Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра систем штучного інтелекту



**Звіт**

про виконання

**Лабораторних та практичних робіт № 3(VNS Lab2. VNS Lab 3.VNS Lab 7. Class Practice Work. Self Practice Work)**

***з дисципліни:*** «Мови та парадигми програмування»

***з розділу***: «Епік 2: « Цикли. Вкладені цикли. Функції. Перевантаження функцій. Рекурсія »

***Виконав:***

***Бачик Анатолій Андрійович*** групи ШІ-13

### Тема роботи:

Цикли. Вкладені цикли. Функції. Перевантаження функцій. Рекурсія

### Мета роботи:

Ознайомлення з виконанням циклів,роботи функцій,перевантаженням програми,рекурсією.

### Теоретичні відомості:

1. Теоретичні відомості з переліком важливих тем:

* Тема №1: Цикли
* Тема №2: Функції
* Тема №3 Перевантаження функцій
* Тема №4: Рекурсія

1. Індивідуальний план опрацювання теорії:

**Тема №1 Цикли**

* + Джерела Інформації
    - <https://acode.com.ua/urok-70-tsykl-while/http://cpp.dp.ua/zminni/>
    - http://cpp.dp.ua/operatory-tsyklu/
  + Що опрацьовано:

Матеріал з джерел інформації

* + Статус: Ознайомлений
  + Початок опрацювання теми: 20.11.2023
  + Звершення опрацювання теми: 20.11.2023

**Тема №2 Функції**

* + Джерела Інформації:

<https://acode.com.ua/urok-15-funktsiyi-i-operator-return/>

http://cpp.dp.ua/ponyattya-funktsiyi/

* + Що опрацьовано:

Матеріал з джерел інформації

* + Статус: Ознайомлений
  + Початок опрацювання теми: 20.11.2023
  + Звершення опрацювання теми: 20.11.2023

**Тема №3 Перевантаження функцій**

* + Джерела Інформації

<https://acode.com.ua/urok-108-perevantazhennya-funktsij/>

https://www.bestprog.net/uk/2018/08/07/overloaded-functions-overloaded-functions-in-classes-overloading-of-class-constructors-access-to-the-overloaded-function-using-a-pointer-examples\_ua/

Що опрацьовано:

Матеріал з джерел інформації

* + Статус: Ознайомлений
  + Початок опрацювання теми: 20.11.2023
  + Звершення опрацювання теми: 20.11.2023

**Тема №4 Рекурсія**

* + Джерела Інформації

<http://cpp.dp.ua/rekursyvni-funktsiyi/>

https://www.bestprog.net/uk/2019/01/07/recursion-examples-of-tasks-solving-advantages-and-disadvantages-of-recursion-ua-2/

* + Що опрацьовано:

Матеріал з джерел інформації

* + Статус: Ознайомлений
  + Початок опрацювання теми: 20.11.2023
  + Звершення опрацювання теми: 20.11.2023

### Виконання роботи:

**Завдання №1**

1. *Опрацювання завдання та вимог до програм та середовища:*

**Завдання №1**

Тут вказуєте

-Варіант 25,VNS Lab 2 Task 1-1

- Знайти суму 7 членів ряду виразу

Запланований час для виконання завдання: 1 день

1. *Код програми*

#include <iostream>

#include <cmath>

using namespace std;

int main(){

int sum = 0; //Лічильник

for(int n = 1;n <= 7; n++){

sum += pow(n,2) \* exp(-sqrt(n));

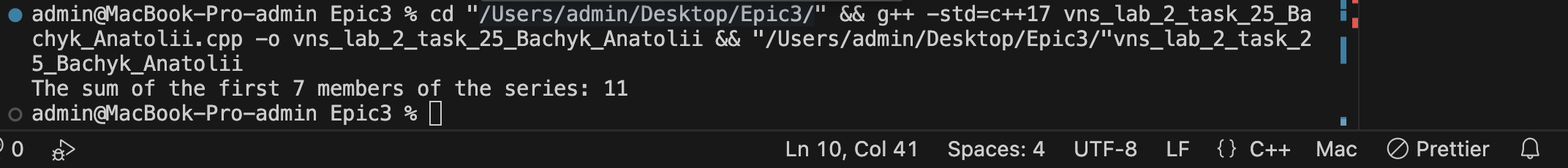
}

cout << "The sum of the first 7 members of the series: " << sum << endl;

return 0;

