МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Дніпровський національний університет

імені Олеся Гончара

Факультет прикладної математики

Кафедра комп'ютерних технологій

**ЗВІТ**

**З ЛАБОРАТОРНОЇ РОБОТИ № 1**

«Інфраструктура розробки проектів (MS Visual Studio + git + повторення 1 курсу***»***

з дисципліни «Об’єктно-орієнтоване програмування»

Виконав:

Студент групи ПА-22-2

Мирошниченко Олександр Ілліч

м. Дніпро

2024 р.

**Зміст**

Оглавление

[Постановка задачі 3](#_Toc175680695)

[Опис розв’язку 4](#_Toc175680696)

[Вихідний текст програми розв’язку задачі 6](#_Toc175680697)

[Опис інтерфейсу програми (керівництво користувача) 16](#_Toc175680698)

[Висновок 19](#_Toc175680699)

# Постановка задачі

1. Створення проекту "Hello, git!"на мові C++ по наданому шаблону

інфраструктури з прив’язкою до github.

2. Настроювання інфраструктури (IDE + git). Робота з проектами. Збереження

проекту в іншу папку. Перенос проекту з комп’ютера на комп’ютер. Зміна

параметрів IDE (кольори, клавіатурні комбінації, тощо). Зміна параметрів

проекту (ключі компілятора, параметри командного рядка, тощо). Створення

гілки (branch), подання ПР (pull request). Результати відображені на github, а

також у вигляді скріншотів у звіті, що відображують специфічне оточення

(підказка командного рядка, поточна папка, кольори IDE, тощо).

3. Багатомодульні програми.

Скласти програму, яка викликає функції, визначені в інших (окремо

транслюємих) файлах. Вміти відкривати/закривати проект, вміти додавати

новий файл до існуючого проекту. Вміти передавати і обробляти параметри

функцій (зокрема функції main) і повертати результати роботи функцій

# Опис розв’язку

1. Загальний опис програми

Ця програма дозволяє користувачеві вводити послідовності чисел, сортувати їх за певним критерієм, класифікувати на "хороші", "погані" та "нейтральні", а також зберігати результати до файлу. Програма підтримує введення даних як з командного рядка, так і з файлу.

2. Основні компоненти програми

Програма складається з трьох основних файлів:

Main.cpp: Головний файл, що керує роботою програми, взаємодіє з користувачем та викликає відповідні функції для обробки послідовностей.

Sequence.cpp: Реалізує методи класу Sequence, що включають додавання чисел, сортування, класифікацію та виведення результатів до файлу.

Sequence.h: Заголовковий файл, що містить оголошення класу Sequence та його методів.

3. Детальний опис роботи програми

Main.cpp:

Обробка командного рядка: Якщо користувач надає числа через командний рядок, програма додає їх до послідовності та зберігає ім'я файлу для виведення результатів. Якщо ж командний рядок не містить чисел, програма просить ввести ім'я вхідного файлу, з якого будуть зчитані дані.

Сортування послідовності: Після введення чисел, послідовність сортується за критерієм дробової частини числа. Функція fractionalPart обчислює дробову частину числа, яка використовується як критерій сортування.

Класифікація: Послідовність класифікується на три категорії: "хороші" (числа більше 10), "нейтральні" (числа від 0 до 10) та "погані" (числа менше або дорівнюють 0).

Збереження результатів: Програма зберігає відсортовану послідовність та результати класифікації у вказаний користувачем файл.

Sequence.cpp:

add(double number): Додає число до послідовності. Якщо кількість елементів досягла максимуму (100), виводиться повідомлення про переповнення.

sort(std::function<double(double)> criteria): Сортує послідовність за заданим критерієм. У цьому випадку критерій — це дробова частина числа, яка визначає порядок сортування.

