Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра систем штучного інтелекту

A blue and white logo

Description automatically generated

**Звіт**

**про виконання розрахунково-графічних робіт блоку № 7**

***з дисципліни:*** «Основи програмування»

до:

ВНС Розрахунково-графічних робіт № 1-4

Практичних Робіт до блоку № 7

**Виконав:**

Студент групи ШІ-12

Кривичко Назар

**Завдання 1:**

Розробити лінійний алгоритм для розв’язання

Задачі.

#define \_USE\_MATH\_DEFINES

#include <iostream>

#include <cmath>

int main(void)

{

int integerVar = 5; // Цілочисельна змінна

float realVar = 2.7f; // Дійсна змінна

double doubleVar = 3.14159; // Дійсна змінна з подвійною точністю

const int INTEGER\_CONST = 10; // Цілочисельна константа

double x = 1.45, y = -1.22, z = 3.5;

double b = (1 + std::pow(z, 2)) / (3 + (std::pow(z, 2) / 5));

double a;

// Використання умовного оператора для розгалуження

if (integerVar > INTEGER\_CONST)

{

a = (2 \* cos(x - (M\_PI / 6)) \* b) / (0.5 + std::pow(sin(y), 2));

}

else

{

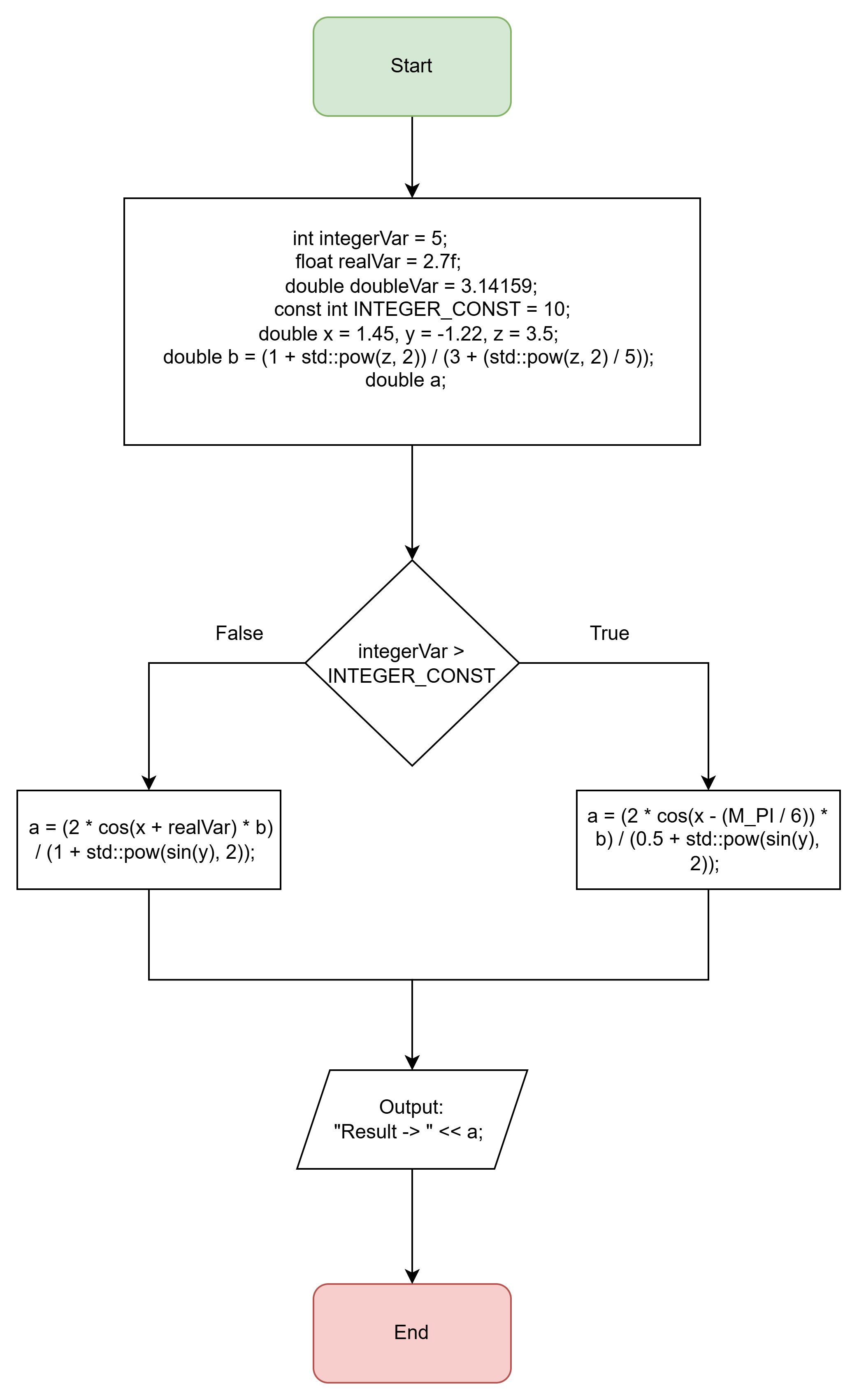
a = (2 \* cos(x + realVar) \* b) / (1 + std::pow(sin(y), 2));

