Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Інститут телекомунікацій, радіоелектроніки та електронної техніки кафедра «Радіоелектронні пристрої та системи»



Звіт з лабораторної роботи №13-14

з дисципліни «Програмування»

Підготував:

ст. групи АП-11

Білоніжко Віталій

Прийняв:

Чайковський І. Б,

**Львів – 2024**

**Тема:**

Структура функції. Локальні та глобальні змінні. Класи пам’яті.

**Мета:**

навчитися використовувати функції у процесі програмування,

розуміти особливості використання локальних та глобальних змінних та

специфікаторів різних класів пам’яті.

**Теоретичні відомості:**

При програмуваннi будь-яких задач, крiм найпростiших, постiйно виникає

потреба виконувати в кiлькох рiзних мiсцях алгоритму однi й тi самi дiї над рiзними значеннями. Наприклад, нехай потрiбно обчислити площу трьох рiзних трикутникiв, перший з яких має сторони a1, b1, c1, другий – a2, b2, c2, третiй – a3, b3, c3.

**Хід роботи:**

1. Ознайомитися з теоретичними відомостями.

2. Здійснити виконання прикладів, представлених у теоретичних відомостях,

після чого представити скріни їх коду та результати виконання у звіті.

3. Написати програму з використанням функції, яка друкує визначену

кількість символів рядка. Уточнення: дана функція повинна приймати рядок

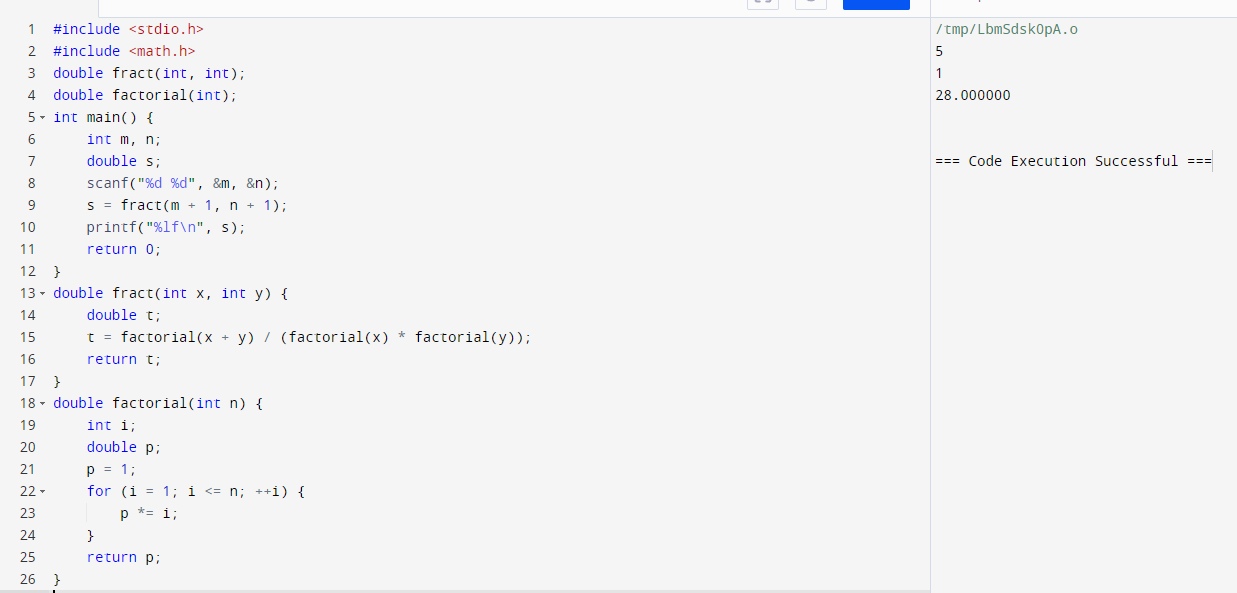
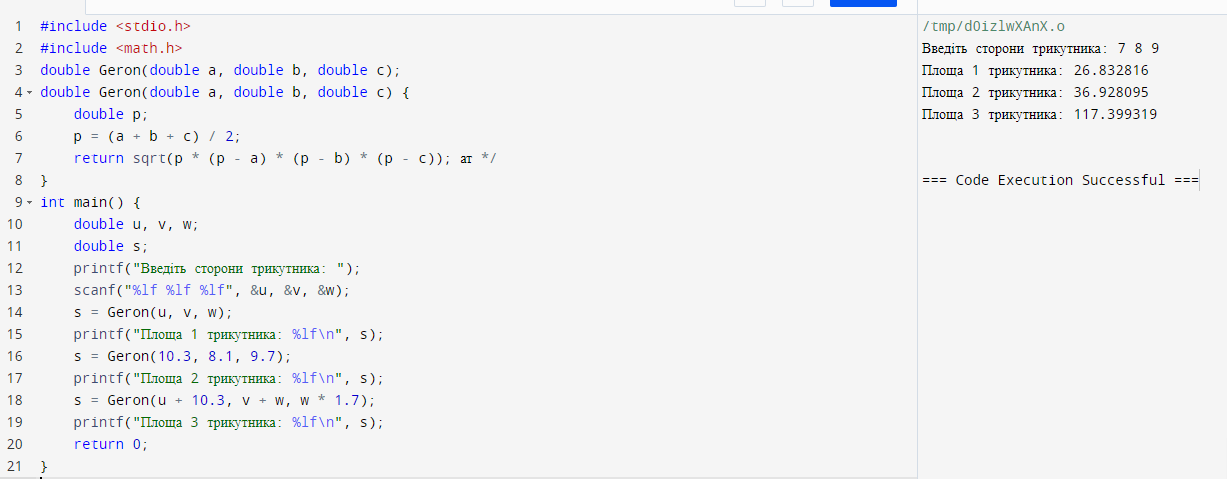
символів і ціле число, яке визначатиме кількість символів, що слід надрукувати.