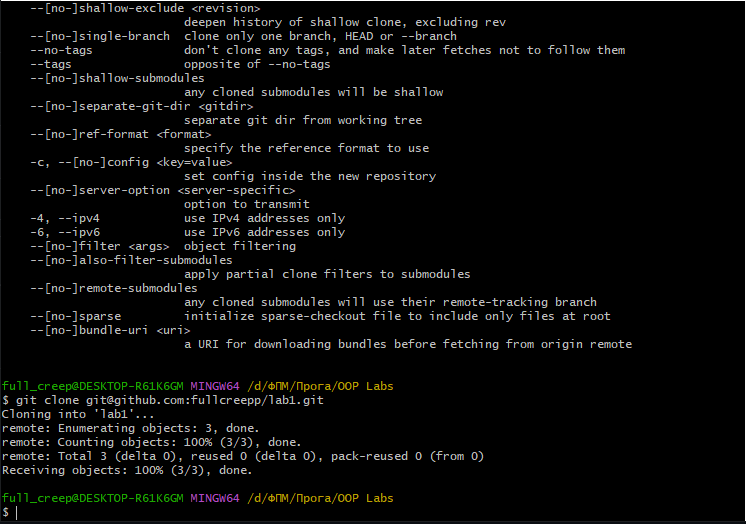
classify(int& good, int& bad, int& neutral): Класифікує елементи послідовності на "хороші", "нейтральні" та "погані" відповідно до заданих умов.

