає від всіх чисел дане число); префіксний ++ виконує циклічний зсув чисел за годинниковою стрілкою на одну позицію; префіксний – виконує циклічний зсув чисел проти годинникової стрілки на одну позицію;

логічною операцією порівняння п’ятірок чисел != (порівнюються відповідні числа).

29. Доповнити клас, створений в попередній роботі, перевантаженим конструктором, деструктором та перевантаженими операціями: +; * (множення елементів матриці на число); префіксний ++ збільшує елементи головної діагоналі на 2, а постфіксний ++ збільшує елементи допоміжної діагоналі матриці на 2; префіксний -- зменшує елементи головної діагоналі матриці на 2, а постфіксний -- зменшує елементи допоміжної діагоналі матриці на 2; логічними операціями порівняння двох матриць <; = =; > (порівнюються визначники матриць).
30. Доповнити клас, створений в попередній роботі, перевантаженим конструктором, деструктором та перевантаженими операціями: + (радіусу кола збільшується на дане число); - (радіуса кола зменшується на дане число); += (до координат точки додається число); -= (від координати точки віднімається число); префіксний ++ збільшує всі координати центру кола на 1; префіксний -- зменшує всі координати центру кола на 1; логічними операціями порівняння об’єктів <; = =; > (порівнюються відстані від точки до центру кола).

Завдання для самостійного виконання

1. Напасати програму, яка за введеними днем народження (формат вводу: рік, місяць, день місяця) обчислює, скільки вам років, скільки місяців, днів та годин ви прожили. Результат виводиться на екран та у файл.
2. Обчислити $y=(ax-b)^{\sqrt{a}}+(ax-b)^{\sqrt{b}}+|a+b|$, $g=\sin(y+\pi/4)$ при $x=2.32$. На екран та у файл вивести g,y,a,b. Значення a та b задати самостійно як константи.
3. Дано два файли. Один містить тривалість телефонних розмов та номер країни, в яку був здійснений дзвінок. Другий – вартість однієї хвилини дзвінка в ту чи іншу країну. Обчислити сумарну вартість всіх виконаних дзвінків.
4. Із клавіатури вводиться порядковий номер дня тижня. Вивести розклад пар у цей день (врахувавши, чи це чисельник чи знаменник). Розклад пар зчитується із файлу.
5. Обчислити $y=\left(4x^2-2x-a^2\right)^{b+4}+\sqrt{b}|\operatorname{tg} ax|$, $t=y+4.34$ при $x=1.7$. На екран та у файл вивести x,y,t,a,b. Значення a та b задати самостійно як константи.
6. В файлі записано 100 довільних цілих чисел (в діапазоні від 0 до 100). Обчислити кількість чисел, що більші A, але менші B (вводяться із клавіатури), а результат вивести на екран та дописати в даний файл.
7. Трикутник задано координатами свої вершин на площині. Визначити, де лежить точка, задана своїми координатами, відносно трикутника (в середині; на трикутнику; за межами трикутника).
8. Коло задано координатами центру та радіусом. Визначити кількість точок перетину кола із прямою виду $ax+by+c=0$.
9. Дано коефіцієнти трьох прямих виду $ax+by+c=0$. Знайти, чи є серед прямих паралельні, чи співпадаючі прямі та вивести відповідне повідомлення.
10. В файлі записано 1000 випадкових чисел (в діапазоні від 0 до 100). У новостворений файл вивести всі прості числа та їх порядкові номери.
11. Обчислити $y=\sqrt{\sin\frac{|x|}{\sqrt{a-x}}-b\cos\frac{|x|}{\sqrt{a-x}}}-\frac{a+b}{2}$, $k=\sqrt[5]{y+1.34}$ при $x=1.7$. На екран та у файл вивести x,y,k. Значення a та b задати самостійно як константи.

12. Дано коефіцієнти двох прямих виду $ax+by+c=0$. Знайти точку перетину прямих, а якщо вони не перетинаються, то вивести відповідне повідомлення.

13. У файлі задано послідовність із 25 цілих чисел. Вивести у новостворений файл всі парні числі, а на екран – їх кількість. Дописати у вихідний файл кількість від'ємних та додатних чисел.

14. Файл містить матрицю F розміром 10×10 елементів. Сформувати із нього новий файл, в якому парні та непарні стовпці поміняні місцями.

15. Дано одномірний масив, що містить 100 дійсних чисел (у вигляді окремого файлу). Обчислити добуток чисел, що знаходяться на місцях, номери яких кратні 3, та не рівні нулю. Результат дописати у вихідний файл.

16. Дано одномірний масив 50 результатів вимірювань (у вигляді файлу). Визначити та вивести у інший файл значення та номери тих вимірювань, які відрізняються від середнього значення не більше ніж на 10%.

17. У файлі у вигляді одномірного масиву записано результати 25 вимірів. Знайти та вивести на екран значення та порядковий номер того виміру, який найближчий до середнього арифметичного результатів вимірювань. Результат вивести на екран та дописати у файл.

18. Знайти добуток елементів двомірного масиву розміром 7×7, які знаходяться між максимальним та мінімальним елементами.

19. Обчислити кількість елементів двомірного масиву розміру 15×10, для яких виконується нерівність $i*j<a_{ij}$. Елементи масиву зчитати із попередньо підготовленого файлу.

20. Написати програму, яка визначає, чи є в даному двомірному масиві (записаному у файлі) цілих чисел хоча б два однакові рядки (або стовпці).