}

std::cout << "Result -> " << a;

return 0;

}



**Завдання 2:**

**Розробити алгоритм, що розгалужується для**

**розв’язання задачі номер якої відповідає порядковому номеру**

**студента в журналі викладача**

**Обчислити функцію**

#include <iostream>

#include <cmath>

int main(void)

{

const int size1D = 10;

const int rows = 3;

const int cols = 3;

// Одновимірний масив для зберігання значень функції

double functionValues[size1D];

// Двовимірний масив для зберігання значень x та y

double xyValues[rows][cols] = { {0} };

int index = 0;

double x = -2.5;

// Використання циклу do-while для обчислення значень функції

do

{

double z = abs(x) + std::pow(x, 2);

double y = z \* ((sin(std::pow(2 + x, 2))) / (2 + x));

// Зберігаємо значення y у одновимірний масив

if (index < size1D)

{

functionValues[index] = y;

}

// Зберігаємо значення x та y у двовимірний масив (обмежено розміром rows x cols)

if (index < rows \* cols)

{

xyValues[index / cols][index % cols] = y;

}

std::cout << "Function value in [" << x << "] is -> " << y << std::endl;

x += 0.5;

index++;

} while (x <= 2);

// Використання циклу while для виведення значень з одновимірного масиву

int i = 0;

while (i < size1D)

{

std::cout << "Value from 1D array [" << i << "] -> " << functionValues[i] << std::endl;

i++;

}

// Використання циклу for для виведення значень з двовимірного масиву

for (int row = 0; row < rows; row++)

{

for (int col = 0; col < cols; col++)

{

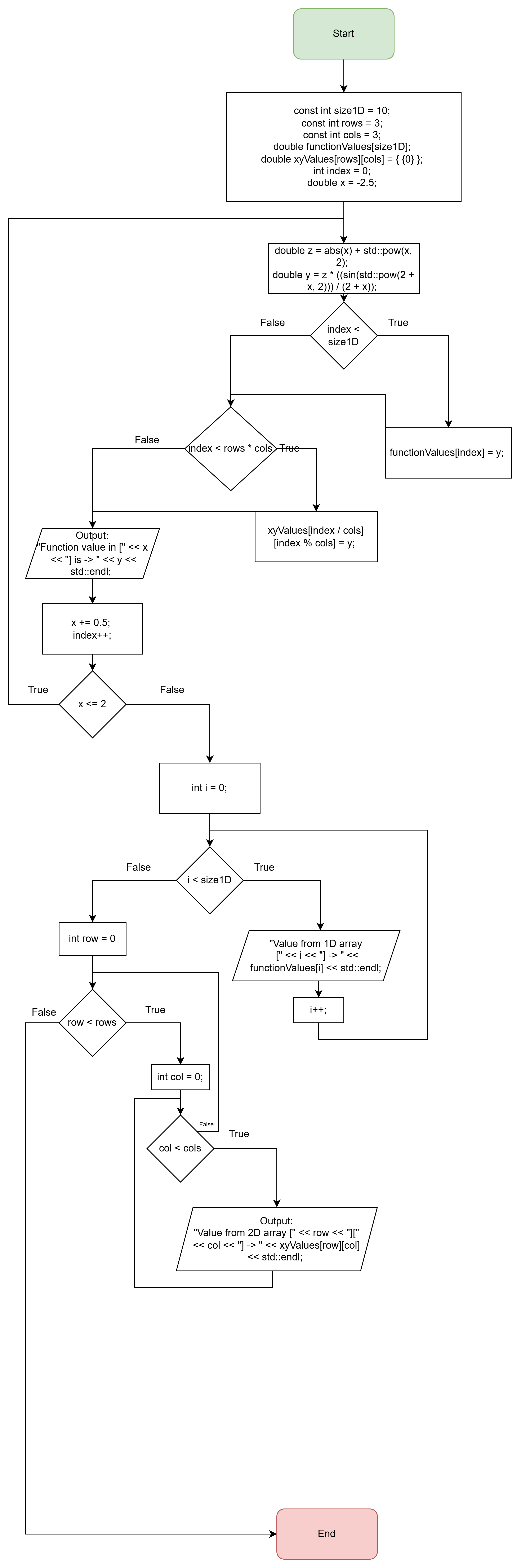
std::cout << "Value from 2D array [" << row << "][" << col << "] -> " << xyValues[row][col] << std::endl;