}

1. *Результати виконання завдань, тестування та фактично затрачений час:*

****

Потрачено часу: 15 хв

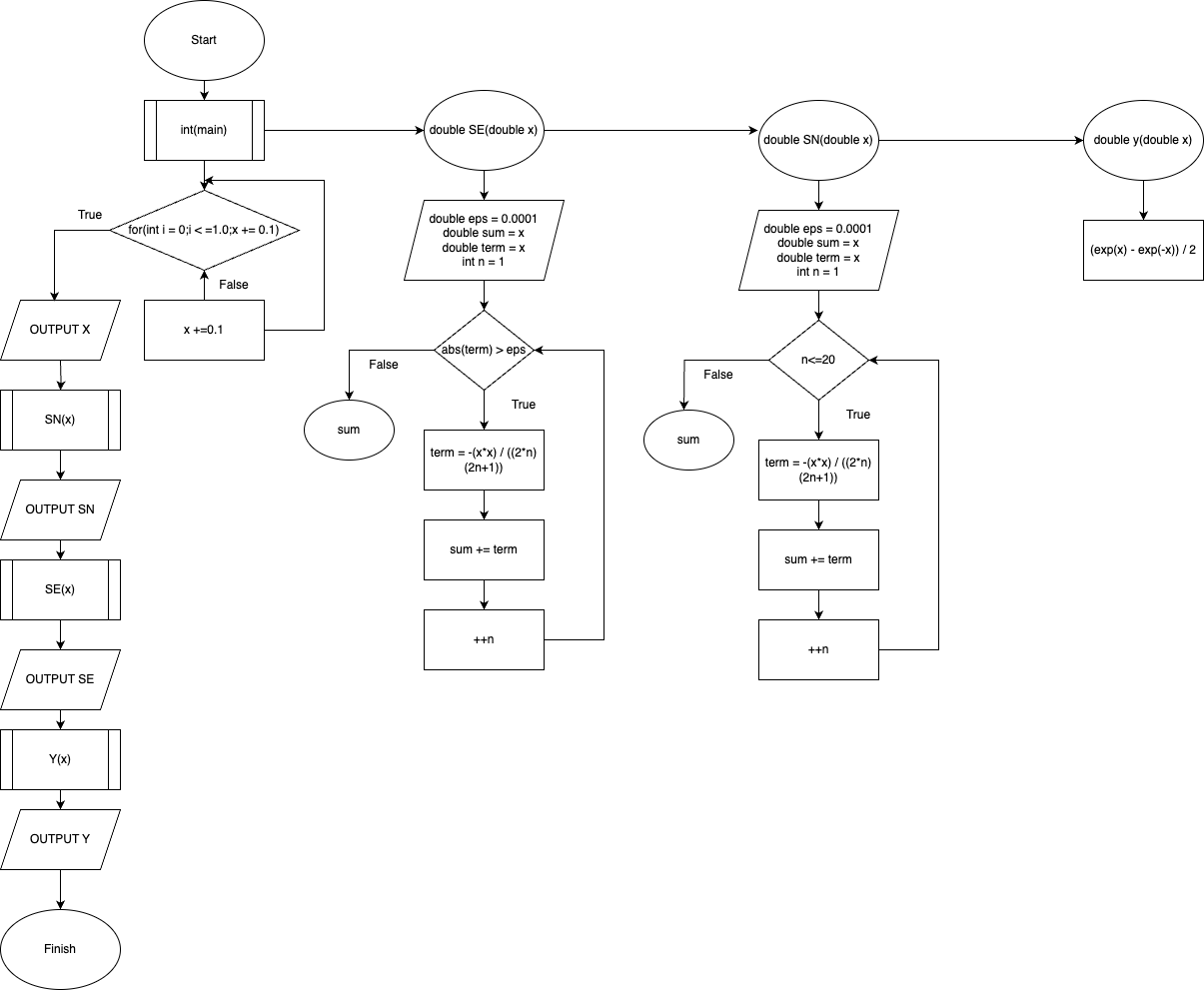
**Завдання №2**

1. *Опрацювання завдання та вимог до програм та середовища:*

-Варіант 25,VNS Lab 3 Task 1

- Обчислити функцію f(x), використовуючи її розклад в степеневий ряд

1. *Дизайн та планована оцінка часу виконання завдань:*

**

1. *Код програми*

#include <iostream>

#include <cmath>

using namespace std;

double SN(double x) {

double sum = x;

double term = x;

for (int n = 1; n <= 20; ++n) {

term \*= -(x \* x) / ((2 \* n) \* (2 \* n + 1));

sum += term;

}

return sum;

}

double SE(double x) {

double eps = 0.0001;

double sum = x;

double term = x;

int n = 1;

while (abs(term) > eps) {

term \*= -(x \* x) / ((2 \* n) \* (2 \* n + 1));

sum += term;

++n;

}

return sum;

}

double y(double x) {