21. Дано два цілочислені масиви A та B розміром 10×10 елементів у вигляді двох файлів. Підрахувати та вивести на екран (із використанням форматного виводу) значення та індекси тих елементів, для яких виконується рівність $A_{i,j}=B_{j,i}$.

22. Обчислити максимальний та мінімальний елементи серед елементів, що знаходяться вище головної діагоналі двомірного масиву. Результат вивести на екран, а у файл вивести масив, в якому мінімальний та максимальний елементи поміняно місцями.

23. Написати програму, яка визначає номер того рядка двомірної матриці, в якому середнє арифметичне елементів максимально близьке до середнього арифметичного всіх елементів матриці. Матриця зчитується із попередньо підготовленого файлу.

24. Написати програму, яка визначає максимальні елементи в кожному із рядків, а серед них знаходить найменший елемент та виводить його значення (та порядковий номер) на екран у зручній для сприйняття формі.

25. Обчислити суму елементів двомірного масиву, які знаходяться на головній діагоналі. Замінити всі елементи вище головної діагоналі отриманим значенням. Матриця зчитується із попередньо підготовленого файлу, а перетворена матриця виводиться у новий файл.

26. У двомірному масиві визначити, в якому рядку чи стовпці сума елементів приймає максимальне значення. Матриця зчитується із попередньо підготовленого файлу.

27. Написати програму, яка переводить десяткове число в двійкову систему числення. Отримані результати записати у файл з використанням форматного виводу у вигляді таблиці в два стовпці із пояснювальною інформацією.

28. У файлі записана виграшна комбінація цифр (5 чисел). Із клавіатури вводиться своя комбінація чисел, а програма визначали, чи Ви вгадали, чи ні та виводить відповідне повідомлення.

29. Написати програму, що обчислює значення суми $1^4+2^4+3^4+4^4+...+N^4$ та перевіряє правильність отриманого результату, який рівний $\frac{(N^2+N)(2N+1)(3N^2+3N-1)}{30}$ і виводить відповідне повідомлення. Результат покрокового сумування вивести у окремий файл.

30. Написати програму, що обчислює значення суми $1^2+3^2+5^2+...+(2N-1)^2$ та перевіряє правильність отриманого результату, який рівний $\frac{N(4N^2-1)}{3}$ і виводить відповідне повідомлення. Значення N читається із файлу, куди також дописується результат сумування.

31. Написати програму, що ідентифікує трапецію за двома прилеглими кутами (звичайна, прямокутна, рівнобедрена, прямокутна, прямо кутник).

32. Написати програму, що ідентифікує паралелепіпед за прилеглими сторонами та кутом між ними (прямокутник, ромб, квадрат, звичайний паралелепіпед).

33. Написати програму, яка обчислює суму нескінченого функціонального ряду $1-\frac{x^2}{3!}+\frac{x^4}{5!}-...+(-1)^N\frac{x^{2N}}{(2N+1)!}+..$ із точністю 10^{-6}. Результат покрокового сумування записати у окремий файл, куди також вивести кількість членів ряду, яка була обчислена для забезпечення заданої точності.

34. Написати програму, яка обчислює суму нескінченого функціонального ряду $x+\frac{x^3}{3!}+\frac{x^5}{5!}+...+\frac{x^{2N+1}}{(2N+1)!}+..$ із точністю 10^{-4}. Результат обчислення записати у окремий файл, куди також вивести кількість членів ряду, яка була обчислена для забезпечення заданої точності.

35. Написати програму, яка обчислює суму нескінченого функціонального ряду $2(\frac{\sin x}{1}-\frac{\sin 2x}{2}+\frac{\sin 3x}{3}-...)$ із точністю 10^{-4}. Результат покрокового сумування записати у окремий файл, куди також вивести кількість членів ряду, яка була обчислена для забезпечення заданої точності. Значення x читається із файлу (в межах $-\pi<x<\pi$).

36. Написати програму, яка обчислює суму нескінченого функціонального ряду $\sum_{i=1}^{\infty}\frac{\sin(ix)}{i}$ із точністю 10^{-6}. Результат обчислення записати у окремий файл, куди також вивести кількість членів ряду, яка була обчислена для забезпечення заданої точності. Значення x ввести із клавіатури (в межах $-\pi<x<\pi$).

37. Написати програму, яка обчислює суму нескінченого функціонального ряду $\sum_{i=1}^{\infty}(-1)^{i+1}\frac{\cos(ix)}{i^2}$ із точністю 10^{-6}. Результат покрокового сумування записати у окремий файл, куди також вивести кількість членів ряду, яка булла обчислена для забезпечення заданої точності. Значення x ввести із клавіатури (в межах $-\pi<x<\pi$).

38. Написати програму, яка обчислює суму нескінченого ряду $1+\frac{1}{2^4}+\frac{1}{3^4}+...$ із точністю 10^{-6}. Результат обчислення записати у окремий файл. Отриманий результат порівняти із точним значенням $\frac{\pi^4}{90}$. Обидва результати вивести в файл із пояснювальною текстовою ін формацією.

39. Написати програму, яка обчислює суму нескінченого ряду $1+\frac{1}{3^2}+\frac{1}{5^2}+...$ із точністю 10^{-5}. Результат обчислення записати у окремий файл.

Отриманий результат порівняти із точним значенням $\frac{\pi^2}{8}$. Обидва результати вивести в файл із пояснювальною текстовою інформацією.

40. Дано масив 100 довільних чисел, який зчитується із файлу. Знайти кількість сусідніх трійок чисел, сума яких не перевищує середнього арифметичного всього масиву чисел. Вивести на екран відповідне повідомлення.