}

}

return 0;

}



**Завдання 3:**

**Обчислення об'єму циліндра. Нижче приведений**

**вид екрану під час виконання програми, що рекомендується**

**(дані, які вводяться користувачем, виділені напівжирним**

**шрифтом).**

**Обчислення об'єму циліндра.**

**Введіть початкові дані:**

**радіус підстави (см) > 5**

**висоту циліндра (см) > 10**

**Об'єм циліндра 1570.80 см. куб.**

**Для завершення натисніть клавішу <Enter>.**

**Обчислення опору електричний ланцюгу складається з трьох**

**послідовно сполучених резисторів.**

#define \_USE\_MATH\_DEFINES

#include <iostream>

#include <cmath>

#include <iomanip>

// Функція для обчислення об'єму циліндра (перевантаження функції)

double calculateVolume(double radius, double height) {

return M\_PI \* std::pow(radius, 2) \* height;

}

// Функція для обчислення послідовного опору трьох резисторів (перевантаження функції)

double calculateResistance(double R1, double R2, double R3) {

return R1 + R2 + R3;

}

// Рекурсивна функція для обчислення факторіалу числа (для демонстрації рекурсії)

int factorial(int n) {

if (n <= 1) return 1;

return n \* factorial(n - 1);

}

int main() {

std::cout << "Обчислення об'єму циліндра.\n";

std::cout << "Введіть початкові дані:\n";

double radius, height;

while (true) {

std::cout << "радіус підстави (см) > ";

std::cin >> radius;

if (radius < 0) {

std::cout << "Радіус не може бути від'ємним! Пропускаємо цю ітерацію.\n";

continue;

}

std::cout << "висоту циліндра (см) > ";

std::cin >> height;

if (height < 0) {

std::cout << "Висота не може бути від'ємною! Пропускаємо цю ітерацію.\n";

continue;

}

double volume = calculateVolume(radius, height);

std::cout << "Об'єм циліндра: " << std::fixed << std::setprecision(2) << volume << " см куб.\n";

break; // Використання оператора break для виходу з циклу після успішного введення

}

std::cout << "Для завершення натисніть клавішу <Enter>.\n";

std::cin.ignore();

std::cin.get();

std::cout << "\nОбчислення опору електричний ланцюгу, що складається з трьох\n";

std::cout << "послідовно сполучених резисторів.\n";

double R1, R2, R3;

std::cout << "Введіть опір першого резистора (Ом) > ";

std::cin >> R1;

std::cout << "Введіть опір другого резистора (Ом) > ";

std::cin >> R2;

std::cout << "Введіть опір третього резистора (Ом) > ";

std::cin >> R3;

double totalResistance = calculateResistance(R1, R2, R3);

std::cout << "Загальний опір ланцюгу: " << totalResistance << " Ом.\n";

int num;

std::cout << "\nВведіть число для обчислення факторіалу: ";

std::cin >> num;

if (num < 0) {

std::cout << "Факторіал не визначено для від'ємних чисел.\n";