return (exp(x) - exp(-x)) / 2;

}

int main() {

for (double x = 0.1; x <= 1.0; x += 0.1) {

cout << "x = " << x;

cout << " sn = " << SN(x);

cout << " se = " << SE(x);

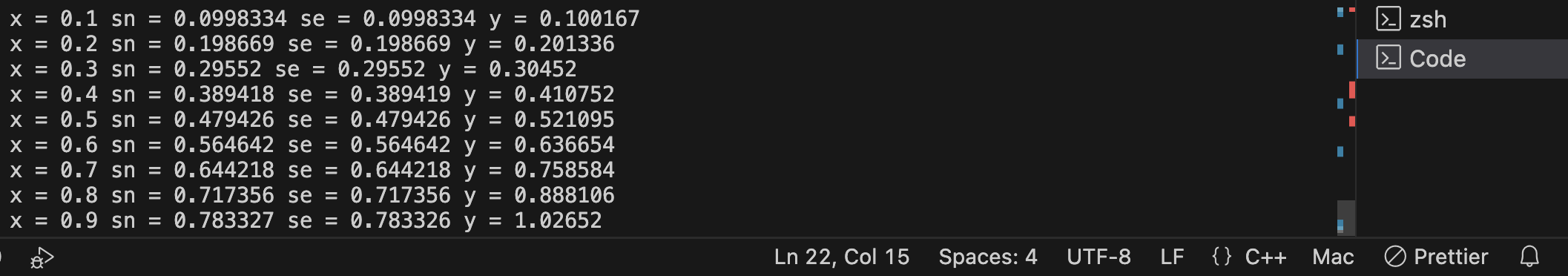
cout << " y = " << y(x) << endl;

}

return 0;

}

1. *Результати виконання завдань, тестування та фактично затрачений час:*



Потрачено часу: 1 день

**Завдання №3**

1. *Опрацювання завдання та вимог до програм та середовища:*

-Варіант 25,VNS Lab 7 Task 1-1

- Написати функцію (або макровизначення), що знаходить довжину сторони по координатах його точок.. Написати функцію square, що обчислює площу

трикутника, заданого координатами вершин. Написати функцію square1 з

змінною кількістю параметрів, що визначає площу трикутника, що містить

діагональ найбільшої довжини опуклого багатокутника, заданого

координатами своїх вершин.