41. Значення функції $y=\exp(4x)$ можна обчислити її розкладом в ряд Маклорена $\exp(4x)=\sum\limits_{n=0}^{\infty}\frac{(4x)^n}{n!}=1+4x+\frac{(4x)^2}{2!}+\ldots+\frac{(4x)^n}{n!}+\ldots$ Обчислити значення функції з точністю ε (обчислення продовжується до моменту, поки абсолютне значення чергового члену ряду не буде меншим ε). Підрахувати кількість членів ряду, які необхідні просумувати для досягнення заданої точності. Результати сумування та кінцевий результат виводити у файл.

42. Написати програму табулювання функції $y=x^x\sin(2x)$ на проміжку $[a,b]$ із кроком h. Результат табулювання вивести у файл у зручному для сприйняття форматі. Визначити максимальне та мінімальне значення функції на проміжку табуляції та вивести їх на екран.

43. Значення функції $y=\ln(1+5x)$ можна обчислити її розкладом в ряд Маклорена $\ln(1+5x)=\sum\limits_{n=1}^{\infty}\frac{(-1)^{n+1}(5x)^n}{n}$ Обчислити значення функції з точністю ε (обчислення продовжується до моменту, поки абсолютне значення чергового члену ряду не буде меншим ε). Підрахувати кількість членів ряду, які необхідні просумувати для досягнення заданої точності. Результати сумування та кінцевий результат виводити у файл.

44. Значення функції $y=\operatorname{arcctg}(4x)$ можна обчислити її розкладом в ряд Маклорена $\operatorname{arcctg}(4x)=\frac{\pi}{2}+\sum\limits_{n=0}^{\infty}\frac{(-1)^{n+1}(4x)^{2n+1}}{2n+1}$ Обчислити значення функції з точністю ε (обчислення продовжується до моменту, поки абсолютне значення чергового члену ряду не буде меншим ε). Підрахувати кількість членів ряду, які необхідні просумувати для досягнення заданої точності. Результати сумування та кінцевий результат виводити у файл.

45. Написати програму табулювання функції $y=x\cdot 2^x+5\cdot tg(x)$ на проміжку $[a,b]$ із кроком h. Результат табулювання вивести у файл у зручному для сприйняття форматі. Визначити максимальне та мінімальне значення функції на проміжку табуляції та вивести їх у файл.

46. Знайти корінь рівняння $2.345 \cdot x^4 + 3.098 - e^{0.556 \cdot x} = 0$. Метод вибрати самостійно.

47. Знайти корінь рівняння $(x + 0.3) \cdot (e^{-0.4 \cdot x} - 1.456) + 2.37 = 0$. Метод вибрати самостійно.

48. Обчислити суму ряду (для $x < 0$) із точністю 0.0001

$$1 - 3x + \frac{3 \cdot 4}{2!} x^2 - \frac{3 \cdot 4 \cdot 5}{3!} x^3 + \ldots$$

Порівняти отриманий результат із точним значенням $((1+x)^{-2}$. Суму та кількість просумованих членів вивести у файл.

49. Обчислити суму ряду (для x<1) із точністю 0.00005

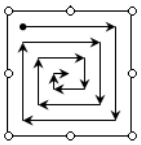
$$1 - 4x + \frac{4 \cdot 5}{2!} x^2 - \frac{4 \cdot 5 \cdot 6}{3!} x^3 + \ldots$$

Порівняти отриманий результат із точним значенням $((1+x)^{-4}$.Суму та кількість просумованих членів вивести у файл.

50. На площині дано N кіл, заданих координатами своїх центрів та радіусами (у вигляді окремого файлу). Визначити та вивести на екран координати тих кіл, які не мають перетинів із іншими колами.

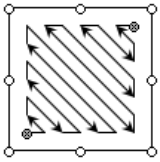
51. Із заданого шестизначного числа побудувати нове, сусідні цифри в якому поміняні місцями (перша з другою, третя з четвертою, п'ята з шостою).

52. Заповнити квадратну матрицю 11×11 цілими числами від 1 до 121 згідно приведеного нижче алгоритму



53. На площині задано M кіл (координати центрів та їх радіуси) та T точок (координати) у двох окремих файлах. Вивести номер та кординати того кола, яке містить максимальну кількість точок.

54. Заповнити квадратну матрицю 15×15 цілими числами від 1 до 225 згідно приведеного нижче алгоритму



55. На площині координатами вершин задано R прямокутників зі сторонами, паралельними осям координат (у вигляді окремого файлу). Визначити та вивести номер та координати прямокутників, що мають максимальну площу та максимальний периметр.

56. Написати програму, яка обчислює добуток $\prod_{N=1}^{\infty}(1-\frac{4x^2}{(2N-1)^2\pi^2})$ із точністю 10^{-6}. Результат обчислення записати у окремий файл у вигляді таблиці при зміні x від a до b з кроком Δx (читаються із файлу). Результати виводити в файл із пояснювальною текстовою інформацією.

57. У файлі записано масив 10×10 із нулів та одиниць. Вивести його у файл, здійснивши дзеркальне відображення відносно головної діагоналі.

58. На площині задано координати трьох точок. Визначити координати центра кола, що проходить через ці точки, якщо відомо його радіус. Результати записати у файл.

59. Дано текст, сформований у вигляді окремого файлу. Підрахувати кількість слів у тексті, які мають букви, що повторюються у слові не менше двох раз.

60. Дано текст, сформований у вигляді окремого файлу. Підрахувати, скільки разів у тексті зустрічається слова, які починаються буквою, введеною із клавіатури.