} else {

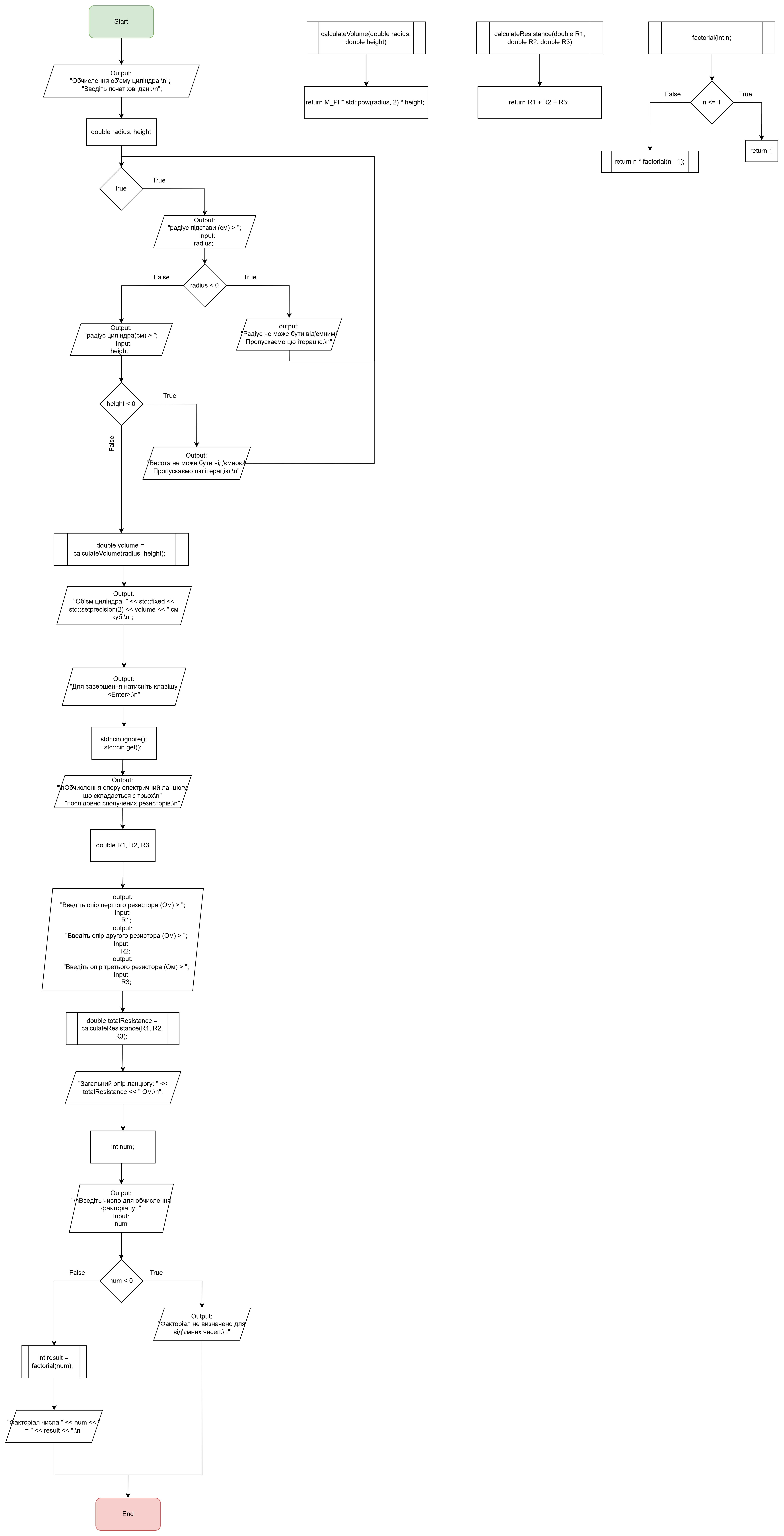
int result = factorial(num);

std::cout << "Факторіал числа " << num << " = " << result << ".\n";

}

return 0;

}



**Завдання 4:**

**Варіант 25. Скласти програму, що обчислює суму і середнє**

**арифметичне послідовності позитивних чисел**

#include <iostream>

#include <vector>

#include <fstream>

#include <cstdlib>

#include <ctime>

// Структура для зберігання векторів і їх середнього значення

struct SequenceData {

std::vector<int> values;

double average;

SequenceData(size\_t size) : values(size), average(0.0) {}

};

void fill\_random(std::vector<int>& v) {

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

v[i] = rand() % 10 + 1;

}

}

double calculate\_average(const std::vector<int>& v) {

int sum = 0;

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

sum += v[i];

}

return static\_cast<double>(sum) / v.size();

}

// Функція для запису даних у файл

void write\_to\_file(const char\* filename, SequenceData\* data) {

std::ofstream outFile(filename);

if (outFile.is\_open()) {

for (int value : data->values) {

outFile << value << ' ';

}

outFile << "\nAverage of sequence: " << data->average << std::endl;

outFile.close();

std::cout << "Data written to file: " << filename << std::endl;

} else {

std::cerr << "Error opening file for writing.\n";

}

}

// Функція для зчитування даних з файлу

void read\_from\_file(const char\* filename) {

std::ifstream inFile(filename);

if (inFile.is\_open()) {

std::cout << "Reading from file: " << filename << std::endl;

std::string line;

while (std::getline(inFile, line)) {

std::cout << line << std::endl;

}

inFile.close();

} else {

std::cerr << "Error opening file for reading.\n";

}

}

int main() {

srand(static\_cast<unsigned int>(time(0)));

SequenceData seq1(10), seq2(10), seq3(10);

fill\_random(seq1.values);

fill\_random(seq2.values);

fill\_random(seq3.values);

seq1.average = calculate\_average(seq1.values);

seq2.average = calculate\_average(seq2.values);

seq3.average = calculate\_average(seq3.values);

const char\* filename = "sequences.txt";

// Записуємо дані у файл за допомогою вказівників на структури

write\_to\_file(filename, &seq1);

write\_to\_file(filename, &seq2);

write\_to\_file(filename, &seq3);