1. *Код програми*

#include <iostream>

#include <cmath>

#include <vector>

#include <algorithm>

using namespace std;

struct Point {

double x, y;

};

double length(double x1, double y1, double x2, double y2) {

return sqrt(pow(x1 - x2, 2) + pow(y1 - y2, 2));

}

double square(double x1, double y1, double x2, double y2, double x3, double y3) {

double a = length(x1, y1, x2, y2);

double b = length(x2, y2, x3, y3);

double c = length(x3, y3, x1, y1);

double p = (a + b + c) / 2;

return sqrt(p \* (p - a) \* (p - b) \* (p - c));

}

double square1(vector<Point> points) {

int n = points.size();

double max\_square = 0;

for (int i = 0; i < n; ++i) {

for (int j = i + 1; j < n; ++j) {

for (int k = j + 1; k < n; ++k) {

double s = square(points[i].x, points[i].y, points[j].x, points[j].y, points[k].x, points[k].y);

max\_square = max(max\_square, s);

}

}

}

return max\_square;

}

int main() {

double x1, y1, x2, y2, x3, y3;

cout << "Enter x1, y1: ";

cin >> x1 >> y1;

cout << "Enter x2, y2: ";

cin >> x2 >> y2;

cout << "Enter x3, y3: ";

cin >> x3 >> y3;

double result = square(x1, y1, x2, y2, x3, y3);

cout << "The area of the triangle is: " << result << endl;

vector<Point> points = {{x1, y1}, {x2, y2}, {x3, y3}};

double result1 = square1(points);

cout << "The area of the largest triangle is: " << result1 << endl;

return 0;