61. Дано текст, сформований у вигляді окремого файлу. Вивести на екран найдовше та найкоротше слово, яке починаються буквою, введеною із клавіатури.

62. Із трьох заданих чисел вибрати та вивести на екран те, сума цифр якого є максимальною.

63. Від початку експерименту пройшло N секунд (>10000). Визначити та вивести на екран кількість діб, годин, хвилин та секунд, що пройшли від початку експерименту. Результат вивести у файл.

64. Трикутник задано координатами його вершин на площині (у вигляді окремого файлу). Визначити його тип (гострокутний, прямокутний, тупокутний, рівносторонній, рівнобедрений).

65. Обчислити значення інтегралу $\int_0^{\pi/2}\sin^6(x)dx$ методом прямокутників та порівняти із його точним значенням ($2\cdot\pi/32$).Результати обчислення виводити на екран та у файл.

66. Обчислити значення інтегралу $\int_0^1 \frac{dx}{1+x+x^2}$ методом трапецій та порівняти із його точним значенням ($\pi/(3\sqrt{3})$).Результати обчислення виводити на екран та у файл.

67. Обчислити значення інтегралу $\int_0^{\pi/2} \frac{1}{1+3\cos(x)} dx$ методом трапецій та порівняти із його точним значенням ($\pi/4$). Результати обчислення виводити на екран та у файл.

68. Дано ціле шестизначне число. Перевірити, чи сума його трьох перших цифр рівна сумі трьох останніх та вивести на екран відповідне повідомлення. Число читається із файлу.

69. Для деякого введеного із клавіатури цілого тризначного числа k знайти значення n, для якого $|k-n!|$ приймає мінімальне значення.

70. Дано два файли. Один містить дані про об'єми рідин, другий – про їх густини. Написати програму, яка у третій файл виводить маси рідин, а на екран їх сумарну масу з використанням форматного виводу та пояснювальної текстової інформації.

71. Дано результати 20 експериментальних вимірювань у вигляді текстового файлу. Написати програму, яка відкидає 2 найкращі та 2 найгірші виміри, а для 16 інших обчислює середнє арифметичне значення, абсолютну та відносну похибку. Результат дописує у вихідний файл з використанням форматного виводу та пояснювальної текстової інформації.

72. Із клавіатури вводиться деяке десяткове ціле число. Написати програму, яка визначає: введене число додатне чи від'ємне; розрядність; переводить його в двійкову систему числення. Результати вивести у файл.

73. Написати програму, яка ціле число в двійковій системі числення переводить в будь-яку іншу систему числення, задану із клавіатури.

74. Написати програму, яка використовує функцію для знаходження мінімальної відстані між прямою та точкою, введеними із клавіатури. Знайдена відстань та введені вхідні дані записуються у файл.

75. Написати програму, яка використовує функцію для знаходження мінімальної відстані між двома паралельними прямими. Всі вхідні дані зчитуються із файлу. Знайдена відстань дописується у вихідний файл.

76. Користуючись розкладом функції *f(x)* в ряд *S(x)*, обчислити її значення. Підрахувати кількість членів сумування, які забезпечують точність 0.00001.

$$S(x)=\sum_{k=0}^{\infty}(-1)^k\frac{x^{2k+1}}{2k+1} \qquad |x|<1 \qquad f(x)=arctg(x)$$

77. Прямокутні області на площині задані координатами своїх вершин, записаних у файлі. Написати програму, яка за введеними із клавіатури координатами точки виводить у інший файл координати тих областей, яким дана точка належить.

78. Один файл містить результати екзамену із інформатики. Інший – прізвища студентів. Вивести у один файл імена всіх відмінників, у другий – всіх двійочників. На екран вивести середній бал групи за екзамен з використанням форматного виводу та пояснювальної текстової інформації.

79. Написати програму, яка шифрує файл із текстом за допомогою кодової таблиці. Результат записується у інший файл.

80. У файлі міститься інформація про виграшну комбінацію із шести цифр. Написати програму, яка пропонує ввести із клавіатури шість цифр, та виводить повідомлення про кількість цифр, які було відгадано (порядок слідування цифр має значення).

81. У файлі записано 50 різноманітних цифрових комбінацій (5 цифр). Написати програму, яка за введеною виграшною комбінацією визначає кількість виграшних комбінацій у файлі та виводить їх порядкові номери у окремий файл. Порядок слідування цифр значення немає.

82. Із клавіатури вводиться послідовність цілих чисел (число 0 означає кінець введення послідовності). Найти мінімальне та максимальне число послідовності.

83. Із клавіатури вводиться послідовність N цілих чисел. Сформувати та вивести у файл нову послідовність, в якій число замінене сумою своїх цифр.

84. В квадратній матриці 20×20 знайти максимальний елемент, що знаходиться на периметрі (крайніх рядках та стовпцях) матриці та мінімальний елемент поза периметром матриці.

85. Дано матриця 20×20, елементами якої є або нулі, або одиничи. Написати програму, яка визначає номер стовпця, що містить максимальну кількість нулів, та номер рядка, що містить максимальну

кількість одиничок. Матрицю сформувати у вигляді окремого файлу.

86. Створити масив структур, що мітить наступну інформацію автобусні маршрути: номер рейсу; пункт призначення; час відправлення; час у дорозі; загальна кількість місць у автобусі (оформлена у вигляді окремого файлу). Зчитати дані із файлу та вивести маршрути, впорядкувавши їх за часом відправлення. Якщо час відправлення один і той самий, то додатково впорядкувати їх за кількістю місць у автобусі.