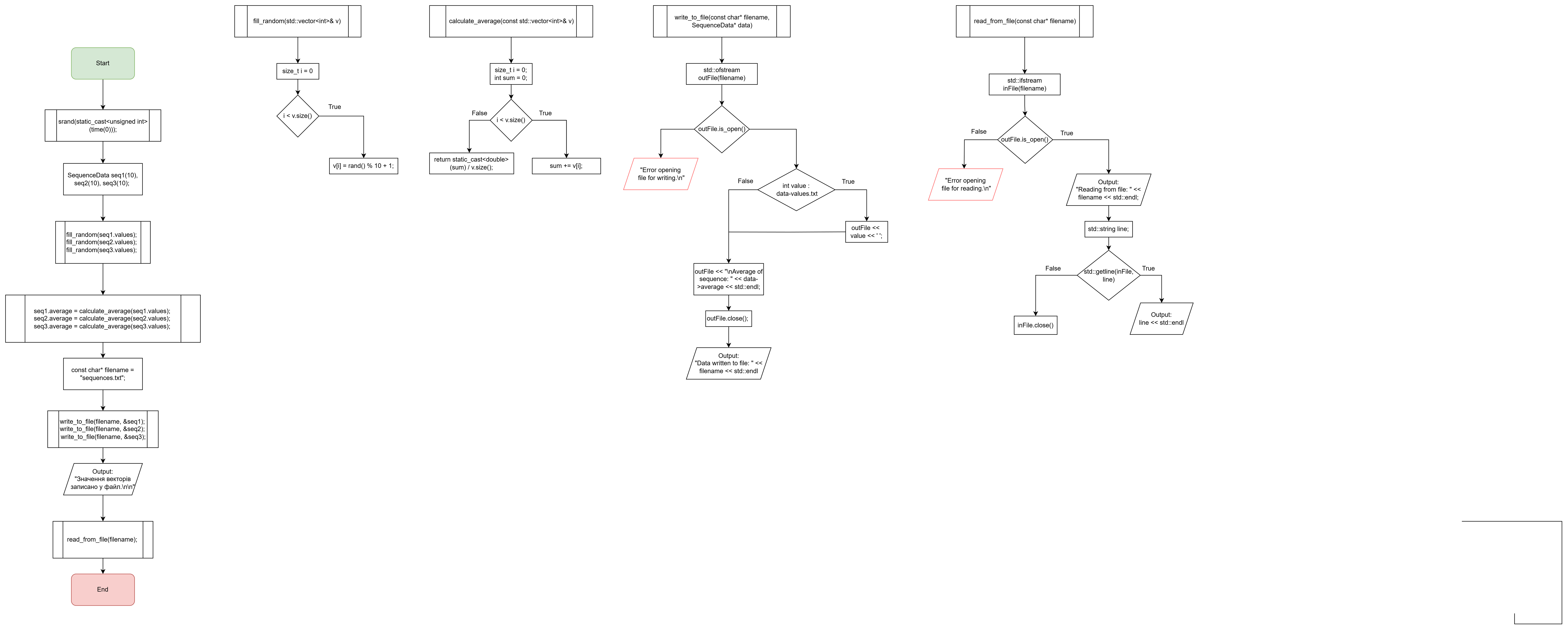
std::cout << "Значення векторів записано у файл.\n\n";

// Зчитуємо дані з файлу

read\_from\_file(filename);

return 0;

}



**Завдання з Algotester:**  
**1.**

#include <iostream>

#include <vector>

using namespace std;

struct Point {

int x, y;

};

int crossProduct(const Point &A, const Point &B, const Point &C) {

int x1 = B.x - A.x;

int y1 = B.y - A.y;

int x2 = C.x - B.x;

int y2 = C.y - B.y;

return x1 \* y2 - y1 \* x2;

}

int main() {

int n;

cin >> n;

vector<Point> points(n);

// Зчитуємо координати всіх точок маршруту

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

cin >> points[i].x >> points[i].y;

}

int leftTurns = 0, rightTurns = 0;

// Обчислюємо повороти між кожними трьома послідовними точками

for (int i = 0; i < n - 2; i++) {

int cp = crossProduct(points[i], points[i + 1], points[i + 2]);

if (cp > 0) {

leftTurns++;

} else if (cp < 0) {

rightTurns++;

