Міністерство освіти і науки України
Центральноукраїнський національний технічний університет
Механіко-технологічний факультет

ЗВІТ
ПРО ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ № 9
з навчальної дисципліни
“Базові методології та технології програмування”

# РЕАЛІЗАЦІЯ ПРОГРАМНИХ МОДУЛІВ РОЗГАЛУДЖЕННЯ ТА ІТЕРАЦІЙНИХ ОБЧИСЛЮВАЛЬНИХ ПРОЦЕСІВ

ЗАВДАННЯ ВИДАВ
доцент кафедри кібербезпеки
та програмного забезпечення
Доренський О. П.
https://github.com/odorenskyi/

ВИКОНАВ
студент академічної групи КН-24
Мироненко Я.М

ПЕРЕВІРИВ
ст. викладач кафедри кібербезпеки
та програмного забезпечення
Коваленко Анастасія Сергіївна

Кропивницький – 2025

**ТЕМА: РЕАЛІЗАЦІЯ СТАТИСТИЧНИХ БІБЛІОТЕК МОДУЛІВ ЛІНІЙНИХ ОБЧИСЛЮВАЛЬНИХ ПРОЦЕСІВ**

**МЕТА: полягає у набутті ґрунтовних вмінь і практичних навичок реалізації технології модульного програмування, застосування операторів C / C++ арифметичних, логічних, побітових операцій, умови, циклів та вибору під час розроблення статичних бібліотек, заголовкових файлів та програмних засобів у кросплатформовому середовищі Code::Blocks.**

*ВАРІАНТ 1*

— *ЗАДАЧА 9.1* —

Вхід: швидкість вітру (км/год) під час торнадо.
Вихід: категорія торнадо за шкалою Фудзіти та частота їх виникнення.

| КАТЕГОРІЯ | ШВИДКІСТЬ ВІТРУ, км/год | ЧАСТОТА |
|---|---|---|
| F0 | 64 – 116 | 38,9 % |
| F1 | 117 – 180 | 35,6 % |
| F2 | 181 – 253 | 19,4 % |
| F3 | 254 – 332 | 4,9 % |
| F4 | 333 – 418 | 1,1 % |
| F5 | 419 – 512 | менше 0,1 % |

— *ЗАДАЧА 9.2* —

Вхід: температура повітря (у градусах за шкалою Цельсія), зафіксовану о 00:00, 04:00, 8:00, 12:00, 16:00, 20:00 год.
Вихід: середньодобова температура за шкалами Цельсія та Фаренгейта.

Знаючи температуру за шкалою Цельсія, температуру за шкалою Фаренгейта можна розрахувати наступним чином:

$$t_F = 32 + \frac{9}{5} t_C ,$$

де $t_F$ – температура за шкалою Фаренгейта, $t_C$ – температура за шкалою Цельсія.

— *ЗАДАЧА 9.3* —

Вхід: натуральне число $N$ від 0 до 65535.
Вихід: якщо біт $D_0$ числа $N$ рівний 0, кількість двійкових нулів у ньому, інакше — кількість двійкових одиниць*.
*під час підрахунку кількості бінарних 0 або 1 рекомендовано використати тернарний оператор « ? : ».

— *ЗАДАЧА 9.4* —

За введеним користувачем символом "z" викликається `s_calculation()`, "r" – функція задачі 9.1, "s" – функція задачі 9.2, "t" – функція задачі 9.3; якщо користувач вводить інші символи, вони ігноруються, при чому видається звуковий сигнал про помилкове введення. Після цього, якщо користувач за запитом додатка вводить символ "q" або "Q", відбувається вихід з програми, інакше — виконання програми повторюється.

**Аналіз і постановка задач 9.1, 9.2 та 9.3**

**Задача 9.1** передбачає визначення категорії торнадо за шкалою Фудзіти на основі введеної швидкості вітру, а також надання інформації про частоту виникнення торнадо кожної категорії. Основна мета цієї задачі – реалізація алгоритму визначення вхідного параметра (швидкості вітру) відповідно до заданих діапазонів та виведення відповідної категорії та частоти появи.

**Задача 9.2** стосується обчислення середньодобової температури на основі вимірів у певні моменти доби. Додатково необхідно перевести отримане значення в шкалу Фаренгейта. Головна вимога – реалізація алгоритму обчислення середнього арифметичного значення та формули перетворення температури між шкалами Цельсія та Фаренгейта.