87. Іде n-та секунда експерименту (вводиться із клавіатури). Визначити, кільки повних днів, годин та хвилин, що пройшли від початку експерименту. Вхідні та вихідні дані вивести у файл.

88. Дано матриця 25×20, елементами якої є або нулі, або одиничи. Написати програму, яка заміняє нулі на одиничи в тому випадку, якщо по сусідству знаходиться не менше трьох одиничок. Матрицю сформувати у вигляді окремого файлу.

89. Дано дійсне число x. Обчислити значення виразу $\frac{(x-2)(x-4)(x-8)...(x-128)}{(x-1)(x-3)(x-5)...(x-127)}$. Число x читається із файлу, а значення виразу дописується в нього-ж.

90. Дано двомірну матрицю 10×10 елементів (у вигляді окремого файлу). Якщо сума елементів над головною діагоналлю рівна сумі елементів під нею, то замінити всі діагональні елементи їх середнім геометричним та вивести її на екран.

91. Із клавіатури вводиться послідовність із 20 чисел. Якщо наступне введене число відрізняється від попереднього більше ніж на 25%, то програма повідомляє про некоректний ввід та пропонує повторити спробу. Числа вивести у окремий файл у рядок. У наступному рядку вивести значення їх добутку та кількість некоректно введених чисел із відповідною пояснювальною текстовою інформацією.

92. Обчислити добуток мінімального та максимального значень функції на заданому проміжку. Кількість значень аргументу рівна 19. Аргумент міняється від початкового значення 0.1 з кроком 0.05 радіан.

$$f(x)=\frac{1+c}{(b-x)\sin^3(x)}\ ,\ c=3.452,\ \ b=1.673.$$

93. Написати програму, яка читає із файлу довільну кількість декартових координат (x,y) точки на площині та обчислює їх полярні координати ρ òà φ. Результат виводиться на екран та у файл окремий файл.

94. Для функції $f(x)=\dfrac{a-\sqrt{|b-x|}}{\ln^2(a+3)}$, $a=1.352$, $b=1.105$ аргумент міняється від початкового значення 1.1 з кроком 0.2 до кінцевого значення 4.1 (читаються із файлу). У один файл вивести тільки від'ємні значення функції та їх кількість, у інший – тільки додатні значення функції та їх кількість.

95. Із клавіатури вводиться рядок слів, розділених довільною кількістю пробілів. Видалити всі лишні пробіли (більше одного) та циклічно зсунути слова на три позиції. Результат вивести к файл. Підрахувати також кількість видалених пробілів.

96. Перший елемент геометричної прогресії рівний 2, а знаменник – 1.5. Наступний член прогресії утворюється множенням попереднього на знаменник прогресії. Знайти добуток членів геометричної прогресії, які більші a та менші b (їх значення читаються із файлу). Значення добутку дописати у вихідний файл.

97. Дано двомірний масив $C_{10\times12}$ чисел, що лежать в межах від 1 до 100 (у вигляді окремого файлу). Знайти найбільший елемент масиву, який ділиться на 3 без остачі.

98. Написати програму для роботи із векторами. Членами класу є декартові координати початку та кінця вектора на площині. Методами класу є: ввід вектора з клавіатури; вивід вектора на екран; обчислення та вивід довжини вектора. Він повинен також містити перевантажений конструктор, деструктор, операції додавання та віднімання векторів, скалярний добуток, множення вектора на число та перевірку ортогональності векторів.

99. Два трикутники на площині задано координатами їх вершин. Визначити та вивести у окремий файл відношення площ та відношення периметрів даних трикутників.

100. Із клавіатури вводиться довільний текст. Вивести цей текст у файл, виділивши символ, заданий із клавіатури по обидва боки символами !.

101. Дано три дроби виду $\dfrac{a_1}{b_1},\dfrac{a_2}{b_2},\dfrac{a_3}{b_3}$ (зчитуються із файлу). Визначити та вивести на екран найбільший та найменший дріб.

102. Із клавіатури вводиться послідовність із N цілих чисел. Вивести у файл нові числа, які отримаються в результаті запису цифр чисел у зворотному порядку.

103. У файлі задано координати N точок на площині. У інший файл

вивести кут між віссю абсцис та променем, що з'єднує точку із початком координат. На екран вивести номер та координати точки, для якої цей кут приймає максимальне значення.

104. У файлі записано координати трьох точок на площині. Визначити, чи лежать ці точки на одній прямій та вивести на екран відповідне повідомлення.

105. Дано координати точок x_1, y_1; x_2, y_2; x_3, y_3; x_4, y_4. Визначити і вивести на екран та у файл токи, для яких кут між віссю абсцис і променем, що з'єднує точку з початком координат, є максимальним та мінімальним. Вивести також значення кута.

106. Функція $y(x)$ задана таблицею:

x_i	x_1	x_2	x_3	x_4	...	x_{20}
y_i	y_1	y_2	y_3	y_4	...	y_{20}

Обчислити значення цієї функції в довільній точці $x_1 \le x \le x_{20}$ згідно формули лінійної інтерполяції: $y(x) = y_i + \dfrac{x - x_i}{x_{i+1} - x_i}\left(y_{i+1} - y_i\right)$, де $x_i \le x \le x_{i+1}$.

107. Одновимірний масив A складається з 30 елементів (у вигляді окремого файлу). Знайти та вивести числа, які зустрічаються в масиві більше одного разу. Підрахувати також, скільки разів вони зустрічаються.

