Лабораторна робота
з дисципліни «Алгоритмізація та програмування» № 1

Тема: Середовища програмування. Введення/виведення даних

Мета роботи: Освоїти порядок виконання основних дій, пов'язаних із роботою в середовищі програмування (MS Visual Code, Microsoft Visual Studio, X Code) та розробленням С++ -програм у консольному режимі, введенням та виведенням даних.

Теоретичні відомості

Середовища програмування

Microsoft Visual Studio

У Microsoft Visual Studio кожне окреме застосування називається рішенням (solution), що складається з одного чи декількох проектів (project). Проект є набором таких файлів: вихідних .cpp, заголовних .h, ресурсів .rc, відлагодження, налаштувань. Для створення нового застосування слід створити новий проект (файл .vcproj). Якщо проект створено, можна відкрити файл рішення (.sin). З відкриттям рішення стають доступними всі файли, що складають проект.

Для створення нового проєкта при запуску на початковій сторінці слід обрати Create a new project – Empty Project– Next. У наступному вікні потрібно вибрати шаблон. Якщо середовище запускається не перший раз, то в лівій частині вікна у списку Recent project templates, будуть відображатись останні проекти.

Далі необхідно вибрати Location - місце зберігання проекту, Project name – ім'я проекту та Solution name – ім'я рішення (рис. 1). Далі потрібно натиснути кнопку Create.

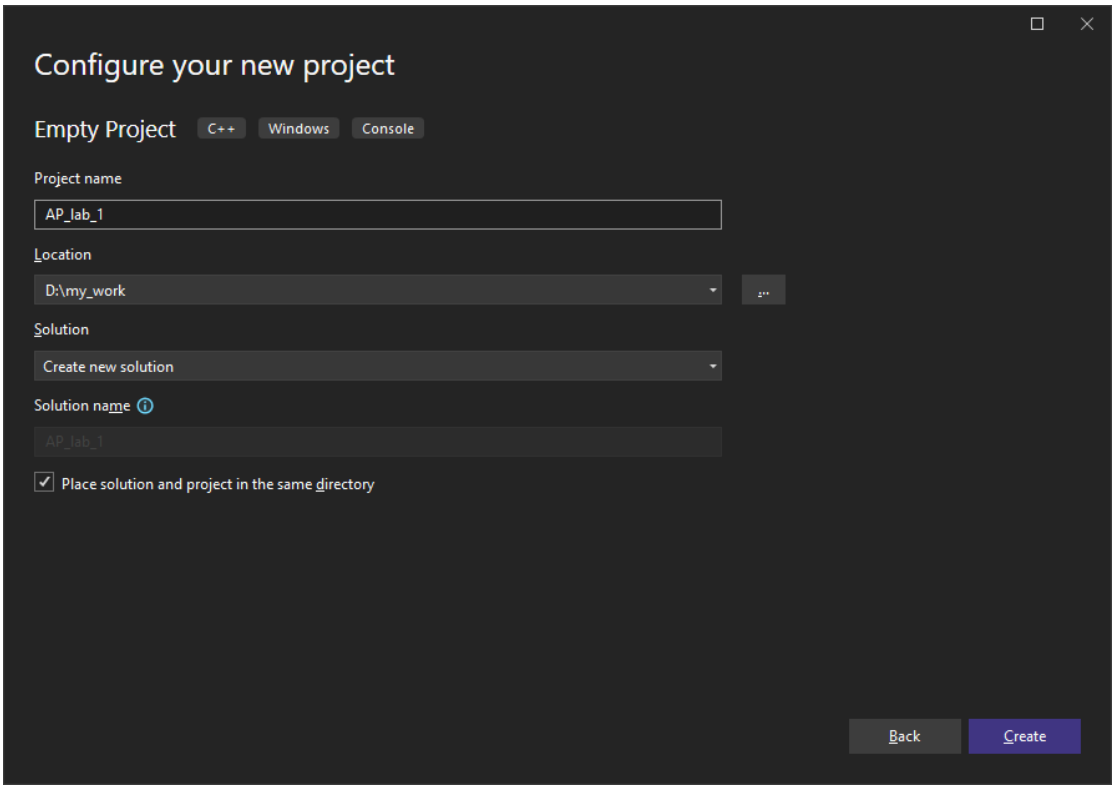

Рис. 1. Створення проєкту у Microsoft Visual Studio

Після цього на екрані з’явиться вікно проекту. Для продовження роботи у проект слід додати файл початкового коду. Для цього у вікні Solution Explorer (Оглядача рішень) проекту, для пункту Source Files (Файли початкового коду) за допомогою миші потрібно викликати контекстне меню, в якому обрати пункт Add (Додати), а потім New Item (Новий елемент).