}

1. *Результати виконання завдань, тестування та фактично затрачений час:*

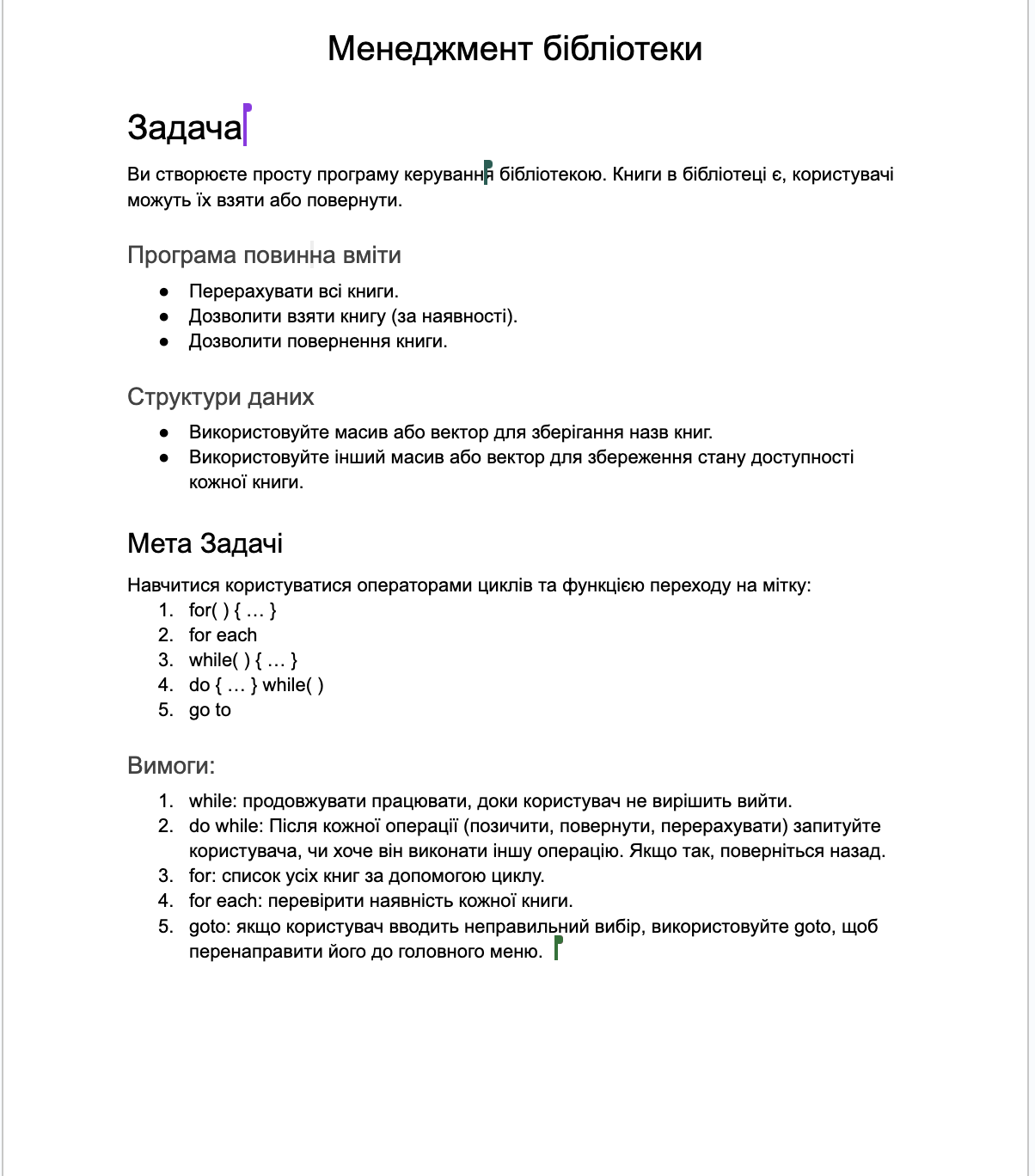


Потрачено часу: 1 день

**Завдання №4**

1. *Опрацювання завдання та вимог до програм та середовища:*

- Class Practice Work

- 

1. *Код програми*

#include <iostream>

#include <string>

#include <vector>

using namespace std;

int main() {

vector<string> books = {"In Search of Lost Time", "Ulysses", " The Great Gatsby"," War and Peace"};

vector<bool> available = {true, true, true, true};

int choice;

string bookName;

menu:

cout << "Library Management\n";

cout << "1. List all books\n";

cout << "2. Borrow a book\n";

cout << "3. Return a book\n";

cout << "4. Exit\n";

cout << "Enter your choice: ";

cin >> choice;

switch(choice) {

case 1:

for (int i = 0; i < books.size(); i++) {

cout << i + 1 << ". " << books[i] << (available[i] ? " (Available)\n": " (Borrowed)\n");

}

break;

case 2:

cin.ignore();

cout << "Enter book name to borrow: ";

getline(cin, bookName);

for (size\_t i = 0; i < books.size(); ++i) {

if (books[i] == bookName && available[i]) {

available[i] = false;

cout << "Book borrowed successfully!\n";

break;

}

}

break;

case 3:

cin.ignore();

cout << "Enter book name to return: ";

getline(cin, bookName);

for (size\_t i = 0; i < books.size(); ++i) {

if (books[i] == bookName && !available[i]) {

available[i] = true;

cout << "Book returned successfully\n";

break;

}

}

break;

case 4:

cout << "Goodbye!";

return 0;

default:

cout << "Invalid choice. Please try again.\n";

goto menu;

}

char anotherOperation;

do {

cout << "Do you want to perform another operation? (Y/N): ";

cin >> anotherOperation;

} while (anotherOperation != 'Y' && anotherOperation != 'N');

if (anotherOperation == 'Y') {

goto menu;

} else {

cout << "Goodbye!";

}

return 0;

}

1. *Результати виконання завдань, тестування та фактично затрачений час:*



Потрачено часу: 1 день

**Завдання №5**

1. *Опрацювання завдання та вимог до програм та середовища:*

-Self Practice Work

- У цьому завданні потрібно знайти максимальну кількість печива, яку може з'їсти Марічка, не привертаючи уваги Зеника. Зеник спостерігатиме за кількістю пачок печива, тому Марічка планує з'їсти саме ту кількість, щоб не викликати підозрів у Зеника. Для цього потрібно розглянути кількість пачок печива в рюкзаку Зеника та кількість штук в кожній пачці. Марічка повинна знайти спосіб з'їсти певну кількість штук так, щоб не спустошити жодну пачку до кінця, тобто не спричинити підозрів Зеника. Вона має з'їсти так, щоб в разі перевірки Зеником кількість пачок печива залишалась незмінною або не ставало менше, ніж має залишитись за розрахунками Зеника. Отже, Марічці потрібно розрахувати оптимальну кількість штук печива для споживання так, щоб вона задовольнила свою бажану кількість, але при цьому не зменшила кількість пачок до такого рівня, який викликав би підозру у Зеника.