Скрін коду програми та результати її виконання представити у звіті.

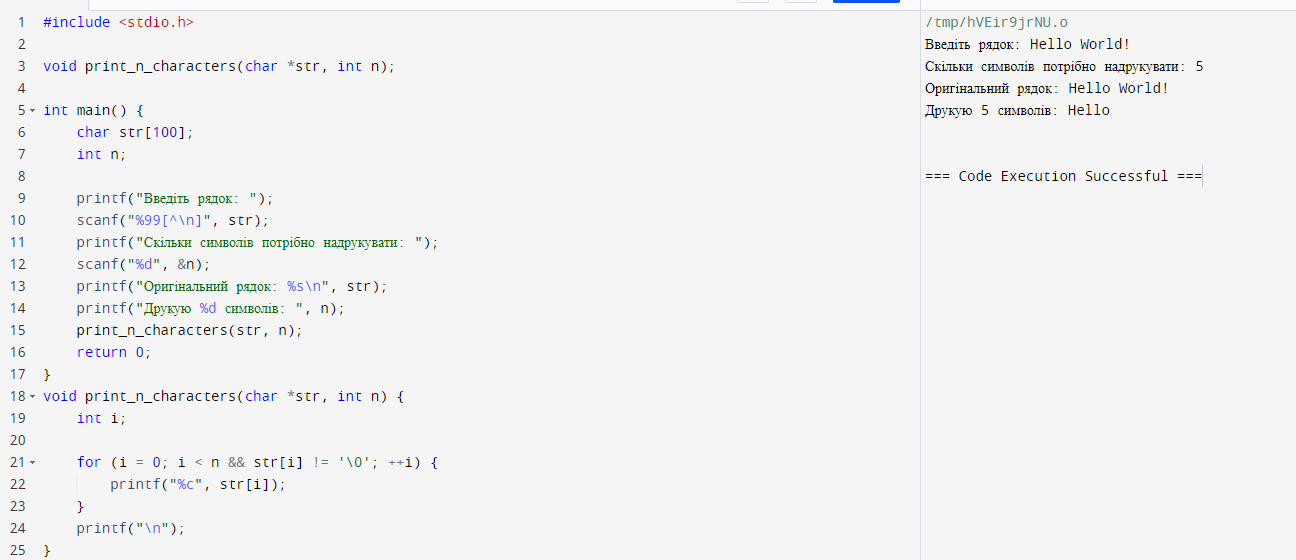
4. Оформити звіт.

**Виконання роботи:**

**Завдання 2**

****

**Завдання 3**

****

**Контрольні запитання:**

1. Дайте визначення поняття функція в мові С.