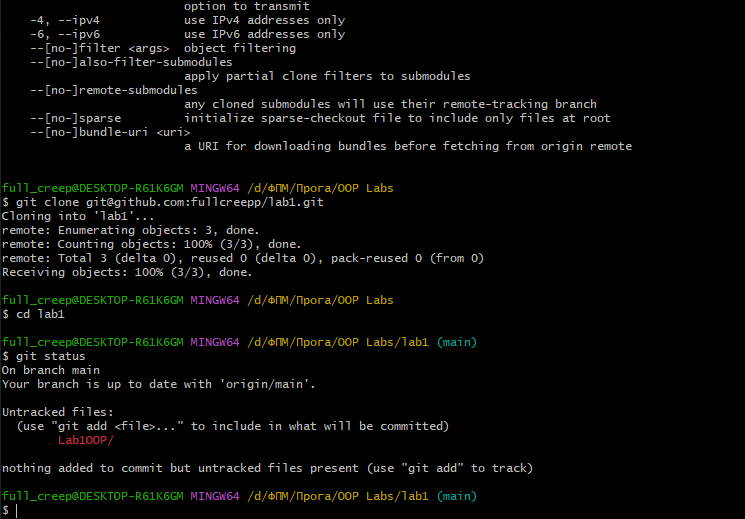
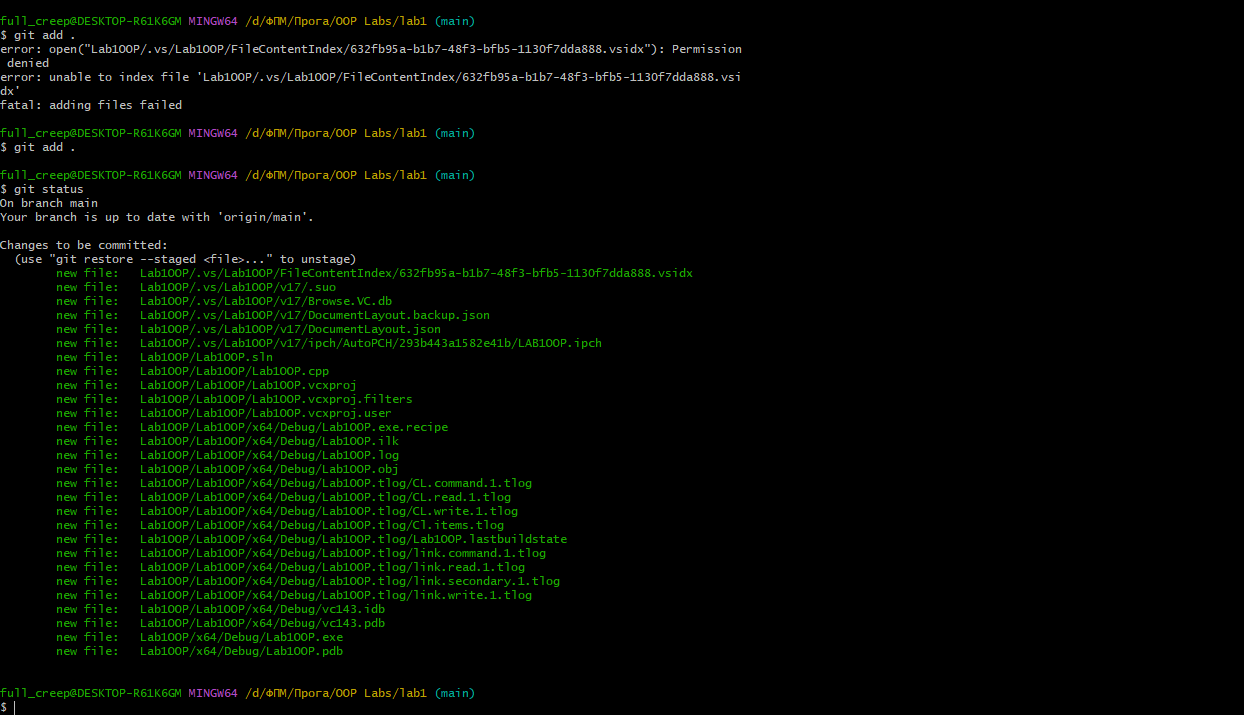
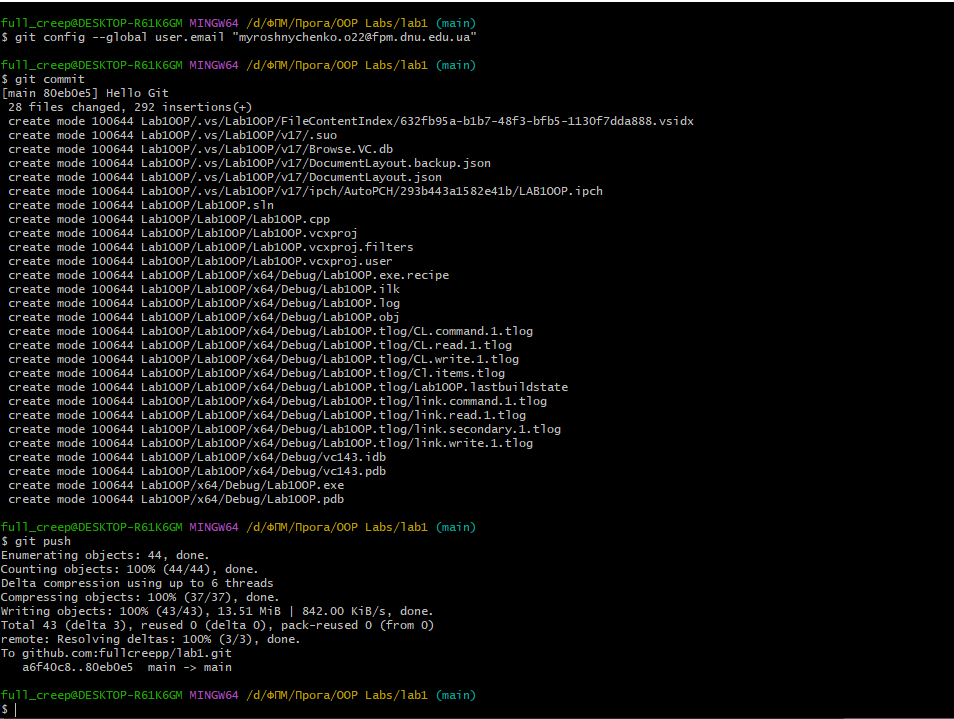
outputToFile(const std::string& filename): Зберігає відсортовану послідовність до вказаного файлу.

outputClassificationToFile(const std::string& filename, int good, int bad, int neutral): Додає до файлу результати класифікації.