На екрані відобразиться вікно майстра Add New Item (Додавання нового елемента), подане на рис. 2. Зі списку встановлених шаблонів потрібно вибрати шаблон C++ File (.cpp) (Файл C++ (.cpp)), у поле Name (Назва) ввести назву файла, наприклад, main і натиснути кнопку Add (Додати).

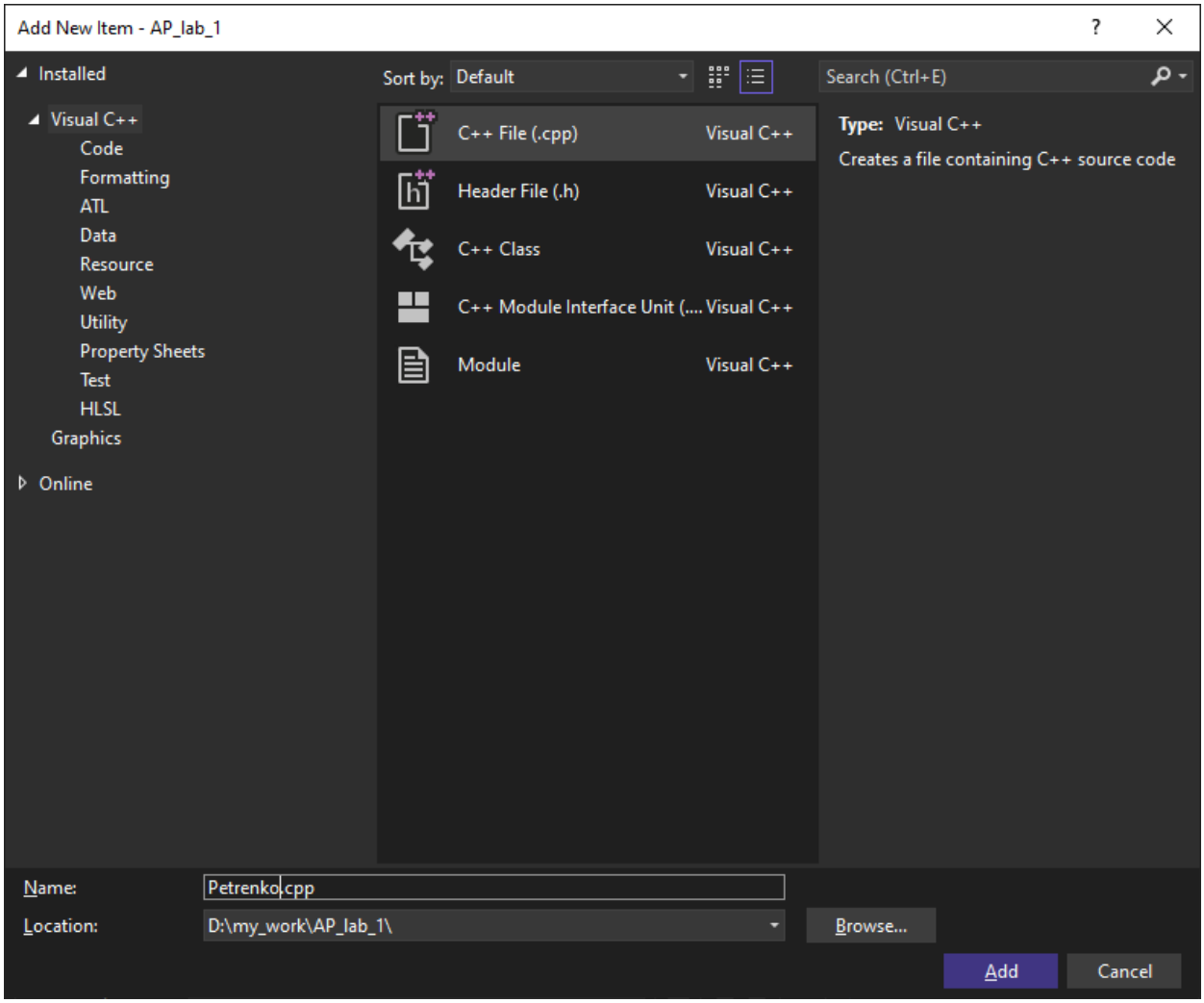

Рис. 2. Вікно додавання нового елемента

Щоб продовжити редагування програми, яка була раніше збережена у файл, слід відкрити проект чи рішення командою File → Open → Project/Solution... або натисканням комбінації клавіш Ctrl+Shift+O. У діалоговому вікні Open Project потрібно вказати шлях до файлів проекту (.vcproj) чи рішення (.sin) і натиснути кнопку Open.

Відкритий раніше в Microsoft Visual Studio проект автоматично закривається. Замість нього у вікні Solution Explorer відкриється вибране користувачем рішення. Вікно з текстом обраного файла з’являться в робочій зоні Microsoft Visual Studio, якщо активізувати cpp-файл з категорії Sourse Files вікна Solution Explorer.

Під час набору тексту програми різні синтаксичні елементи її коду виділяються різними кольорами. За замовчуванням ключові слова зображуються синім кольором, коментарі – зеленим, основний текст коду — чорним. Різнокольорова схема допомагає уникнути випадкових синтаксичних помилок на етапі редагування коду.