**Задача 9.3** передбачає аналіз двійкового представлення натурального числа в діапазоні від 0 до 65535. Якщо молодший (нульовий) біт числа дорівнює 0, необхідно визначити кількість нулів у його двійковому представленні, інакше – кількість одиниць. Це завдання вимагає побітової обробки числа та використання тернарного оператора для визначення необхідного підрахунку.

**Аналіз вимог, проектування архітектури та програмних модулів**

**Задача 9.1:**

Вхідні дані: швидкість вітру (км/год).

Вихідні дані: категорія торнадо за шкалою Фудзіти та частота їх виникнення.

Архітектура: функція, що приймає швидкість вітру, аналізує її та повертає відповідні значення.

Основні етапи: порівняння вхідного значення з діапазонами шкали Фудзіти та виведення відповідних даних.

**Задача 9.2:**

Вхідні дані: шість значень температури (градуси Цельсія) у визначені години.

Вихідні дані: середньодобова температура у градусах Цельсія та Фаренгейта.

Архітектура: функція, яка приймає масив значень температури, обчислює середнє значення та конвертує його у шкалу Фаренгейта.

Основні етапи: обчислення середнього арифметичного значення, конвертація в шкалу Фаренгейта за формулою.

**Задача 9.3:**

Вхідні дані: натуральне число N (0-65535).

Вихідні дані: кількість двійкових нулів або одиниць (залежно від молодшого біта).

Архітектура: функція, яка аналізує двійкове представлення числа та визначає потрібну кількість.

Основні етапи: побітова перевірка молодшого біта, підрахунок потрібних значень за допомогою тернарного оператора.

**Задача 9.4:**

Управління програмою через введення символів: "r" (виконання задачі 9.1), "s" (виконання задачі 9.2), "t" (виконання задачі 9.3), "z" (виклик функції calculation()).

Реакція на некоректне введення – звуковий сигнал про помилку.

Вихід з програми за введенням "q" або "Q", в іншому випадку повторення виконання.

Для забезпечення коректного виконання програми необхідно реалізувати заголовковий файл для визначення нестандартних типів, якщо такі будуть використовуватися.

**Задача 9.1: Категорія торнадо за шкалою Фудзіти**

Набір контрольних прикладів:

1) Вхід: 100 км/год

   Очікуваний результат: Категорія: FO, Частота: 38.9%

2) Вхід: 150 км/год

   Очікуваний результат: Категорія: F1, Частота: 35.6%

3) Вхід: 250 км/год

   Очікуваний результат: Категорія: F2, Частота: 19.4%

4) Вхід: 350 км/год

Очікуваний результат: Категорія: F4, Частота: 1.1%

5) Вхід: 600 км/год

Очікуваний результат: Категорія: None, Частота: None

**Задача 9.2: Середньодобова температура**

Набір контрольних прикладів:

1) Вхід: [20, 22, 21, 23, 20, 19]

Очікуваний результат: Середня температура Цельсія: 21.5, Середня температура Фаренгейта: 70.7

2) Вхід: [10, 12, 13, 14, 15, 16]

Очікуваний результат: Середня температура Цельсія: 13.33, Середня температура Фаренгейта: 55.99

3) Вхід: [5, 5, 5, 5, 5, 5]

Очікуваний результат: Середня температура Цельсія: 5.0, Середня температура Фаренгейта: 41.0

4) Вхід: [-5, 0, 5, 10, 15, 20]

Очікуваний результат: Середня температура Цельсія: 7.5, Середня температура Фаренгейта: 45.5

5) Вхід: [30, 32, 34, 35, 36, 37]

Очікуваний результат: Середня температура Цельсія: 34.67, Середня температура Фаренгейта: 94.4

**Задача 9.3: Кількість двійкових 0 або 1 у числі**

Набір контрольних прикладів:

1) Вхід: 5 (двійкове представлення: 101)

Очікуваний результат: Кількість одиниць: 2

2) Вхід: 8 (двійкове представлення: 1000)

Очікуваний результат: Кількість нулів: 3

3) Вхід: 31 (двійкове представлення: 11111)

Очікуваний результат: Кількість одиниць: 5

4) Вхід: 16 (двійкове представлення: 10000)

Очікуваний результат: Кількість нулів: 4

5) Вхід: 1023 (двійкове представлення: 1111111111)

