**Міністерство освіти і науки України**

**Львівський національний університет імені Івана Франка**

**Факультет електроніки та комп'ютерних технологій**

**Звіт**

Про виконання лабораторної роботи №7

**“Функції”**

Виконав:

Студента групи ФЕП-11

**Поцелуйко Назарій**

Викладач:

**асист. Кужій Ю.В.**

Львів – 2024

*Мета роботи:*

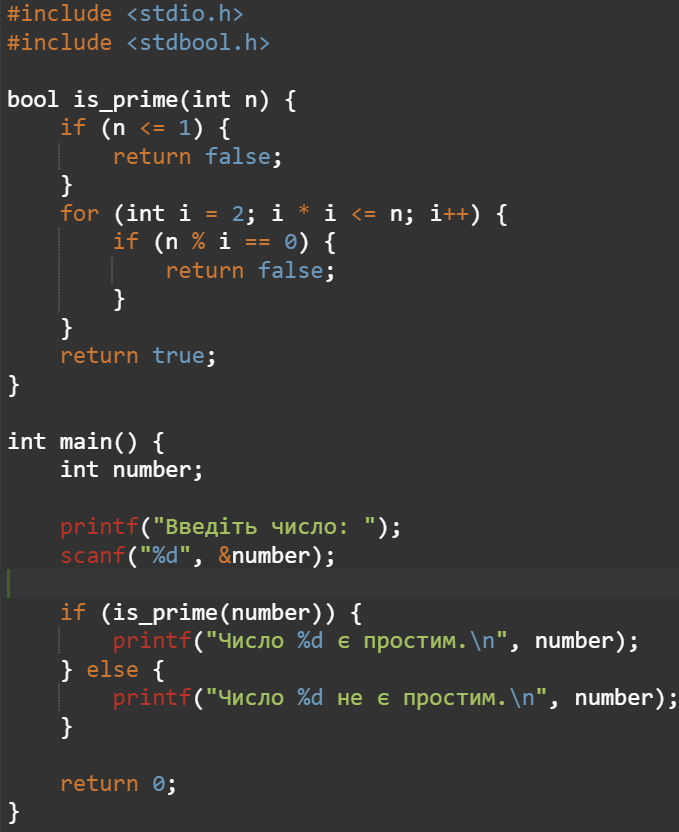
Вивчити поняття і застосування функцій.

*Обладнання та програмне забезпечення:*

– IBM сумісна персональна обчислювальна машина;

– онлайн компілятор мови програмування Сі, доступний за посиланням [**https://www.onlinegdb.com/online\_c\_compiler**](https://www.onlinegdb.com/online_c_compiler)

**ПОРЯДОК ВИКОНАННЯ РОБОТИ**

**2. Створити функцію, яка перевіряє, чи введені з консолі число є простим**

**1. Підключення бібліотек:**

**#include <stdio.h>**

**#include <stdbool.h>**

**#include <stdio.h> —** це стандартна бібліотека вводу/виводу в C, яка дозволяє використовувати функції printf і scanf для роботи з консоллю.

**#include <stdbool.h>** — бібліотека для використання булевих змінних (true і false), що спрощує роботу з логічними значеннями.

**2. Функція для перевірки простого числа:**

**bool is\_prime(int n) {**

**if (n <= 1) {**

**return false;**

**}**

**for (int i = 2; i \* i <= n; i++) {**

**if (n % i == 0) {**

**return false;**

**}**

**}**

**return true;**

**}**

**is\_prime(int n)** — функція, яка приймає ціле число n і повертає true (просте число) або false (не просте число).

**if (n <= 1)** — перевіряється, чи є число меншим або рівним 1, тому що прості числа повинні бути більшими за 1.

**for (int i = 2; i \* i <= n; i++)** — цикл перевіряє, чи ділиться число на інші числа від 2 до квадратного кореня з n. Якщо число ділиться без залишку, воно не є простим.