Редактор коду підтримує його зображення у вигляді блочної структури. Включити цей режим можна командою меню Edit → Outlining → Collapse to Definitions. Вузлами блочної структури є такі елементи: фігурні дужки, що виділяють тіло функції; блок операторів і виразів; блоки коментарів. Розгорнутий блок позначається символом '–', що стоїть зліва від нього. Його можна згорнути, натиснувши на символ '–'. Згорнутий блок позначається символом '+'. Тіло такого блоку позначається символами {…}.

Завершення роботи у Microsoft Visual Studio здійснюється командою File → Exit або закриттям вікна середовища. При цьому користувачеві буде запропоновано зберегти всі файли, до яких було внесено зміни після останнього збереження.

Visual Studio Code

Підтримка C/C++ для Visual Studio Code забезпечується розширенням Microsoft C/C++ , щоб уможливити кросплатформну розробку на C і C++ у Windows, Linux і macOS. Коли створюється *.cpp файл, розширення додає такі функції, як підсвічування синтаксису (розфарбовування), розумне завершення та наведення (IntelliSense), а також перевірку помилок.

Для роботи з даними засобами C++ у VS Code потрібно спочатку встановити саме середовище а потім встановити спеціальне розширення так компілятор.

Для встановлення розширення потрібно обрати піктограму перегляду розширень на панелі активності, обрати розширення, наведене на рис. 3 та встановити його.

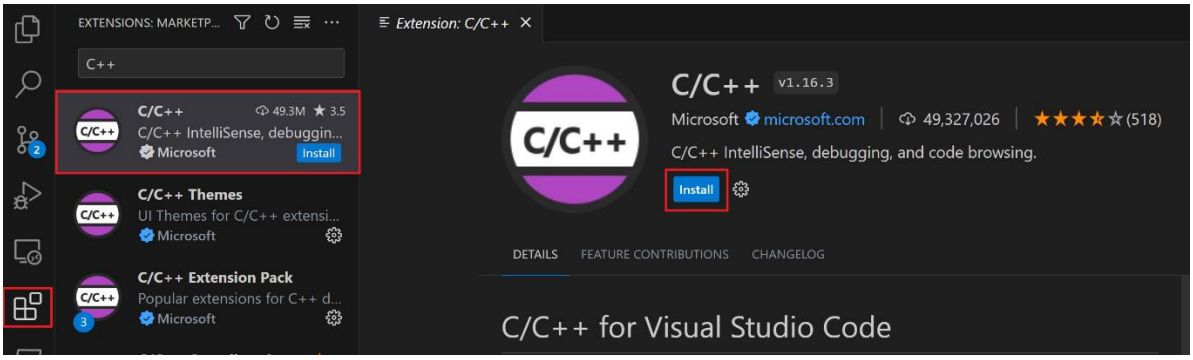

Рис. 3. Встановлення розширення

Встановлене розширення C/C++ не включає компілятор або відладчик C++, оскільки VS Code як редактор покладається на інструменти командного рядка для робочого процесу розробки. Ці інструменти потрібно встановити додатково.