Sequence.h:

Описує структуру класу Sequence, його поля та методи, включаючи конструктор для ініціалізації об'єкта послідовності.

Налаштування Git:  




# Вихідний текст програми розв’язку задачі

main.cpp:

#include <iostream>

#include <fstream>

#include <cmath>

#include <functional>

#include "Sequence.h"

#include <sstream>

// Criterion: Fractional part of the number

double fractionalPart(double number) {

return std::fabs(number - static\_cast<long long>(number));

}

int main(int argc, char\* argv[]) {

Sequence sequence;

std::string outputFileName;

// Handling command-line arguments

if (argc > 1) {

for (int i = 1; i < argc - 1; ++i) {

sequence.add(std::stod(argv[i]));

}

outputFileName = argv[argc - 1];

}

else {

// If there are no command-line arguments, ask for the input file name

std::string inputFileName;

std::cout << "Enter the input file name: ";

std::cin >> inputFileName;

std::ifstream inputFile(inputFileName);

if (!inputFile) {

std::cerr << "Failed to open the input file!" << std::endl;

return 1;

}

double number;

while (inputFile >> number) {

sequence.add(number);

}

inputFile.close();

std::cout << "Enter the output file name: ";

std::cin >> outputFileName;

}

// Sorting the sequence based on the fractional part

sequence.sort(fractionalPart)

// Classifying the sequence

int good, bad, neutral;

sequence.classify(good, bad, neutral);

// Writing results to the output file

sequence.outputToFile(outputFileName);

sequence.outputClassificationToFile(outputFileName, good, bad, neutral);

std::cout << "Good: " << good << "\nBad: " << bad << "\nNeutral: " << neutral << std::endl;

return 0;

}

Sequence.cpp:

#include "Sequence.h"

void Sequence::add(double number)

{

if (size < 100)

{

data[size++] = number;

}

else

{

std::cerr << "Послідовність переповнена!" << std::endl;

}

}

void Sequence::sort(std::function<double(double)> criteria)

{

for (int i = 0; i < size - 1; ++i)

{

for (int j = 0; j < size - i - 1; ++j)

{

if (criteria(data[j]) < criteria(data[j + 1]))

{

std::swap(data[j], data[j + 1]);

}

}

}

}

void Sequence::classify(int &good, int &bad, int &neutral) const

{

good = bad = neutral = 0;

for (int i = 0; i < size; ++i)

{

if (data[i] > 10)

{ // Приклад: Вважати числа > 10 як "хороші"

good++;

}

else if (data[i] > 0 && data[i] <= 10)

{ // Приклад: Вважати числа між 0 і 10 як "нейтральні"

neutral++;

}

else if (data[i] <= 0)

{ // Приклад: Вважати числа <= 0 як "погані"

bad++;

}

}

}

void Sequence::outputToFile(const std::string &filename) const

{

std::ofstream outputFile(filename);

if (!outputFile)

{

std::cerr << "Не вдалося відкрити вихідний файл!" << std::endl;

return;

}

outputFile << "Відсортована послідовність: ";

for (int i = 0; i < size; ++i)

{

outputFile << data[i] << " ";

}

outputFile.close();

}

void Sequence::outputClassificationToFile(const std::string &filename, int good, int bad, int neutral) const

{

std::ofstream outputFile(filename, std::ios\_base::app);

if (!outputFile)

{

std::cerr << "Не вдалося відкрити вихідний файл!" << std::endl;

return;

}

outputFile << "\nХороші: " << good << "\nПогані: " << bad << "\nНейтральні: " << neutral << std::endl;

outputFile.close();

}  
Sequence.h:  
#ifndef SEQENCE\_H

#define SEQENCE\_H

#include <iostream>

#include <fstream>

#include <cmath>

#include <functional>

class Sequence

{

private:

private:

double data[100]; // Масив для зберігання чисел

int size;

public:

Sequence() : size(0) {}

void add(double number) ;

void sort(std::function<double(double)> criteria);

void classify(int& good, int& bad, int& neutral) const;

void outputToFile(const std::string& filename) const;

void outputClassificationToFile(const std::string& filename, int good, int bad, int neutral) const;

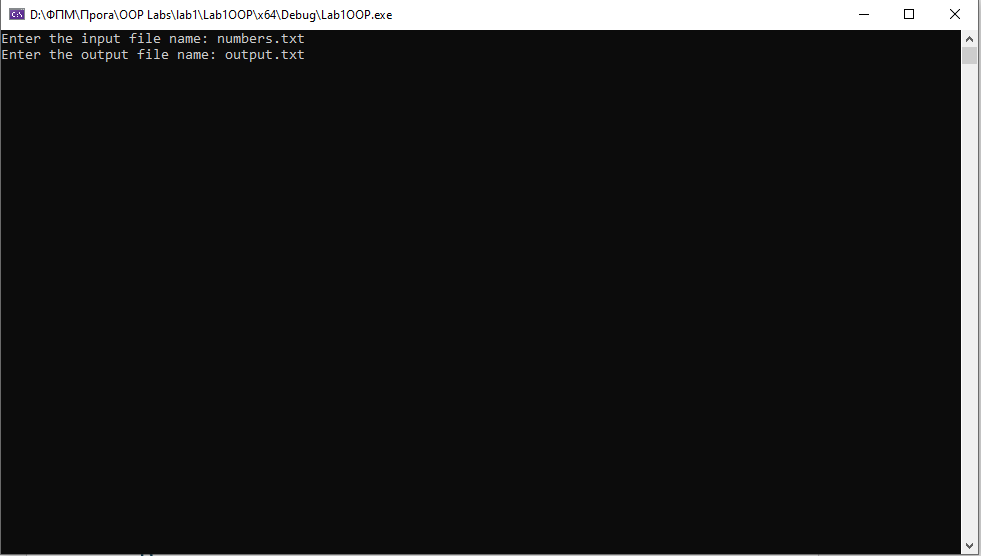
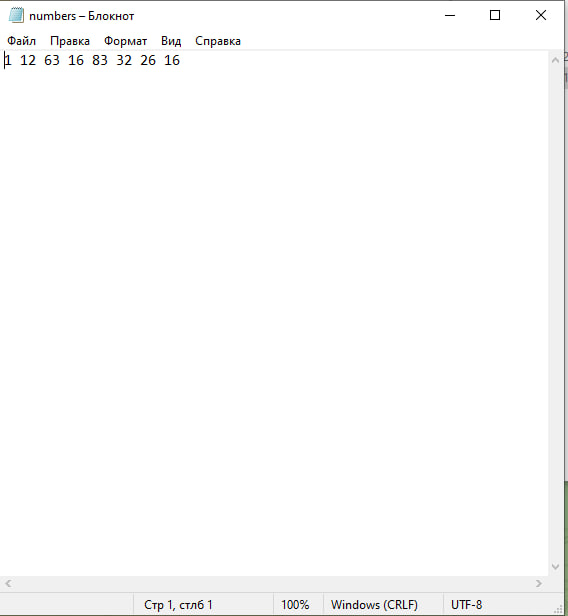
};