108. Дано текст, в якому міститься два символи "+". Поміняти місцями послідовність до першого символу, із послідовністю після другого символу. Вихідний текст вводиться із клавіатури, а результуючий виводиться у файл.

109. Створити клас із для роботи зі списком студентів. Полями класу є прізвище, номер групи та оцінки за три семестрові екзамени. Дані класу читаються із файлу. На екран виводяться двійочники із групи, заданої із клавіатури.

110. Створити клас для роботи із трикутниками. Членами класу є координати вершин трикутника на площині. Методами класу є перевірка існування трикутника, обчислення та вивід відомостей про нього: довжина сторін, кути, площа, периметр. Передбачити логічні операції порівняння трикутників за їх площами та перевірку, чиє трикутник рівностороннім.

Додаток №1. Основні заголовочні файли C++

Заголовочний файл	Короткий опис
ctype.h	Перетворення та обробка символів
math.h	Математична бібліотека
stdio.h	Стандартна бібліотека вводу-виводу
stdlib.h	Функції форматного перетворення даних (рядка в число і навпаки)
string.h	Функції для роботи із рядковими змінними
windows.h	Файл для підтримки робити Windows-додатків
iostream	Стандартна бібліотека потокового вводу-виводу
fstream	Стандартна бібліотека потокового вводу-виводу при роботі з файлами
conio.h	Функції для організації консольного вводу-виводу при роботі із текстовими даними

Додаток №2. Базові типи даних

Тип даних	Назва	Діапазон значень
char	Цілий довжиною не менше 8 біт	-128 . . 127
unsigned int	Без знаковий цілий	0 . . 65535
short int (short)	Короткий цілий	-32768 . . 32767
unsigned short	Беззнаковий короткий цілий	0 . . 65535
int	Цілий	-32768 . . 32767
unsigned long	Беззнаковий довгий цілий	0 . . 4294967295
long int (long)	Довгий цілий	-214748348 . . 2147483647
float	Дійсний одинарної точності	3.4E-38 . . 3.4E+38
double	Дійсний подвійної точності	1.7E-308 . . 1.7E+308
long double	Дійсний максимальної точності	3.4E-4932 . . 1.1E+4932

Додаток №3. Основні специфікатори формату функції ***printf***

Специфікатор	Формат
%c	Символ
%d	Десяткове ціле число зі знаком
%i	Десяткове ціле число зі знаком
%e	Науковий формат (рядкова буква e)
%E	Науковий формат (прописна буква E)
%f	Десяткове число із плаваючою крапкою
%g	Дійсна величин як f або E в залежності від значення
%s	Рядок символів
%u	Десяткове ціле число без знаку
%%	Знак %

Можна дещо розширити визначення специфікації формату: після `%` можна задати число позицій, що виділяються для даного формату. Наприклад:

%10.5f (для десяткового числа із плаваючою комою)

%10i (для десяткового цілого числа зі знаком).

Додаток №4. Основні операції мови С++

Операція	Короткий опис	Приклад застосування
Бінарні арифметичні операції		
+	Плюс (додавання арифметичних операндів)	**x+y**
-	Мінус (віднімання арифметичних операндів)	**x-y**
*****	Множення	**x*y**
/	Ділення (якщо операнди цілочисельні, абсолютне значення результату заокруглюється до цілого, тобто **20/3** дорівнює **6**)	**x/y**
%	Залишок від ділення целочисельних операндів (Наприклад **17%3=1**)	**x%y**
Унарні арифметичні операції		
+	Унарний плюс (підтвердження знаку)	**+x**
-	Унарний мінус (зміна знаку)	**-x**
++	Інкремент (збільшення на одиницю): префіксна операція (**++x**) збільшує операнд на 1 до його використання; постфіксна операція (**x++**) збільшує операнд на 1 після його використання.	**m++** **++n**
--	Декремент (зменшення на одиницю): префіксна операція (**--x**) зменшує операнд на 1 до його використання; постфіксна операція (**x--**) зменшує операнд на 1 після його використання.	**--x** **y--**

Операції присвоєння

Операція	Короткий опис
x=y	Змінній **x** присвоюється значення змінної **y**
x+=y	Означає **x=x+y**
x-=y	Означає **x=x-y**
x*=y	Означає **x=x*y**
x/=y	Означає **x=x/y**
x%=y	Означає **x=x%y**

Додаток №5. Основні математичні функції бібліотеки *math.h*

№ п/п	Функція бібліотеки	Опис
1.	***abs(x)***	Обчислення модуля числа (тільки для цілих чисел)
2.	***acos(x)***	Обчислення арккосинуса ***x*** (в радіанах)
3.	***asin(x)***	Обчислення арксинус ***x*** (в радіанах)
4.	***atan(x)***	Обчислення арктангенса ***x*** (в радіанах)
5.	***atan2(x/y)***	Обчислення арктангенса відношення ***x*** та ***y*** (в радіанах)
6.	***ceil(x)***	Повертає дійсне значення, що відповідає найменшому цілому числу, більшому або рівному ***x***
7.	***cos(x)***	Обчислення косинуса ***x***, заданого в радіанах
8.	***cosh(x)***	Обчислення гіперболічного косинуса ***x***, заданого в радіанах
9.	***exp(x)***	Повертає результат піднесення числа ***e*** до степені ***x***
10.	***fabs(x)***	Обчислення абсолютного значення дійсного числа ***x***
11.	***floor(x)***	Повертає дійсне значення, що відповідає найбільшому цілому числу, меншому або рівному ***x***
12.	***fmod(x,y)***	Остача від цілочисленого ділення ***x*** на ***y*** дійсного типу
13.	***log(x)***	Обчислення натурального логарифма ***x***
14.	***log10(x)***	Обчислення десяткового логарифма ***x***
15.	***pow(x,y)***	Піднесення ***x*** до степені ***y***
16.	***sin(x)***	Обчислення синуса ***x***, заданого в радіанах
17.	***sinh(x)***	Обчислення синуса гіперболічного ***x***, заданого в радіанах
18.	***sqrt(x)***	Обчислення квадратного кореня ***x***
19.	***tan(x)***	Обчислення тангенса ***x***, заданого в радіанах
20.	***tanh(x)***	Обчислення тангенса гіперболічного ***x***, заданого в радіанах

Додаток №6.