Очікуваний результат: Кількість одиниць: 10

## Лістинг функції для визначення категорії торнадо за шкалою Фудзіти

```
string fujita_scale (double speed) {

    if (speed < 65)  // Перевірка, чи є швидкість меншою за мінімальну межу
        return "Мала швидкість для виникнення торнадо";

    if (speed <= 115) // Перевірка діапазону швидкості для F0 категорії
        return "F0 (38.9% виникнення)";

    if (speed <= 180) // Перевірка діапазону швидкості для F1 категорії
        return "F1 (35.6% виникнення)";

    if (speed <= 253) // Перевірка діапазону швидкості для F2 категорії
        return "F2 (19.4% виникнення)";

    if (speed <= 332) // Перевірка діапазону швидкості для F3 категорії
        return "F3 (4.9% виникнення)";

    if (speed <= 418) // Перевірка діапазону швидкості для F4 категорії
        return "F4 (1.1% виникнення)";

    if (speed <= 512) // Перевірка діапазону швидкості для F5 категорії
        return "F5 (менше 0.1% виникнення)";

    return "Такої швидкості вітру не існує"; // Повернення повідомлення про невалідне
число
}
```

## Лістинг функції для визначення середньодобової температури

```
void average_daily_temp () {
    // Обчислення середньодобової температури
    double temp[6];
    int hours[] = {0, 4, 8, 12, 16, 20}; // Часи вимірювання

    cout << "Введіть температуру в наступні години:\n";
    for (int counter = 0; counter < 6; counter++) { // Перебір значень з масиву
        cout << hours[counter] << ":00 -> "; // Додавання всіх значень температури
        cin >> temp[counter];
    }
```

```
    // Обчислення середньої температури
    double total_temp = 0;
    for (int counter = 0; counter < 6; counter++) {
        total_temp += temp[counter];
    }

    double average_celsius = total_temp / 6; // Знаходження середньодобової
температури

    // Конвертація в градуси Фаренгейта
    double average_fahrenheit = (average_celsius * 9.0 / 5.0) + 32.0;

    cout << "\nСередньодобова температура:" << endl;
    cout << "В градусах Цельсія: " << average_celsius << " °C" << endl;
    cout << "В градусах Фаренгейта: " << average_fahrenheit << " °F" << endl;
}
```

## Лістинг функції для визначення кількості бітів, які відповідають молодшому біту

```
int count_bits (unsigned short N) {
    if (N == 0) {
        cout << "Число дорівнює 0, бітів зі значенням 1 немає." << endl;
        return 0;
    }

    bool bit = N & 1; // Перевіряємо значення молодшого біта (1 або 0)
    int count = 0;

    while (N) {  // Поки число не стане нулем
        if ((N & 1) == bit) {
            count++;  // Підраховуємо біти, які відповідають молодшому біту
        }
        N >>= 1;  // Зсуваємо число праворуч на 1 біт
    }

    return count;
}
```

## Лістинг TestDriver

```
#include <iostream>
#include "ModulesMyronenko.h"
#include <windows.h>
#include <limits>
```

```cpp
using namespace std;



// Функція отримання валідної швидкості вітру
double get_valid_speed() {
    double speed;
    while (true) {
        cout << "\nВизначення категорії торнадо\nВведіть швидкість вітру (число більше 0): ";
        cin >> speed;

        if (cin.fail() || speed <= 0) {
            cout << "Помилка: введіть коректне додатнє число!" << endl;
            cin.clear(); // Очищуємо стан потоку
            cin.ignore(numeric_limits<streamsize>::max(), '\n'); // Видаляємо некоректне введення
        } else {
            return speed;
        }
    }
}

// Функція отримання валідного числа для бітової обробки
unsigned short get_valid_number() {
    unsigned short number;
    while (true) {
        cout << "\nВизначення побітових нулів і одиниць" << endl;
        cout << "Введіть ціле число (0 - 65535): ";
        cin >> number;

        if (cin.fail()) {
            cout << "Помилка: введене значення не є валідним! Введіть число від (0 до 65535)" << endl;
            cin.clear();
            cin.ignore(numeric_limits<streamsize>::max(), '\n');
        } else {
            return number;
        }
    }
}

int main()
{
```

```
    SetConsoleOutputCP(65001); // Переведення консолі в формат UTF-8
    SetConsoleCP(65001);

    Dev_inf(); //функція викликана зі статичної бібліотеки з інф. про розробника

    double speed = get_valid_speed();
    cout << "Категорія торнадо за швидкістю " << speed << " км/год = " <<
fujita_scale(speed) << endl;

    cout << "\nОбчислення середньої температури ";
    average_daily_temp();

    // Введення числа з валідацією
    unsigned short number = get_valid_number();
    int result = count_bits(number);
    cout << "Кількість бітів, які відповідають молодшому біту: " << result << endl;

    return 0;
}
```