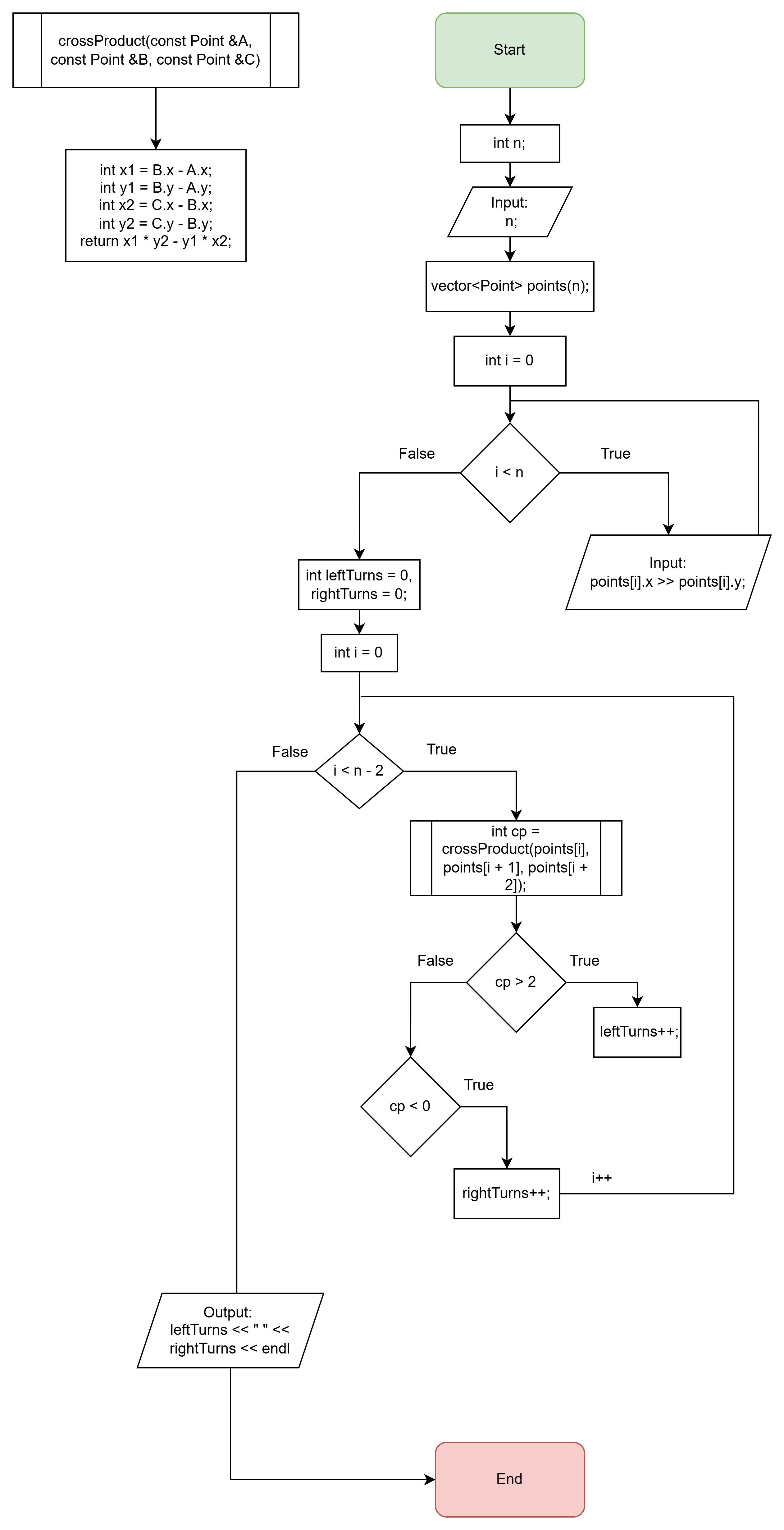
}

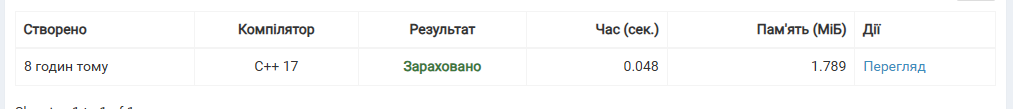
}

cout << leftTurns << " " << rightTurns << endl;

return 0;

}





**2.**

#include <iostream>

#include <cmath>

int main(void)

{

const int size1D = 10;

const int rows = 3;

const int cols = 3;

// Одновимірний масив для зберігання значень функції

double functionValues[size1D];

// Двовимірний масив для зберігання значень x та y

double xyValues[rows][cols] = { {0} };

int index = 0;

double x = -2.5;

// Використання циклу do-while для обчислення значень функції

do

{

double z = abs(x) + std::pow(x, 2);

double y = z \* ((sin(std::pow(2 + x, 2))) / (2 + x));

// Зберігаємо значення y у одновимірний масив

if (index < size1D)

{

functionValues[index] = y;

}

// Зберігаємо значення x та y у двовимірний масив (обмежено розміром rows x cols)

if (index < rows \* cols)

{

xyValues[index / cols][index % cols] = y;

}

std::cout << "Function value in [" << x << "] is -> " << y << std::endl;

x += 0.5;

index++;

} while (x <= 2);

// Використання циклу while для виведення значень з одновимірного масиву

int i = 0;

while (i < size1D)

{

std::cout << "Value from 1D array [" << i << "] -> " << functionValues[i] << std::endl;

i++;

