**Міністерство освіти і науки України**

**Львівський національний університет імені Івана Франка**

**Факультет електроніки та комп'ютерних технологій**

**Звіт**

Про виконання лабораторної роботи №6

**“Цикли”**

Виконав:

Студент групи ФЕП-11

**Поцелуйко Назарій**

Викладач:

**асист. Кужій Ю.В.**

Львів – 2024

*Мета роботи:*

**вивчити поняття та застосування циклів**

*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. Написати програму, котра дозволяє ввести речення і підрахувати кількість голосних і приголосних у ньому.**

****

**1. Бібліотеки**

* **#include <stdio.h>** — підключає стандартну бібліотеку для вводу/виводу, яка дозволяє використовувати функції для введення даних, такі як printf і fgets.
* **#include <ctype.h>** — забезпечує функції для роботи з символами, такі як tolower(), яка перетворює літеру у нижній регістр.
* **#include <locale.h>** — використовується для налаштування локалізації програми, щоб вона могла працювати з українськими символами.

**2. Функції для перевірки символів**

* **Функція is\_vowel(char c)**:
  + Перевіряє, чи є символ голосною буквою.
  + Спочатку символ приводиться до нижнього регістру за допомогою функції tolower(), щоб порівнювати без врахування регістру.
  + Голосні букви підтримуються як англійські (a, e, i, o, u), так і українські (а, е, є, и, і, ї, о, у, ю, я).
  + Якщо символ є голосною буквою, функція повертає 1, в іншому випадку — 0.
* **Функція is\_consonant(char c)**:
  + Перевіряє, чи є символ приголосною буквою.
  + Також спочатку символ приводиться до нижнього регістру.
  + Для англійських приголосних перевіряється, чи знаходиться символ у діапазоні від 'b' до 'z' і не є голосною.
  + Для українських приголосних перевіряється, чи знаходиться символ у діапазоні від 'б' до 'щ', і виключаються спеціальні символи 'й' і 'ь', які не вважаються приголосними.

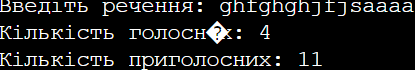
**3. Основна частина програми**

* **setlocale(LC\_ALL, "")** — налаштовує програму на використання системної локалізації, що дозволяє коректно обробляти українські літери.
* **Оголошення змінних**:
  + **char sentence[256];** — масив символів для зберігання введеного речення.
  + **int vowel\_count = 0, consonant\_count = 0;** — лічильники для підрахунку кількості голосних і приголосних.
* **Введення речення**:
  + Функція **fgets()** зчитує введене речення від користувача і зберігає його у змінну sentence.
* **Цикл підрахунку**:
  + Програма перебирає кожен символ у введеному реченні.
  + Для кожного символу перевіряється, чи є він голосною за допомогою функції is\_vowel(). Якщо так — збільшується лічильник голосних.
  + Якщо символ не є голосною, перевіряється, чи є він приголосною за допомогою функції is\_consonant(). Якщо так — збільшується лічильник приголосних.

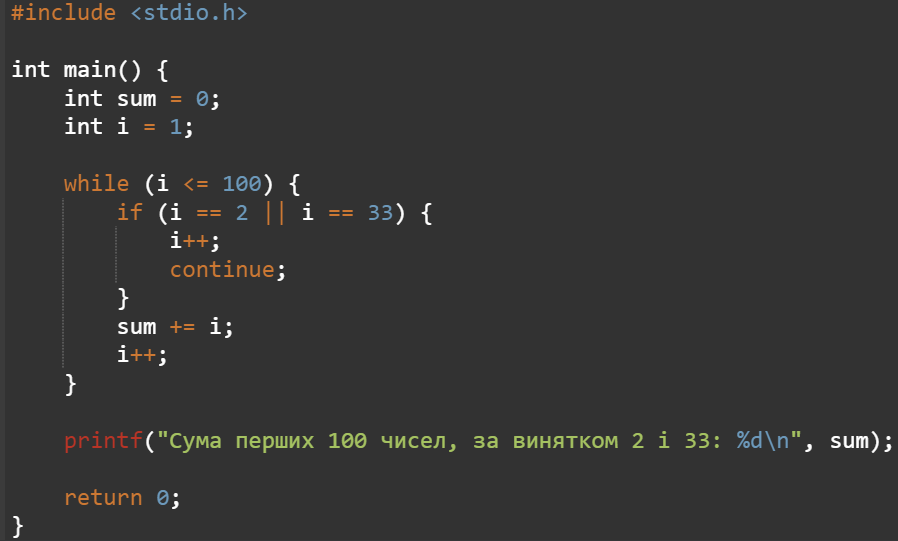
**4. Виведення результатів**

* Після завершення циклу програма виводить результати:
  + **printf("Кількість голосних: %d\n", vowel\_count);** — виводить кількість голосних у реченні.
  + **printf("Кількість приголосних: %d\n", consonant\_count);** — виводить кількість приголосних.

**Приклад роботи:**

****

**3. Написати програму, котра в циклі while вираховує суму перших 100-та чисел за винятком Вашого порядкового номера в журналі (2) і числа 33.**

****

**Лічильник i та змінна sum**

Ми оголошуємо змінну **sum**, яка зберігатиме суму всіх чисел. Початкове значення — 0.

Змінна i використовується як лічильник у циклі, щоб пройти всі числа від 1 до 100.

**Цикл while**

Цикл починається з i = 1 і працює, поки i <= 100. Це означає, що ми будемо перевіряти кожне число від 1 до 100.

**В тілі циклу перевіряється умова:**

c

if (i == 2 || i == 33)

Якщо значення i дорівнює **2** або **33**, ці числа будуть пропущені. Оператор continue вказує програмі пропустити решту коду і перейти до наступної ітерації циклу (до наступного числа).