## Запуск тестового драйвера

```
Ярослав Мироненко ©
Починаючий розробник
прагну до вдосконалення навичок
та створення ефективних рішень.

Визначення категорії торнадо
Введіть швидкість вітру (число більше 0): 234
Категорія торнадо за швидкістю 234 км/год = F2 (19.4% виникнення)

Обчислення середньої температури Введіть температуру в наступні
години:
0:00 -> 45
4:00 -> 34
8:00 -> 23
12:00 -> 12
16:00 -> 23
```

```
20:00 -> 45

Середньодобова температура:
В градусах Цельсія: 30.3333 °C
В градусах Фаренгейта: 86.6 °F

Визначення побітових нулів і одиниць
Введіть ціле число (0 - 65535): 23456
Кількість бітів, які відповідають молодшому біту: 8
```

**Аналіз і постановка задачі 9.4**

Задача 9.4 передбачає розробку програмного забезпечення, яке реагує на введені користувачем символи. У відповідь на символи "z", "r", "s" та "x" викликаються відповідні функції, пов’язані з іншими задачами. Якщо користувач вводить невірний символ, виводиться звуковий сигнал помилки. Програма дозволяє продовжити роботу або завершити виконання за введенням "q" або "Q".

**Аналіз вимог до програмного забезпечення**

**Функціональні вимоги:**

Реагування на введені символи:

"z" → виклик функції calculation().

"r" → виклик функції задачі 9.1.

"s" → виклик функції задачі 9.2.

"x" → виклик функції задачі 9.3.

Інші символи → звуковий сигнал помилки.

Повторення запиту після кожного введення.

Завершення роботи програми після введення "q" або "Q".

**Лістинг програми Myronenko_task**

```
#include <iostream>
#include <windows.h>
#include "ModulesMyronenko.h"
```

```cpp
#include <string>
#include <limits>

using namespace std;


double get_valid_speed() { // Функція отримання валідної швидкості вітру
    double speed;
    while (true) {
        cout << "\nВизначення категорії торнадо\nВведіть швидкість вітру (число більше 0): ";
        cin >> speed;

        if (cin.fail() || speed <= 0) {
            cout << "Помилка: введіть коректне додатнє число!" << endl;
            cin.clear(); // Очищуємо стан потоку
            cin.ignore(numeric_limits<streamsize>::max(), '\n'); // Видаляємо некоректне введення
        } else {
            return speed;
        }
    }
}


unsigned short get_valid_number() { // Функція отримання валідного числа для бітової обробки
    unsigned short number;
    while (true) {
        cout << "\nВизначення побітових нулів і одиниць" << endl;
        cout << "Введіть ціле число (0 - 65535): ";
        cin >> number;

        if (cin.fail()) {
            cout << "Помилка: введене значення не є валідним! Введіть число від (0 до 65535)" << endl;
            cin.clear();
            cin.ignore(numeric_limits<streamsize>::max(), '\n');
        } else {
            return number;
        }
    }
}
```

```
char get_valid_choice() { // Функція для отримання валідного символу
    char choice;
    while (true) {
        cout << "\nМеню програми:" << endl;
        cout << "z - Виклик s_calculation()" << endl;
        cout << "r - Виклик функції fujita_scale(double speed) задачі 9.1" << endl;
        cout << "s - Виклик функції average_daily_temp() задачі 9.2" << endl;
        cout << "t - Виклик функції count_bits(unsigned short N) задачі 9.3" << endl;
        cout << "q - Вихід" << endl;
        cout << "Введіть символ: ";
        cin >> choice;

        cin.ignore(numeric_limits<streamsize>::max(), '\n'); // Очищення залишкових
символів у потоці

        // Перевірка на коректність вводу
        if (choice == 'z' || choice == 'r' || choice == 's' || choice == 't' ||
choice == 'q' || choice == 'Q') {
            return choice; // Якщо символ правильний, повертаємо його
        } else {
            cout << "\aНевірний символ! Спробуйте ще раз." << endl; // Звуковий
сигнал
            cin.clear(); // Очищаємо потік для уникнення помилок вводу
            cin.ignore(numeric_limits<streamsize>::max(), '\n'); // Ігноруємо
залишкові символи в потоці
        }
    }
}

int main(){

    SetConsoleOutputCP(65001); // Встановлення UTF-8 для консолі
    SetConsoleCP(65001);

    Dev_inf(); // Виклик функції з бібліотеки про розробника

    while (true) {
    char choice = get_valid_choice(); // Отримуємо валідний символ

    switch (choice) {
            case 'z': {
                cout << "Виклик функції s_calculation() задачі 9.1" << endl;
                double x, y, z;
                cout << "Введіть значення x, y, z: ";
                cin >> x >> y >> z;
```

```
                double result = s_calculation(x, y, z);
                cout << "Результат: " << result << endl;
                break;
            }
            case 'r': {
                cout << "Виклик функції fujita_scale(double speed) задачі 9.1" <<
endl;
                double speed = get_valid_speed();
                cout << "Категорія торнадо за швидкістю " << speed << " км/год = " <<
fujita_scale(speed) << endl;
                break;
            }
            case 's': {
                cout << "Виклик функції average_daily_temp() задачі 9.2" << endl;
                average_daily_temp(); // Функція з бібліотеки
                break;
            }
            case 't': {
                cout << "Виклик функції count_bits(unsigned short N) задачі 9.3" <<
endl;
                unsigned short number = get_valid_number();
                int result = count_bits(number);
                cout << "Кількість бітів, які відповідають молодшому біту: " <<
result << endl;
                break;
            }
            case 'q':
            case 'Q':
                cout << "Вихід з програми..." << endl;
                return 0;
            default:
                break;
        }
    }

    return 0;
}
```