1. *Код програми*

#include <iostream>

using namespace std;

int sum(int count, ...) {

int sum = 0;

int \*ptr = &count;

ptr++;// Переміщення вказівника на наступні аргументи

for (int i = 0; i < count; ++i) {

sum += \*ptr;

ptr++;

}

return sum;

}

int main() {

int result1 = sum(3, 10, 20, 30);

int result2 = sum(7, 5, 15, 25, 10, 20, 30, 40);

int result3 = sum(11, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22);

cout << "Сума з першого виклику: " << result1 << endl;

cout << "Сума з другого виклику: " << result2 << endl;

cout << "Сума з третього виклику: " << result3 << endl;

return 0;

}

1. *Результати виконання завдань, тестування та фактично затрачений час:*



Потрачено часу: *1 година*

**Завдання №5**

1. *Опрацювання завдання та вимог до програм та середовища:*

-Варіант 25,VNS Lab 7 Task 2

- Написати перевантажені функції й основну програму, що їх викликає.

а) для двовимірного масиву знищує всі парні рядки;

б) для одновимірного масиву знищує всі елементи, що містяться між двома

нульовими елементами.

1. *Код програми*

#include <iostream>

#include <vector>

using namespace std;

void process(vector<vector<int>>& arr) {

for (size\_t i = 0; i < arr.size(); i++) {

if (i % 2 == 0) {

arr.erase(arr.begin() + i);

i--; // Зменшуємо ітератор, оскільки видалили рядок

}

}

}

void process(vector<int>& numbs) {

bool foundFirstZero = false;

size\_t start = 0;

for (size\_t i = 0; i < numbs.size(); i++) {

if (numbs[i] == 0) {

if (!foundFirstZero) {

foundFirstZero = true;

start = i;

} else {

size\_t end = i;

numbs.erase(numbs.begin() + start + 1, numbs.begin() + end);

foundFirstZero = false;

i = start; // Починаємо знову з першого нуля

}

}

}

}

int main() {

vector<vector<int>> arr {{1, 2}, {1, 2}, {1, 3}};

vector<int> numbs {1, 0, 1, 2, 3, 4, 0, 5};

process(arr);

for (size\_t i = 0; i < arr.size(); i++) {

cout << "{";

for (size\_t j = 0; j < arr[i].size(); j++) {

cout << arr[i][j];

}

cout << "}";

}

cout << endl;

process(numbs);

for (size\_t i = 0; i < numbs.size(); i++) {

cout << numbs[i] << " ";

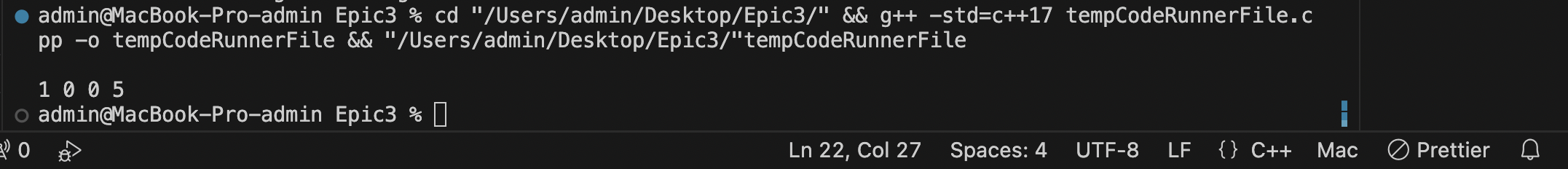
}

cout << endl;

return 0;

}

1. *Результати виконання завдань, тестування та фактично затрачений час:*



Потрачено часу: *2 дня*

### Висновки:

Озномлений з циклами,рекурсією,перевантаженням функції,рекурсією.Використання їх в поставлених задачах.