Додавання до суми

Якщо число не є **2** або **33**, воно додається до змінної **sum** за допомогою:

**c**

**sum += i;**

Це означає "додай поточне значення i до змінної sum".

Збільшення лічильника

Після кожної перевірки і додавання (або пропуску) числа до суми, лічильник i збільшується на **1**:

c

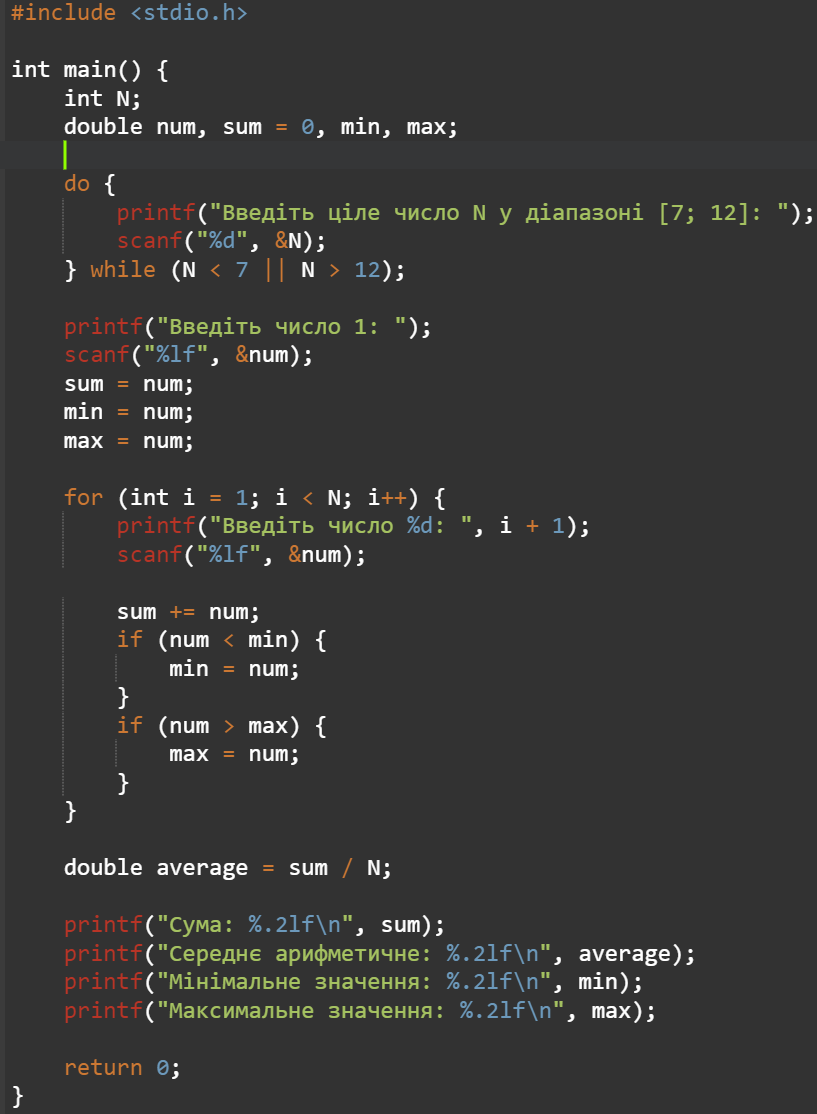
i++;

Таким чином, ми переходимо до наступного числа.

**Результат виконання:**

****

**4. Написати програму, котра дозволя користувачу задати ціле число NЄ[7; 12], далі в циклі ввести N дійсних чисел, розрахувати і вивести суму введених чисел, середнє арифметичне, мінімальне і максимальне значення.**

****

**Введення числа N**:

* Користувачу пропонується ввести ціле число N у діапазоні від 7 до 12.
* Для цього використовується цикл do-while, який гарантує, що програма не вийде з нього, поки користувач не введе правильне значення N.

**Введення дійсних чисел**:

* Після задання N, програма просить користувача ввести N дійсних чисел.
* Перше число вводиться окремо для ініціалізації змінних для суми, мінімуму і максимуму.

**Обчислення суми, мінімуму, максимуму**:

* При кожному введенні числа, воно додається до змінної sum.
* Також кожне нове введене число перевіряється, чи є воно меншим за поточне мінімальне значення, або більшим за максимальне, щоб оновити відповідні змінні.

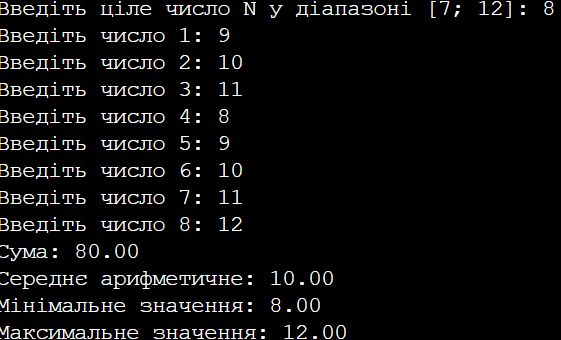
**Обчислення середнього арифметичного**:

* Середнє арифметичне обчислюється як сума всіх чисел, поділена на кількість N.

**Виведення результатів**:

* Після введення всіх чисел програма виводить результати: суму, середнє арифметичне, мінімальне та максимальне значення, з точністю до двох знаків після коми.

**Приклад роботи:**

****

**Висновок: Отже, з використанням мови програмуванння Сі, я навчився використовувати цикли в ньому та в загальному практику з програмування на даній мові.**