## Тестовий запуск програми

```
Ярослав Мироненко ©
Починаючий розробник
прагну до вдосконалення навичок
та створення ефективних рішень.

Меню програми:
z - Виклик s_calculation()
```

```
r - Виклик функції fujita_scale(double speed) задачі 9.1
s - Виклик функції average_daily_temp() задачі 9.2
t - Виклик функції count_bits(unsigned short N) задачі 9.3
q - Вихід
Введіть символ: z
Виклик функції s_calculation() задачі 9.1
Введіть значення x, y, z: 4 5 2
Результат: 14.553

Меню програми:
z - Виклик s_calculation()
r - Виклик функції fujita_scale(double speed) задачі 9.1
s - Виклик функції average_daily_temp() задачі 9.2
t - Виклик функції count_bits(unsigned short N) задачі 9.3
q - Вихід
Введіть символ: r
Виклик функції fujita_scale(double speed) задачі 9.1

Визначення категорії торнадо
Введіть швидкість вітру (число більше 0): 234
Категорія торнадо за швидкістю 234 км/год = F2 (19.4% виникнення)

Меню програми:
z - Виклик s_calculation()
r - Виклик функції fujita_scale(double speed) задачі 9.1
s - Виклик функції average_daily_temp() задачі 9.2
t - Виклик функції count_bits(unsigned short N) задачі 9.3
q - Вихід
Введіть символ: s
Виклик функції average_daily_temp() задачі 9.2
Введіть температуру в наступні години:
0:00 -> 5
4:00 -> 6
8:00 -> 7
12:00 -> 8
16:00 -> 9
20:00 -> 8

Середньодобова температура:
В градусах Цельсія: 7.16667 °C
В градусах Фаренгейта: 44.9 °F

Меню програми:
z - Виклик s_calculation()
r - Виклик функції fujita_scale(double speed) задачі 9.1
```

```
s - Виклик функції average_daily_temp() задачі 9.2
t - Виклик функції count_bits(unsigned short N) задачі 9.3
q - Вихід
Введіть символ: t
Виклик функції count_bits(unsigned short N) задачі 9.3

Визначення побітових нулів і одиниць
Введіть ціле число (0 - 65535): 56788
Кількість бітів, які відповідають молодшому біту: 6

Меню програми:
z - Виклик s_calculation()
r - Виклик функції fujita_scale(double speed) задачі 9.1
s - Виклик функції average_daily_temp() задачі 9.2
t - Виклик функції count_bits(unsigned short N) задачі 9.3
q - Вихід
Введіть символ: q
Вихід з програми...
```

**Висновки:** У ході виконання лабораторної роботи було проведено детальний аналіз алгоритмів та їх реалізацію в середовищі програмування C++. Основна увага була зосереджена на розробці, налагодженні та тестуванні функцій, що виконують обчислення та обробку введених користувачем даних. Важливою частиною роботи стало впровадження механізмів перевірки коректності введених значень, що дозволило уникнути некоректного функціонування програми при помилковому вводі.