}

for (int row = 0; row < rows; row++)

{

for (int col = 0; col < cols; col++)

{

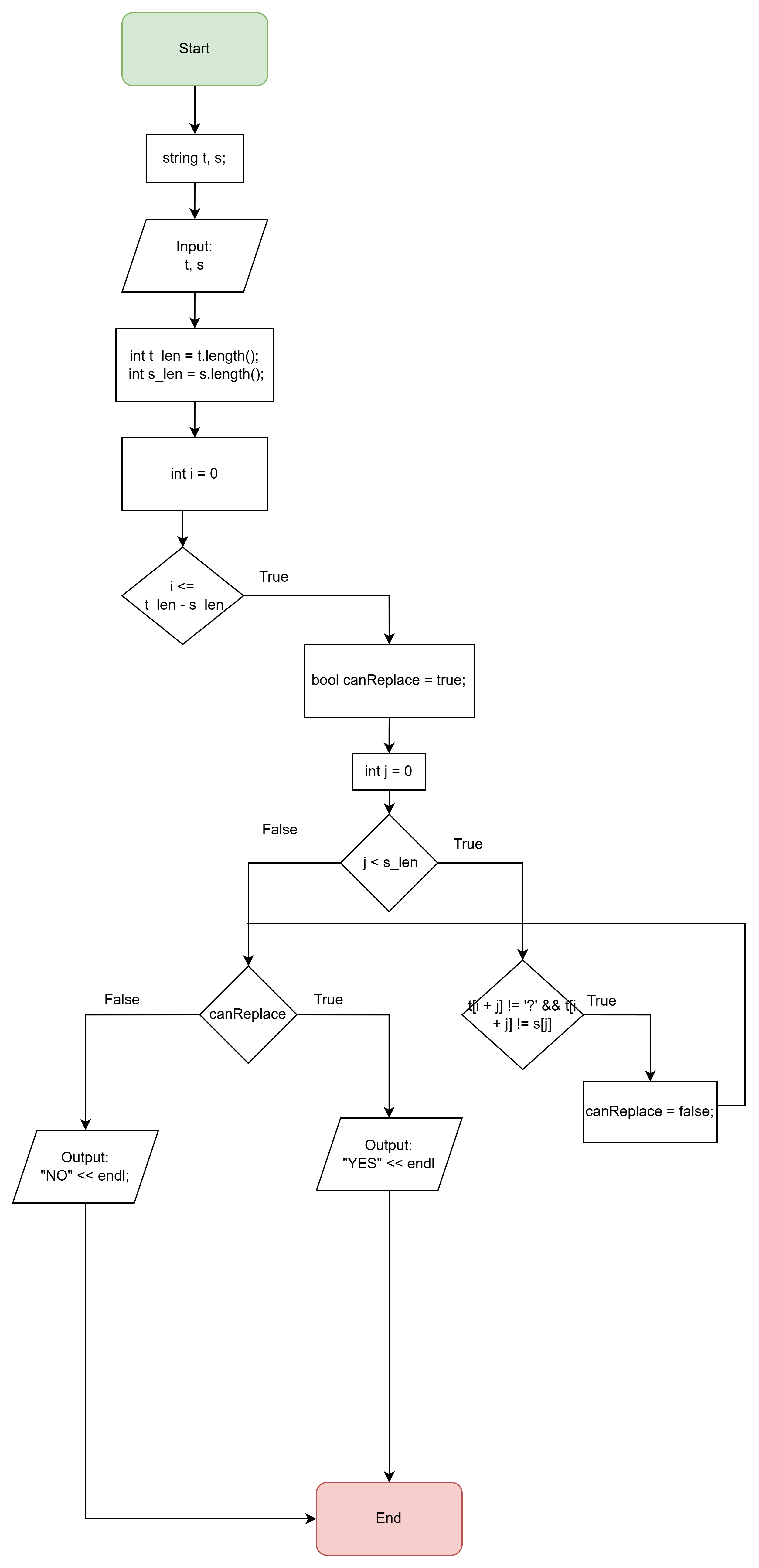
std::cout << "Value from 2D array [" << row << "][" << col << "] -> " << xyValues[row][col] << std::endl;

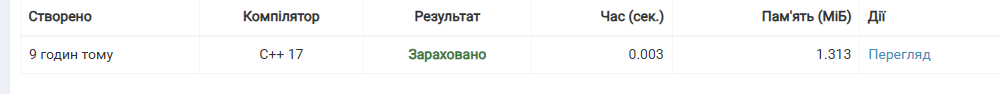
}

}

return 0;

}





**3.**

#include <iostream>

#include <iomanip>

using namespace std;

int main() {

int n, k;

cin >> n >> k;

if (n == 0 || k == 0) {

cout << fixed << setprecision(6) << 0.0 << endl;