Операції порівняння

Операція	Значення
`<`	Менше
`<=`	Менше або рівно
`==`	Рівно
`>=`	Більше або рівно
`>`	Більше
`!=`	Не рівно

Логічні операції

Операція	Значення
`&&`	Логічне І (AND)
`||`	Логічне АБО (OR)
`!`	Логічне заперечення (NOT)

Додаток №7. Можливі специфікатори аргументу *mode*

Значення	Короткий опис
`r`	Відкриття файлу тільки для читання.
`w`	Відкриття файлу тільки для запису.
`a`	Відкриття файлу тільки для додавання інформації в кінець файлу. Якщо файл не існує, то він буде створений.
`r+`	Відкриття вже існуючого файлу для читання та запису.
`w+`	Створення нового файлу для читання та запису.
`a+`	Відкриття файлу у режимі читання та запису для додавання нової інформації у кінець файлу. Якщо файл не існує, то він буде створений.

Додаток №8. Основні функції для роботи із рядковими змінними бібліотеки ***string.h***

Функція	Короткий опис	Пояснення
strlen(ch)	повертає дожину рядку ***ch***	Повертає фактичну довжину рядка, не враховуючи нуль-символ
strcmp(ch1,ch2)	Порівнює посимвольно рядки ***ch1*** та ***ch2***	Якщо ***ch1***<***ch2***, то результат від'ємний, якщо ***ch1***=***ch2***, то результат рівний 0, якщо ***ch1>ch2***– результат додатний.
strncmp(ch1,ch2,n)	Порівнює перші ***n*** символів рядків ***ch1*** та ***ch2***	Для перших ***n*** символів: Якщо ***ch1***<***ch2***, то результат від'ємний, якщо ***ch1***=***ch2***, то результат рівний 0, якщо ***ch1>ch2***– результат додатний.
strcpy(ch1,ch2)	Копіює символи рядка ***ch2*** у рядок ***ch1***	Нуль-символ при цьому теж включається
strncpy(ch1,ch2,n)	Копіює ***n*** символів рядка ***ch2*** у рядок ***ch1***	Кінець рядка відкидається. Якщо нуль-символ у вихідному рядку зустрінеться раніше, копіювання припиняється, а решта символів рядка доповнюються '\0'-ми.
strcat(ch1,ch2)	Дописує рядок ***ch2*** до рядка ***ch1***	Перший символ ***ch2*** записується на місце нуль-символу рядка ***ch1***. До результуючого ***ch1***додається '\0'.

strcatn(ch1,ch2,n)	Дописує перші ***n*** символів рядка ***ch2*** до рядок ***ch1***	***n*** символів рядка ***ch2*** записується до s1, починаючи з місця нуль-символу ***ch1***.
strchr(ch,s)	Шукає символ ***s*** у рядку ***ch***	Повертає вказівник на перше входження символу в рядок справа. Якщо його немає – повертається NULL
strrev(ch)	Змінює порядок символів у рядку ***ch*** на протилежний	Дзеркальне відображення рядка ***ch***
strstr(ch1,ch2)	Шукає підрядок ***ch2*** у рядку ***ch1***	Пошук першого входження ***ch2*** у ***ch1***. В разі вдалого пошуку повертається вказівник на елемент з ***ch1***, з якого починається ***ch2***, інакше – NULL.
strtok(ch1,ch2)	Розбиває рядок на лексеми	Функція повертає вказівник на лексему в ***ch1***, відокремлену символом з набору ***ch2*** (пробілами або розділовими знаками).
atoi(ch)	Перетворює рядкову змінну ***ch*** у цифровий формат цілого типу	Повертає числове значення цілого типу без перевірки коректності перетворення
atof(ch)	Перетворює рядкову змінну ***ch*** до типу ***double***	Повертає числове значення дійсного типу подвійної точності без перевірки коректності перетворення

Додаток №9. Основні функції для роботи із символами бібліотеки ***ctype.h***

Функція	Короткий опис	Пояснення
isalpha(ch)	Перевіряє чи є символ ***ch*** буквою або цифрою (A-Z, a-z, 0-9).	Повертається ***true***, якщо ***ch*** є буквою або цифрою, інакше false
isspace(ch)	Перевіряє чи є символ ***ch*** пропуском (пробіл, табуляція, символ нового рядка, нової сторінки).	Повертається ***true***, якщо ***ch*** є узагальненим пробілом, інакше false
isdigit(ch)	Перевіряє чи ***ch*** цифрою (0-9).	Повертається ***true***, якщо ***ch*** є цифрою, інакше false
islower(ch)	Перевіряє чи ***ch*** є буквою нижнього регістру.	Повертається ***true***, якщо ***ch*** є буквою нижнього регістру, інакше false
isupper(ch)	Перевіряє чи ***ch*** є буквою верхнього регістру.	Повертається ***true***, якщо ***ch*** є буквою верхнього регістру, інакше false
ispunct(ch)	Перевіряє чи ***ch*** є символом пунктуації (. , : ; ? ! тощо).	Повертається ***true***, якщо ***ch*** є символом пунктуації, інакше false