Функція в мові С — це самостійний блок коду, що виконує певну задачу і може бути викликаний з інших частин програми. Функція має визначене ім'я, може приймати аргументи та повертати значення. Основна мета використання функцій — організація коду, зменшення дублювання і підвищення читабельності та підтримуваності програмного коду.

Приклад функції в C:

#include <stdio.h>

// Оголошення функції

int add(int a, int b);

int main() {

int result = add(3, 4); // Виклик функції

printf("Result: %d\n", result);

return 0;

}

// Визначення функції

int add(int a, int b) {

return a + b;

}

2. Поясніть призначення типу void.

Тип `void` в мові С використовується у кількох контекстах:

* Функції, що не повертають значення:

void printMessage() {

printf("Hello, World!\n");

}

Тут `void` вказує, що функція `printMessage` не повертає ніякого значення.

- Функції без аргументів:

void foo(void) {

// код функції

}

Тут `void` вказує, що функція `foo` не приймає жодних аргументів.

- Указівники на невизначений тип:

void \*ptr;

Указівник `void ` може вказувати на об'єкти будь-якого типу і часто використовується в загальних структурах даних і функціях.

3. Чим локальні змінні відрізняються від глобальних?

Локальні змінні:

- Оголошуються всередині функцій або блоків коду.

- Доступні тільки в межах блоку, де вони оголошені.

- Ініціалізуються щоразу при виклику функції, де вони оголошені.

- Автоматично знищуються після завершення виконання функції.

Приклад:

void exampleFunction() {

int localVar = 10; // Локальна змінна

}

Глобальні змінні:

- Оголошуються поза всіма функціями.

- Доступні у всій програмі після їх оголошення.

- Зберігають своє значення протягом всього часу виконання програми.

- Використовуються для зберігання даних, які повинні бути доступні з різних функцій.

Приклад:

int globalVar = 20; // Глобальна змінна

void anotherFunction() {

globalVar = 30; // Доступ до глобальної змінної

4. Яку функцію називають рекурсивною?

Рекурсивною функцією називають функцію, яка викликає саму себе, прямо або опосередковано. Рекурсія використовується для розв'язання задач, які можна розбити на підзадачі аналогічного типу. Важливою умовою рекурсивної функції є наявність базового випадку, який зупиняє рекурсію.

Приклад рекурсивної функції для обчислення факторіалу числа:

int factorial(int n) {

if (n == 0) {

return 1; // Базовий випадок

} else {

return n \* factorial(n - 1); // Рекурсивний виклик

}

}

```

5. Назвіть переваги і недоліки використання рекурсивної функції.

Переваги:

- Зручність: Рекурсивні функції часто легше читати та розуміти, особливо для задач, які природно розбиваються на підзадачі (наприклад, обхід дерев, обчислення факторіала).

- Елегантність: Код стає більш компактним та елегантним.

Недоліки:

- Надмірне використання пам'яті: Кожен рекурсивний виклик створює новий запис у стеку викликів, що може призвести до переповнення стека (stack overflow) при великій глибині рекурсії.

- Продуктивність: Рекурсивні функції можуть бути менш ефективними в плані швидкодії порівняно з ітеративними рішеннями через накладні витрати на керування стеком викликів.

6. Назвіть специфікатори класів пам’яті, які використовуються у мові С у відповідях не використовувати \*.

Специфікатори класів пам'яті в мові С визначають область видимості і тривалість збереження змінних та функцій:

- auto: За замовчуванням використовується для локальних змінних. Означає, що змінна автоматично створюється при вході в блок і знищується при виході з нього.

void function() {

auto int localVar = 10;

}

- static: Використовується для локальних змінних і глобальних змінних. Для локальних змінних означає, що змінна зберігає своє значення між викликами функції. Для глобальних змінних означає, що змінна або функція видима тільки в межах файлу.

void function() {

static int count = 0;

count++;

printf("Count: %d\n", count);

}

- extern: Використовується для оголошення глобальних змінних або функцій, визначених в іншому файлі. Це дозволяє використовувати змінні або функції в різних файлах.

extern int globalVar; // Оголошення глобальної змінної, визначеної в іншому файлі

- register: Пропонує компілятору зберігати змінну в регістрі процесора, а не в основній пам'яті, для більш швидкого доступу. Використання цього специфікатора не гарантує, що змінна буде збережена в регістрі.

void function() {

register int fastVar = 5;

}

Ці специфікатори допомагають керувати пам'яттю та продуктивністю програм на мові С.