Для створення нового проєкту потрібно відкрити заздалегідь створений робочий каталог File – Open folder – обрати відповідний каталог. Далі слід додати до каталогу новий файл з розширенням .cpp (рис.4). Після відкривання створеного файлу в редакторі з правого боку буде розташована робоча область для роботи з кодом.

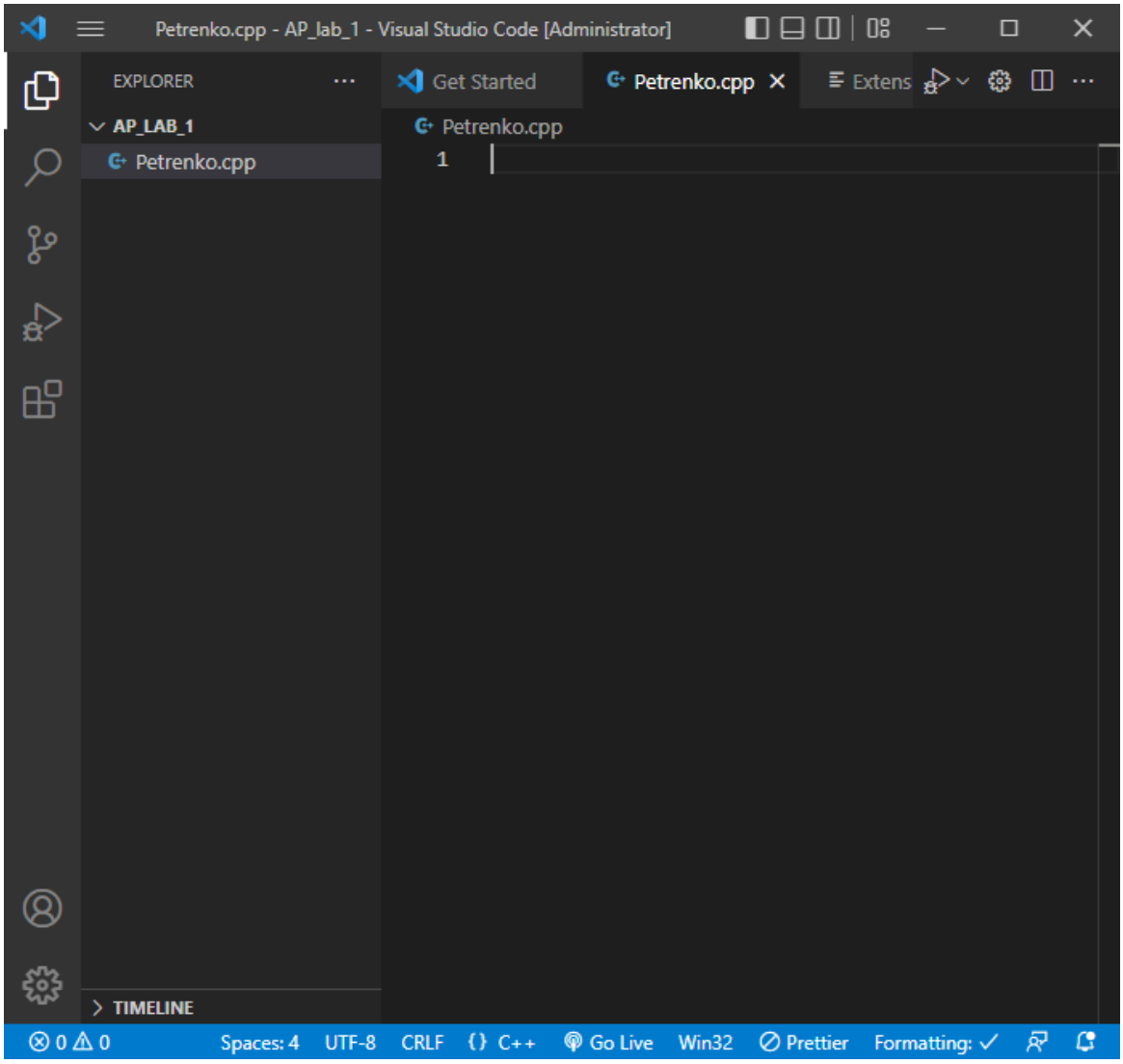

Рис. 4. Створення cpp файлу у VS Code

Для запуску проєкту на виконання потрібно виконати команду “Run C/C++ File” (рис. 5).

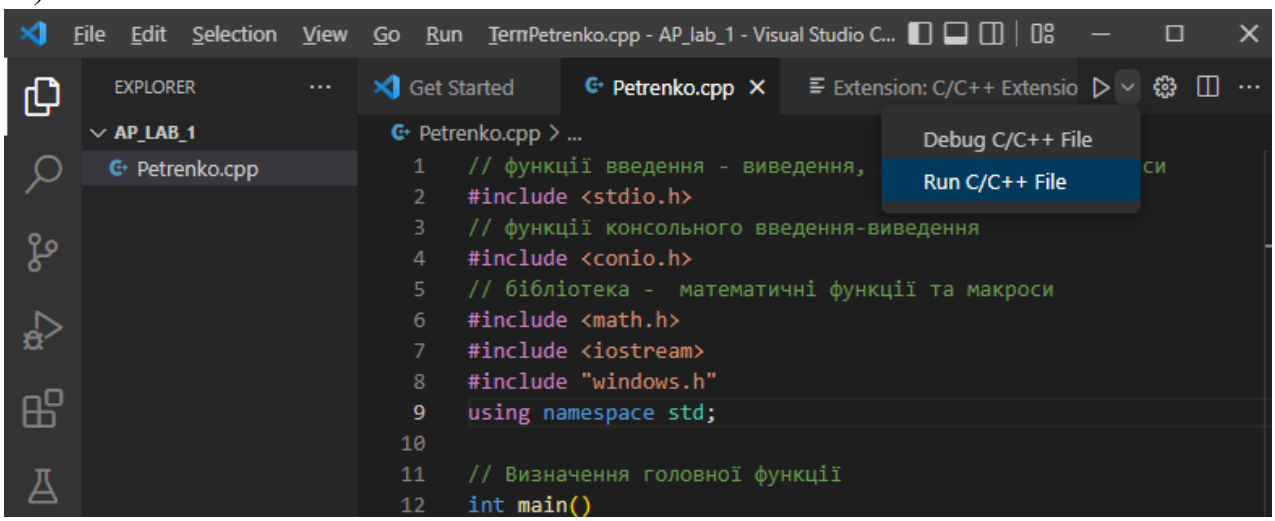

Рис. 5. Запуск файлу на виконання у VS Code

Під час першого запуску буде запропоновано вибрати відповідний компілятор. Цей компілятор стає компілятором "за замовчуванням", встановленим у вашому tasks.json файлі.

Щоб зберегти файл можна скористатись комбінацією Ctrl+S . Можна також ввімкнути функцію автозбереження для автоматичного збереження змін у файлі, встановивши прапорець « **Автозбереження»** в головному меню **«Файл»** .

Структура програми

У загальному випадку програма мовою С++ має таку структуру:

1. Коментар про призначення програми.
2. Підключення бібліотек.
3. Визначення препроцесора (констант, макросів тощо).
4. Оголошення прототипів функцій.
5. Оголошення глобальних змінних.
6. Визначення головної функції.
   - 6.1. Оголошення локальних змінних.
   - 6.2. Виклик функцій.
   - 6.3. Оператори.
7. Визначення інших функцій.

Основні функції мови С визначаються в системних бібліотеках і описуються в заголовних файлах (табл. 1). Список заголовних файлів визначений стандартом мови.