Завдяки виконанню лабораторної роботи було отримано практичні навички роботи з мовою програмування C++, зокрема з операторами введення-виведення, умовними конструкціями, циклами та функціями. Було досліджено принципи побудови програм із використанням модульного підходу, що дозволило оптимізувати код та покращити його читабельність. Крім того, реалізовано перевірку коректності введених даних, що значно підвищило надійність програми та її стійкість до некоректного введення користувачем.

Особливу увагу приділено роботі з потоками введення/виведення та обробці помилок. Було виявлено, що без очищення вхідного буфера можливе виникнення

проблеми нескінченного циклу при некоректному вводі. Для вирішення цієї проблеми використано методи cin.clear() та cin.ignore(), що дозволили уникнути зациклення та забезпечили коректну обробку помилкових введень користувачем.

У процесі виконання роботи вдалося досягти значного прогресу у розумінні принципів структурного програмування та організації коду. Реалізація меню з можливістю вибору дій дозволила зробити програму більш зручною для використання. Було впроваджено додаткову функціональність, зокрема перевірку валідності введених даних перед передачею їх у функції обчислення.

Загалом лабораторна робота була корисною, оскільки дозволила не тільки засвоїти основні концепції програмування, а й розвинути навички аналізу та усунення помилок. Виникали певні труднощі з налагодженням програми та виправленням помилок, проте завдяки самостійному пошуку інформації та аналізу роботи коду вдалося досягти бажаного результату.

У майбутніх роботах варто приділити більше уваги структуризації коду, впровадженню об’єктно-орієнтованого підходу та більш глибокому тестуванню функцій. Також можна покращити інтерфейс взаємодії з користувачем, зробивши його ще більш інтуїтивним та зручним.

У ході виконання лабораторної роботи було здійснено детальний аналіз поставлених завдань, розроблено програмне забезпечення відповідно до вимог, а також проведено його тестування. Основною метою роботи було навчитися використовувати механізми виклику функцій із бібліотек, обробляти введені користувачем дані, реалізовувати перевірку введення та забезпечувати коректне функціонування програми.

Одним із головних аспектів лабораторної роботи було проектування архітектури програми, що передбачало розподіл функціональності між основною програмою та статичною бібліотекою. Це дозволило створити гнучке та зручне у використанні програмне рішення. У ході роботи було реалізовано функції для виконання

математичних обчислень, аналізу погодних умов, а також обробки числових даних на рівні бітових операцій.

Окремо варто відзначити процес обробки введених користувачем даних. Була реалізована перевірка коректності введення, що дозволяє уникнути небажаних помилок і забезпечити стабільність програми. Наприклад, введення некоректних символів не призводить до аварійного завершення програми, а викликає відповідне повідомлення про помилку з проханням повторити введення.

Під час виконання лабораторної роботи виникали певні труднощі, серед яких:

Налаштування взаємодії між основною програмою та статичною бібліотекою.

Обробка некоректного вводу та забезпечення коректної роботи програми без зациклення.

Валідація введених користувачем значень у межах дозволених діапазонів.

Ці труднощі були успішно подолані шляхом глибшого аналізу вимог, тестування різних підходів та внесення відповідних коригувань у код.

Загалом, виконання лабораторної роботи дозволило закріпити навички роботи з функціями, структурою програмного забезпечення та обробкою введених користувачем даних. Також було розглянуто основи тестування програмного забезпечення шляхом перевірки коректності роботи окремих функцій та всієї програми в цілому.

**Особисті враження та рекомендації**

Лабораторна робота виявилася цікавою та корисною, оскільки дозволила застосувати на практиці отримані знання з програмування та проектування програмного забезпечення. Виконання завдань допомогло краще зрозуміти принципи побудови програмної архітектури, використання бібліотек та обробки введення користувача.

Серед можливих покращень даної лабораторної роботи можна запропонувати:

Додавання графічного інтерфейсу, що покращило б взаємодію з користувачем та зробило програму більш інтуїтивно зрозумілою.

Розширення можливостей тестування, наприклад, через автоматичне тестування введених значень у межах дозволених діапазонів.

Використання більш структурованих підходів до організації коду, наприклад, впровадження об'єктно-орієнтованого програмування для підвищення гнучкості та масштабованості програми.

Загалом, лабораторна робота сприяла підвищенню рівня знань та вдосконаленню навичок програмування, а також розширила уявлення про процес розробки програмного забезпечення. Отримані знання та досвід будуть корисними для подальшого навчання та професійного розвитку в галузі програмування.