Якщо не було знайдено жодного дільника, функція повертає **true,** що означає, що число **просте.**

**3. Основна функція програми:**

**int main() {**

**int number;**

**printf("Введіть число: ");**

**scanf("%d", &number);**

**int main()** — це точка входу в програму.

**int number;** — оголошується змінна number, яка буде містити введене користувачем число.

**printf("Введіть число: ");** — виводиться запит на введення числа.

**scanf("%d", &number);** — функція scanf зчитує введене число з консолі та зберігає його у змінну number.

**4. Виклик функції та виведення результату:**

**if (is\_prime(number)) {**

**printf("Число %d є простим.\n", number);**

**} else {**

**printf("Число %d не є простим.\n", number);**

**}**

**return 0;**

**}**

Викликається функція **is\_prime(number)**, щоб перевірити, чи є число простим.

Якщо функція повертає **true**, виводиться повідомлення, що число є **простим.**

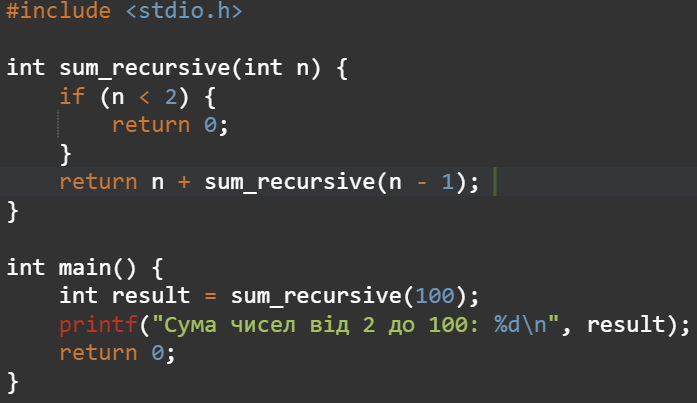
Якщо повертає **false,** виводиться, що число не є простим.

**return 0;** означає успішне завершення програми.

**Приклад роботи:**

****

**3. Написати програму, котра рекурсивно вираховує суму чисел від Вашого порядкового номера в журналі до 100.**

****

**1. Функція sum\_recursive(int n):**

Функція приймає **ціле число n** і повертає суму **всіх** чисел від **2** до **n.**

**Базовий випадок:** якщо **n < 2,** функція повертає **0 (**оскільки потрібно починатиз **2).**

**Рекурсивний виклик:** додається число **n** до результату виклику функції **з n - 1.**

**2. Основна функція main():**

Викликається рекурсивна функція з початковим аргументом **100.**

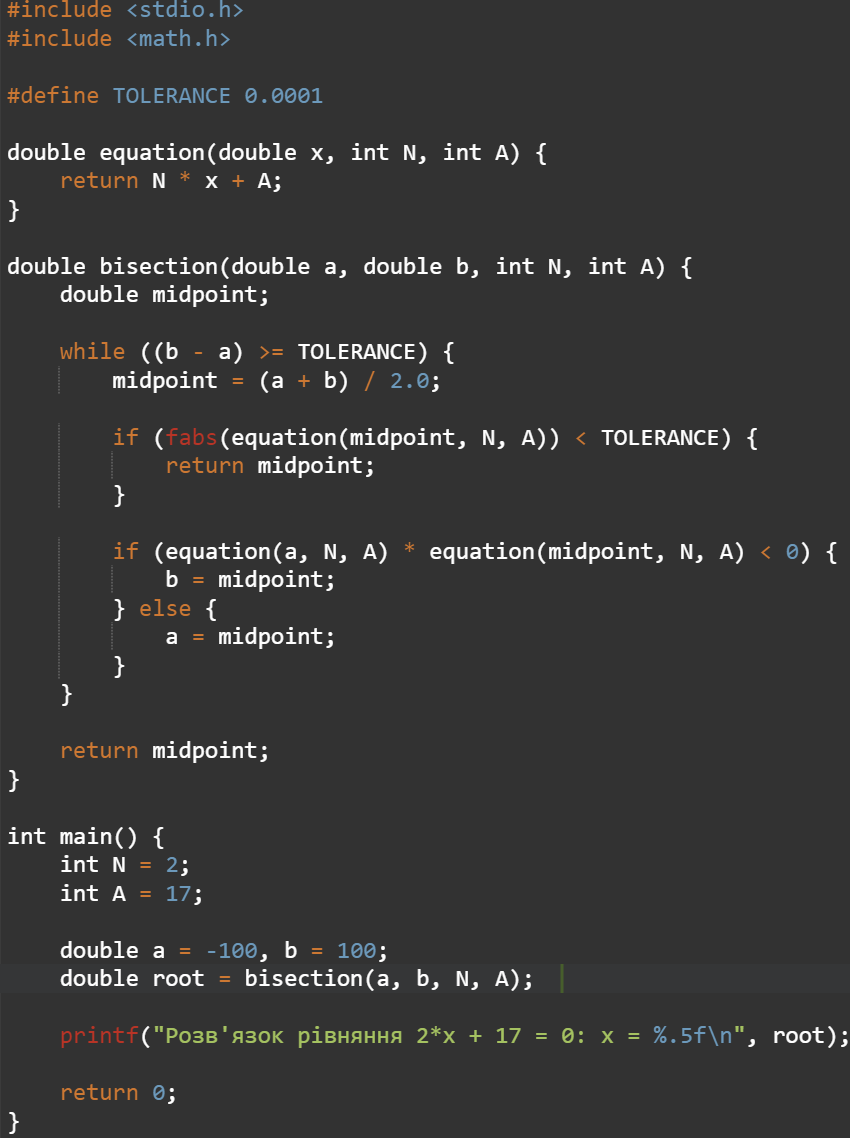
Результат зберігається у змінну **result** і виводиться на екран за допомогою **printf.**