Нижче наведені деякі з них

- conio.h – функції консольного введення-виведення
- math.h – математичні функції та макроси
- stdio.h – функції введення-виведення, а також типи та макроси
- string.h – функції для роботи з рядками символів
- time.h – типи і функції, пов’язані з датою та часом

Заголовні файли мають розширення .h і підключаються директивою #include. При підключенні заголовного файла його текст, а з ним і текст відповідного сфайла, автоматично вставляється у програму замість рядка з відповідною директивою #include. При компіляції програми усі підключені заголовні файли з файлами їхньої реалізації перекомпільовуються, тому підключення надмірної кількості таких файлів сповільнює компіляцію програми. Хоча в заголовних файлах містяться всі описи стандартних функцій, у код програми включаються лише ті функції, які використовуються у програмі.

Коментарі

Коментарі використовуються лише для пояснень і жодних дій у програмі не виконують, тому що ігноруються компілятором.

Однорядкові коментарі позначаються символами // і записуються перед текстом коментаря:

```
//  Текст після цього символа і до кінця рядка є коментарем
//  Це також однорядковий коментар
```

Багаторядкові коментарі записуються між символами /* та */.

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

Також коментарі часто використовуються під час відлагодження програми для тимчасового “вимкнення” певного її фрагмента.

Введення/виведення даних

Функція cin

Функція cin використовується для введення даних. Якщо функція введення має декілька параметрів, то дані потрібно вводити послідовно через пробіл, а після останнього введеного значення необхідно натиснути клавішу Enter. А коли вводяться числові та символьні значення відмінні від пропуску (пробілу), то пробіл як розділовий символ не використовується, оскільки він також є звичайним символом.

Приклад.
```
int n; double m;
// введення значень окремо cin >> n; cin >> m;
// введення значень через пробіл cin >> n >> m;
```

Функція cout

Для виведення потоку використовується функція cout. У цій функції також можуть використовуватись маніпулятори, тобто функції, які з потоковими об’єктами cin, cout записуються як звичайні змінні, проте виконують певні операції над потоком. При використанні маніпуляторів потрібно підключити бібліотеку iomanip:

```
#include <iomanip>
```

До основних маніпуляторів належать такі:

- endl – забезпечує переведення курсору на новий рядок;
- setw(n) – задає ширину поля, де n – це значення ширини, ціле число;
- setprecision(n) – задає точність значення, де n – це значення точності, ціле число;
- dec, hex, oct – задають системи числення (десяткову, шістнадцяткову та вісімкову відповідно).

Функція printf()

Для запису даних у вихідний потік stdout використовують функцію printf(). Аргументами функції printf() є форматний рядок та список змінних, значення яких виводиться на екран. Прототип функції printf() має такий вигляд:

```
int printf (const char *format [, argument, ...]);
```

де format – форматний рядок; argument – змінні або вирази, значення яких виводяться на екран.

Форматний рядок функції printf складається зі специфікацій форматів.

Специфікація форматованого виведення має вигляд:

% [flags] [width] [.precision] [size_modifier] type_char де flags={-,+|пропуск,#} – набір прапорців;

Символи перетворення даних для форматного виведення

Символ	Призначення
d	десяткове ціле число
o	беззнакове вісімкове ціле число (без префікса 0)

x, X	беззнакове шістнадцяткове ціле число (без префікса 0x та 0X)
i	ціле число у 10-ій, 8-ій або 16-ій системі числення
u	беззнакове десяткове ціле число
c	один символ
s	рядок символів до символа \0 або у кількості, заданною точністю
f	дійсне десяткове число з фіксованою крапкою, за замовчуванням точність дорівнює 6
e, E	дійсне десяткове число з експонентою (з плаваючою крапкою), за замовчуванням точність дорівнює 6
g, G	дійсне десяткове число; використовує %e чи %E, якщо експонента менше ніж –4, або більше чи рівна точності; в іншому випадку використовується %f
p	вказівник (адреса в 16-ій системі числення)
%	аргумент не перетворюється; друкується символ %

Крім специфікацій форматів у форматному рядку можна задавати інші друковані та керуючі символи, які будуть виведені на екран. Наприклад:

- \n – перехід на новий рядок
- \t – горизонтальна табуляція
- \0 – нульовий (порожній) символ

Функція scanf()

Функція scanf() зчитує введений з вхідного потоку stdin текст, транслює поля введення в двійкові значення відповідно до правил перетворення, що задаються форматним рядком. Значення, що утворилися після трансляції, запам’ятовуються за вказаними адресами, які посилаються на змінні програми.