ЛІТЕРАТУРА

1. Шилдт Г. Полный справочник по C++ / Герберт Шилдт ; [пер. с англ. – 4-е изд.].– М. : Вильямс, 2006. – 800 с.
2. Страуструп Б. Язык программирования C++. Специальное издание / Бьерн Страуструп. – С. ПБ. : Бином, Невский Диалект, 2008. – 1104 с.
3. Шилдт Г. Самоучитель C++ / Герберт Шилдт. – С. Пб. : BXB, 2005. – 688 с.
4. Хэзфилд Р. Искусство программирования на C : Фундаментальные алгоритмы, структуры данных и примеры приложений / Р. Хэзфилд, Л. Кирби. – К. : ДиаСофт, 2001. – 736 с. – (Серия: Энциклопедия программиста).
5. Прата С. Язык программирования C++ : лекции и упражнения / Стивен Прата. – К. : ДиаСофт ЮП, 2005. – 1104 с.
6. Глинський Я. М. C і C++ : навч. посіб. / Глинський Я. М., Анохін В. Ї., Ряжська В. А. – Л. : Деол, СПД Глинський, 2006. – 192 с.
7. Буч Г. Обьектно-ориентированный анализ и проектирование с примерами приложений на C++ / Гради Буч. – С. Пб : Бином, Невский Диалект, 2001. – 388 с.
8. Павловская Т. А. C/C++ : Программирование на языке высокого уровня / Татьяна Павловская. – С. Пб. : Питер, 2003. – 461 с. – (Учебник для ВУЗов).
9. Павловская Т. А. C/C++ : Объектно-ориентированное программирование : Практикум / Т. Павловская, Ю. Щупак. – С. Пб. : Питер, 2006. – 265 с.
10. Пауэрс Л. Microsoft Visual Studio 2008 / Л. Пауэрс, М. Снелл ; [пер. с англ.]. – С. Пб. : БХВ-Петербург, 2009. – 1200 с.
11. Єфіменко С. В. Програмування : мови C і C++ / С. В. Єфіменко, О. В. Сугакова. – К. : Київський університет, 2006.
12. Эллис М. Справочное руководство по языку программирования C++ с комментариями / М. Эллис, Б. Страуструп. – М. : Мир, 1992. – 445 с.
13. Дьюхарст С. Программирование на C++ / С. Дьюхарст, К. Старк. – К. : ДиаСофт, 1993. – 272 с.
14. Вирт Н. Алгоритмы и структуры данных / Н. Вирт. – С. Пб. : Невский диалект, 2005. – 406 с.
15. Дэвис С. Р. C++ для «чайников» / Стефан Р. Дэвис. – М. : Вильямс, 2003. – 336 с.
16. Николенко Д. В. Самоучитель по Visual C++ / Дмитрий Николенко. – С. Пб. : Наука и техника, 2001. – 368 с.

17. Шмидский Я. К. Программирование на языке С/С++ : Самоучитель / Яков Шмидский. – М. : Вильямс, 2004. – 368 с.
18. Голуб А. И. Правила программирования на Си и Си++ / Ален Голуб. – М. : БИНОМ, 2001. – 242 с.
19. Лаптев В. С++ : Экспресс-курс / Валерий Лаптев. – С. Пб. : БХВ-Петербург, 2004. – 512 с.
20. Круглински Д. Программирование на Microsoft Visual С++ 6.0 для профессионалов / Д. Круглински, С. Уингоу, Дж. Шеферд. – С. Пб. : Питер ; М. : Русская редакция, 2001. – 864 с.
21. Липпман С. Б. Язык программирования С++ : Вводный курс / С. Б. Липпман, Ж. Лажойе. – С. Пб. : Невский проспект ; М. : ДМК Пресс, 2001. – 1104 с.
22. Нейбауэр А. Моя первая программа на С/С++ / А. Нейбауэр. – С. Пб. : Питер, 1996. – 364 с.
23. Сэджвик Р. Фундаментальные алгоритмы на С++ : анализ, структуры данных, сортировка, поиск / Р. Сэджвик. – К. : ДиаСофт, 2001. – 688 с.
24. Хенкеманс Д. Программирование на С++ / Д. Хенкеманс, М. Ли. – С. Пб. : Символ-Плюс, 2002. – 416 с.
25. Шилдт Г. Теория и практика С++ / Герберт Шилдт. – С. Пб. : БХВ-Петербург, 2000. – 416 с.

ЗМІСТ

Навчальне видання

ТКАЧУК Валерій Михайлович

ПРОГРАМУВАННЯ НА С++
ЛАБОРАТОРНИЙ ПРАКТИКУМ

В авторській редакції

Головний редактор *В. М. Головчак*
Комп'ютерна верстка *В. Д. Яремко*

Підп. до друку 21.12.2011. Формат 60x84/16. Папір офсет.
Гарнітура "Times New Roman". Ум. друк. арк. 9,3.
Тираж 100 пр. Зам. № .

Видавець і виготовлювач
Видавництво Прикарпатського національного університету
імені Василя Стефаника
76000, м. Івано-Франківськ, вул. С. Бандери, 1
Тел. 71-56-22. E-mail: vdvcit@pu.if.ua
Свідоцтво суб'єкта видавничої справи ДК № 2718 від 12.12.2006