Таким чином, програма*рекурсивно*обчислює суму чисел від **2** до **100.**

**Приклад роботи:**

****

**4.** Написати програму, котра знаходить розв’язок рівняння N\*x + А = 0 методом половинного ділення. N – Ваш порядковий номер в журналі (2), А – Ваш вік (17). Розв’язок знаходиться в інтервалі [–100..100].

****

**1. Константа TOLERANCE:**

**Визначає точність, з якою потрібно знайти корінь. У цьому випадку це 0.0001, що означає, що ми шукаємо розв'язок з такою точністю.**

**2. Функція equation(double x, int N, int A):**

**Обчислює значення рівняння . У програмі використовуються значення та**

**при кожному виклику функція повертає значення цього рівняння для даного значення x.**

**3. Функція bisection(double a, double b, int N, int A):**

**Реалізує метод половинного ділення для пошуку кореня рівняння.**

**Вхідними параметрами є:**

**a і b — початкові границі інтервалу, на якому шукаємо корінь.**

**N і A — коефіцієнти рівняння.**

**У циклі звужуємо інтервал до тих пір, поки різниця між межами інтервалу не стане меншою за задану точність.**

У кожній ітерації обчислюється **середня точка (midpoint),** і перевіряється, чи є вона **коренем (значення рівняння близьке до 0).**

**Якщо корінь не знайдено, вибирається новий інтервал на основі знаків значень рівняння на кінцях інтервалу.**

**4. Основна функція main():**

**Визначає початкові значення змінних N = 2 і A = 17, а також інтервал пошуку кореня .**

**Викликається функція bisection, яка знаходить корінь рівняння.**

**Після знаходження кореня програма виводить результат з точністю до п'яти знаків після коми.**

**Принцип роботи методу половинного ділення:**

**1. Початково задається інтервал , де передбачається, що є корінь.**

**2. Обчислюється середина цього інтервалу.**

**3. Якщо значення функції в середині інтервалу є досить близьким до нуля (з врахуванням точності), ця точка є коренем**

**4. Якщо ні, то інтервал звужується в залежності від знаків функції на краях інтервалу (обирається та половина, де знак функції змінюється).**

**5. Процес повторюється, доки не досягається необхідна точність.**

**У цьому прикладі метод половинного ділення працює з рівнянням і знаходить, що корінь рівняння — це , що задовольняє рівняння.**

**Приклад роботи:**

****

**Висновок: Отже, практикувавшись з мовою Сі, я створював функції, які перевіряли чи число є простим, яка вираховувала суму чисел від мого порядкованого номера до 10 та які розв’язували рівняння методом половинного ділення.**