МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

Кафедра КІТАМ
Дисципліна «Програмування»

Звіт

з лабораторної роботи №4
«Умовні оператори мови С++»

Виконала:
ст.гр. АКТСІз-23-1
Рацебарська А.Д.

Прийняла:
доц. Максимова С.С.,

Харків 2023

4. ОПЕРАТОРИ ЦИКЛІВ МОВИ С++

2.1 Мета роботи:
Вивчити особливості використання операторів циклу while, for і do while.

2.2 Теоретичні відомості:

Оператори циклів є невід'ємною частиною мови програмування С++, дозволяючи виконувати повторення певних дій. Вони не тільки спрощують код, але й надають потужні інструменти для ефективного управління ітераціями. У С++ існують три основних типи циклів: **for**, **while** та **do-while**.

1. Цикл for: Це найбільш структурований цикл. Цей цикл часто використовується, коли точно відома кількість ітерацій. Його синтаксис складається з трьох частин: ініціалізація, умова продовження циклу та інкрементування/декрементування. Цикл **for** є особливо корисним для перебору елементів у масивах та контейнерах. Наприклад: **for(int i = 0; i < n; i++) { /* код */ }**. Цікавим є той факт, що усі три компоненти циклу **for** не є обов'язковими, що дозволяє створювати цикли з різними формами поведінки.

2. Цикл while: Цей цикл використовується для виконання блоку коду, поки задана умова є істинною. Цикл **while** краще підходить для ситуацій, де кількість ітерацій не відома заздалегідь. Він перевіряє умову перед кожною ітерацією, що може впливати на продуктивність у випадку великої кількості ітерацій.

3. Цикл do-while: Цей цикл схожий на цикл **while**, але з однією ключовою відмінністю: умова перевіряється після виконання ітерації, гарантуючи, що блок коду буде виконаний хоча б один раз. Це корисно в сценаріях, де необхідно спочатку виконати дію, а потім перевірити, чи слід продовжувати. Наприклад, це може бути використано для меню програми, яке має показатися принаймні один раз.

Нюанси та ефективність

- Безкінечні цикли: Цикли можуть бути безкінечними, якщо умова виходу з них ніколи не виконується. Це може бути як помилкою, так і навмисним рішенням для створення постійно виконуваного коду, наприклад, в основному циклі гри або серверної програми.
-
- Керування циклами: У С++ існують оператори **break** та **continue**, які дозволяють більш гнучко управляти виконанням циклів. **Break** негайно завершує цикл, тоді як **continue** завершує поточну ітерацію і переходить до наступної.
-
- Вибір між for та while: Хоча обидва ці цикли можуть використовуватися замінно, важливо вибирати **for** для ситуацій, де кількість ітерацій заздалегідь відома або

має чітку межу. Цикл **while** ідеально підходить для ситуацій, де ітерації повинні продовжуватися до зміни певного стану або виконання умови.

2.3 Хід роботи:

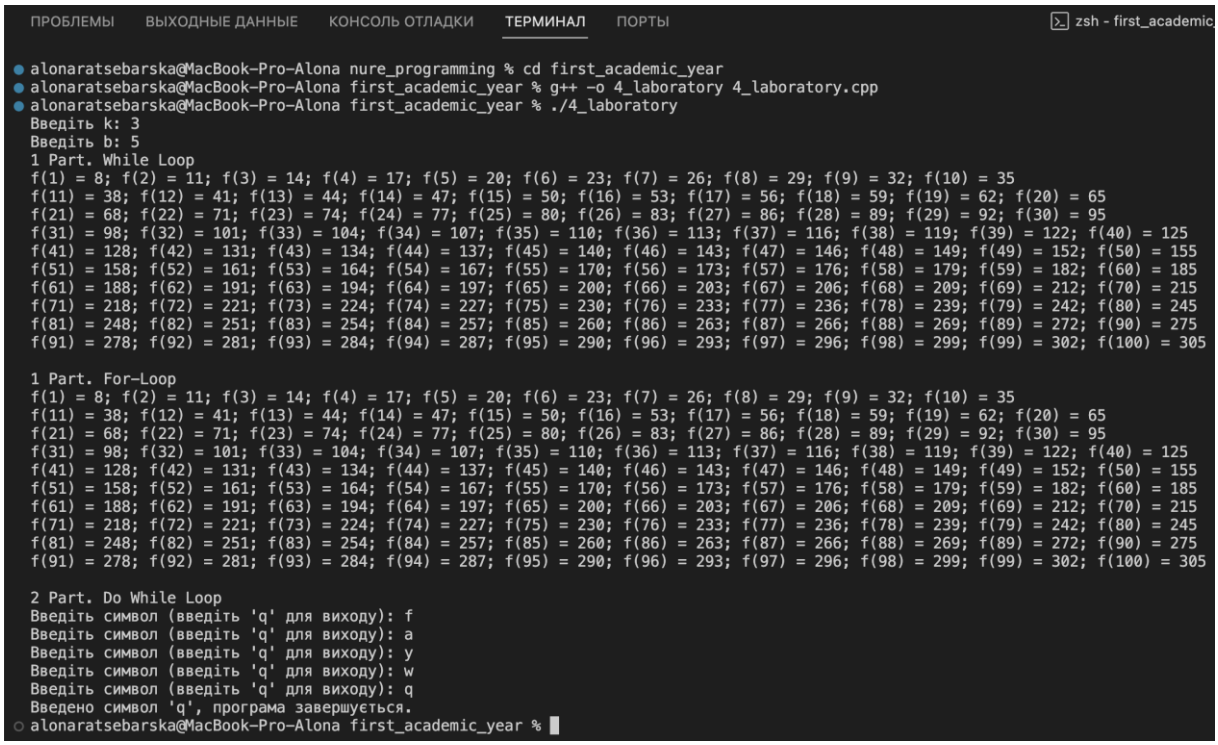
Варіант **2**

Написати програму на С++ :

1.Знайти за допомогою while f(x) = kx + b, x =1,2,...100 (Оператор while та for)

2.Написати програму введення довільних символів до тих пір,
поки не буде введений символ q (Оператор do while)

```cpp
#include <iostream>
using namespace std;

int main() {
    double k, b;
    cout << "Введіть k: ";
    cin >> k;
    cout << "Введіть b: ";
    cin >> b;

    // 1. Знаходження f(x) = kx + b
    // Використання оператора while
    cout << "1 Part. While Loop" << endl;
    int x = 1;
    while (x <= 100) {
        double f_x = k * x + b;
        // Доп перевірка яка робить перенос на нову строку після 10 ітерацій (для зручності зчитування інформ.
        if (x % 10 == 0) {
            cout << "f(" << x << ") = " << f_x << endl;
            // Доп перевірка задля пропуску рядка при завершенні циклу (зручність зчитування в консолі
            if (x == 100) {
                cout << "\n";
            }
        } else {
            cout << "f(" << x << ") = " << f_x << "; ";
        }
        x++;
    }

    // Альтернативний варіант за допомогою оператора for
    cout << "1 Part. For-Loop" << endl;
    for (int x = 1; x <= 100; x++) {
        double f_x = k * x + b;
        if (x % 10 == 0) {
            cout << "f(" << x << ") = " << f_x << endl;
            if (x == 100) {
                cout << "\n";
            }
        } else {
            cout << "f(" << x << ") = " << f_x << "; ";
        }
    }

    // 2. Введення символів до 'q' (оператор do while)
    cout << "2 Part. Do While Loop" << endl;
    char ch;
    do {
        cout << "Введіть символ (введіть 'q' для виходу): ";
        cin >> ch;
    } while (ch != 'q');

    cout << "Введено символ 'q', програма завершується." << endl;

    return 0;
}

```



ВИСНОВКИ

У процесі виконання лабораторної роботи оволоділа важливими навичками роботи з циклами **for**, **while** та **do-while**, що є важливими для розробки програм, здатних виконувати повторювані дії та обробляти послідовності даних. Зокрема, навчилася ефективно використовувати цикл **for** для випадків, коли кількість ітерацій заздалегідь відома, та цикл **while** для ситуацій, де ітерації залежать від виконання певних умов. Оволоділа застосуванням циклу **do-while** для гарантії виконання тіла циклу хоча б один раз, що є корисним у створенні інтерактивних консольних програм. У рамках лабораторної роботи було розроблено програму, що демонструє практичне використання цих циклів для розрахунку функції з заданими параметрами та введення даних користувачем до заданої умови.