Функція scanf() повертає кількість змінних, що отримали значення. Прототип функції scanf() має такий вигляд:

int scanf (const char *format [, address, ...]);

де format – форматний рядок; address – адреси змінних, значення яких вводяться з клавіатури. Форматний рядок функції scanf() складається із специфікацій форматів. Специфікація форматного введення даних має такий вигляд:

% [flag] [width] [size_modifier] type_char

де flag=‘*’ – пропуск введення поля даних із вхідного потоку; width – кількість символів, які будуть прочитані із вхідного потоку; size_modifier={F|N|h|l|L} – модифікатор розміру типу; type_char={d|D|i|I|o|O|u|U|x|X|f|e|g|E|G|c|s|p} – символ типу. Для форматного введення використовуються ті ж символи, що і для форматного виведення. Наприклад:

```
char k, ch;
    scanf("%c", &k);  // Вхідний потік: 5. Результат k = 5
    scanf("%c", &ch); // Вхідний потік: a. Результат ch = ‘a’
```

Завдання

1. Запустити середовище програмування (Visual Studio Code, MS Visual Studio чи X-Code).

2. Створити порожній консольний C++ -проект. Зберегти проект.

3. Додати файл з початковим кодом (назва файлу має відповідати Вашому прізвищу).

4. Робота з кодом.

4.1. Записати наступний програмний код:

```
//функції введення - виведення, а також типи та макроси
#include <stdio.h>
//функції консольного введення-виведення
#include <conio.h>
//бібліотека -  математичні функції та макроси
#include <math.h>
#include <iostream>
#include "windows.h"
using namespace std;

//Визначення головної функції
int main() {
//Оголошення локальних змінних
    int X, Y, Z, rs;
    //зчитування даних
    cout << "Enter an time X / integer: "<< endl; //window
    cin >> X;
    cout << "Enter an time Y / integer: "<< endl; //erase
    cin >> Y;
    cout << "Enter an time Z / integer: "<< endl; //image
    cin >> Z;
    //обчислення
    rs = ((120 - X - Y - Z) / 5);
    //виведення результату
    cout << "Amount of programs = %d"<< rs;
    // виклик функцій
//_getch();
return 0;
}
```

Запустити програму на виконання.

4.2. Змінити вміст файлу на наступний програмний код:

```
#include <stdio.h>
#include <conio.h>
#include <math.h>
int main() {
 double z=pow(5,4);
 int i, a=1, b=1, c=1;
 scanf("%d", &a);
 for (i=1; i<a; i++) {
   c=a+b;
   b+=a;
   printf("a=%d\tb=%d\tc=%d\n",a,b,c);
  }
getch();
return 0;
}
```

Додати до програми коментарі-пояснення, як в попередньому прикладі. Запустити програму на виконання.

5. Робота з помилками.

5.1. Видалити з коду програми або закоментувати підключення деякої бібліотеки:

- запустити програму на виконання у режимі “виконання без відлагодження”.
- запустити програму на виконання у режимі “виконання з відлагодженням”;
- проаналізувати отримані повідомлення, переглянути та виправити помилки.

5.2. Повторити ці дії, видаливши спочатку будь-яку дужку, а потім будь-яку кому.

5.3. Проаналізувати повідомлення про помилки. Для кожної помилки слід вказати:

a. Повний текст повідомлення системи про помилку (англійською мовою).
b. Що означає це повідомлення (переклад тексту повідомлення).
c. Дії, які слід виконати для виправлення помилки

6. Оголосити та визначити власну змінну, зчитати значення з клавіатури, змінити його програмно та вивести отримане значення.

7. Точка переривання:

7.1.Встановити точку переривання;
7.2.Виконати покрокове виконання програми в режимі відлагодження;
7.3.Здійснити контроль значень змінних;
7.4.Видалити точку переривання.