#endif // SEQENCE\_H

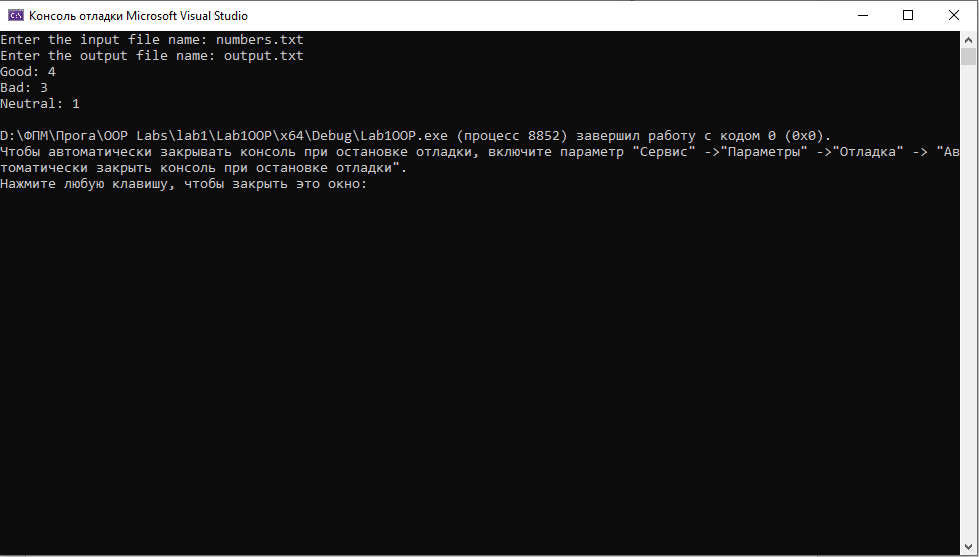
# Опис інтерфейсу програми (керівництво користувача)

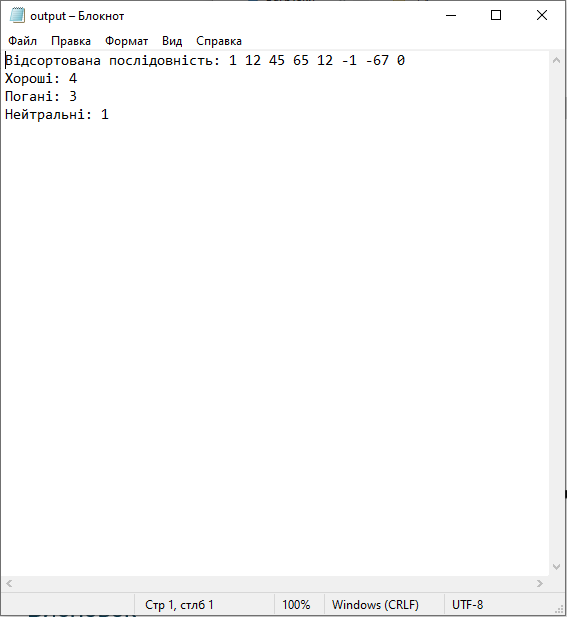
Приклад роботи з програмою:

1. Користувач вводить числа або через командний рядок, або через файл.



1. Програма сортує введені числа за дробовою частиною.
2. Програма класифікує числа як "хороші", "погані" та "нейтральні".



1. Результати зберігаються у файл та відображаються на екрані.

# Висновок

Виконуючи дану лабораторну роботу, я освоїв роботу з рядками, вказівниками, функціями та використанням стандартних бібліотек у мові програмування C++. В ході роботи було використано різноманітні концепції програмування, включаючи роботу з функціями-вказівниками, динамічне виділення пам'яті та використання функцій стандартної бібліотеки для обробки рядків і роботи з файлами.

Ця лабораторна робота дозволила поглибити розуміння основних концепцій програмування на мові C++ та навчитися вирішувати завдання з обробки текстових даних.

У процесі розробки програми також було використано три модулі:

1. Модуль "Sequence.h": в цьому модулі оголошено прототипи функцій, що використовуються в головній програмі. Це дозволяє ізолювати інтерфейс програми від її реалізації та полегшує роботу з кодом.
2. Модуль "Sequence.cpp": у цьому модулі реалізовано функції, оголошені у модулі " Sequence.h". Це дозволяє розділити логіку програми на окремі складові та забезпечує більшу модульність коду.
3. Головний модуль "main.cpp": це виконавчий файл, де викликаються функції та проводиться основна логіка програми.

Використання модулів дозволило мені більше зануритися у знання мови та реалізацію читабельності коду.