} else {

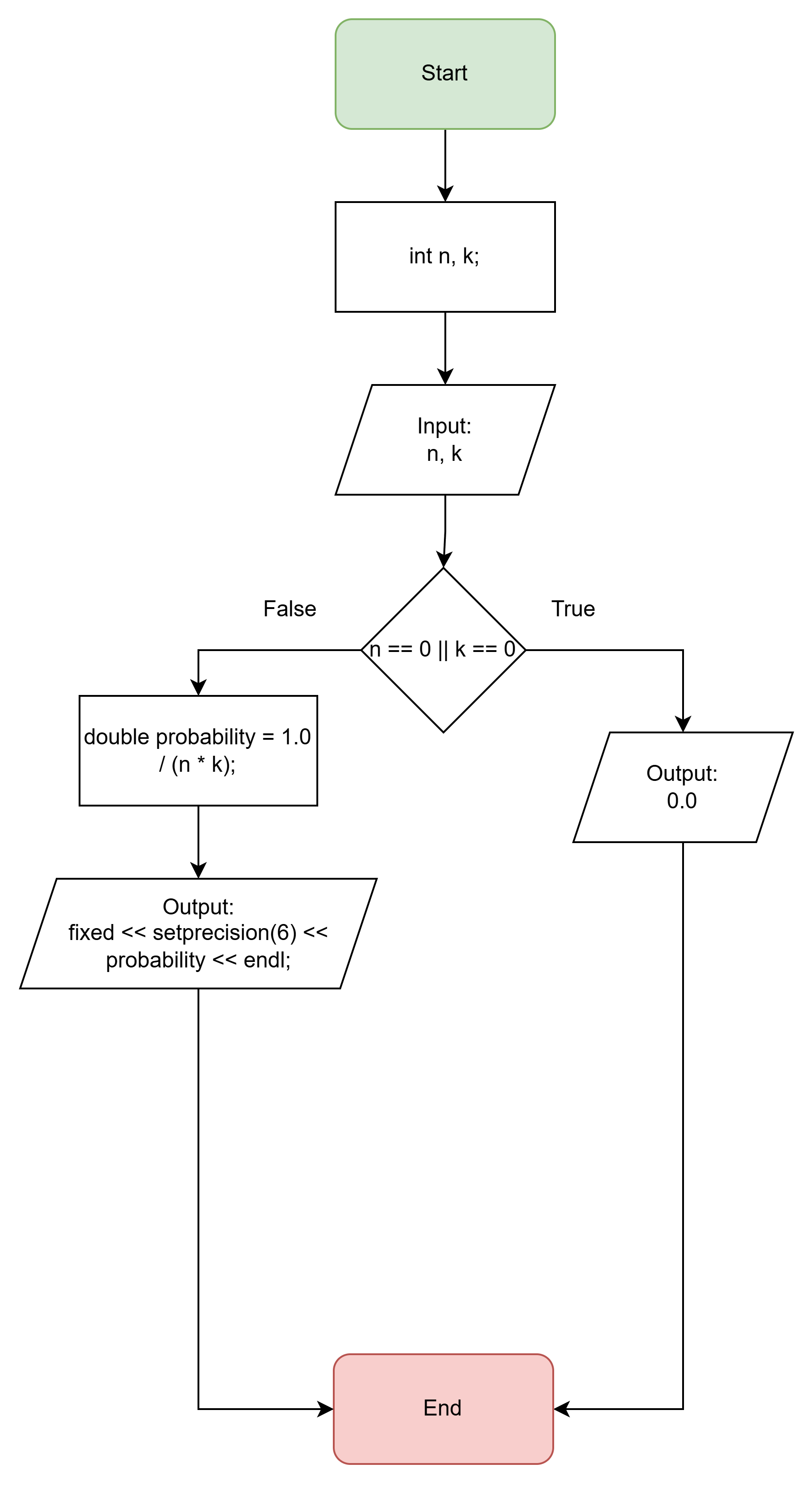
double probability = 1.0 / (n \* k);

cout << fixed << setprecision(6) << probability << endl;

}

return 0;

}



**4.**

#include <iostream>

#include <vector>

#include <string>

#include <unordered\_map>

using namespace std;

int main() {

string d;

cin >> d;

int n;

cin >> n;

vector<string> days = {"Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday"};

unordered\_map<string, int> day\_index;

for (int i = 0; i < 7; i++) {

day\_index[days[i]] = i;

}

int start\_day = day\_index[d];

int full\_weeks = n / 7;

int remaining\_days = n % 7;

vector<int> counts(7, full\_weeks);

for (int i = 0; i < remaining\_days; i++) {

counts[(start\_day + i) % 7]++;

}

for (int count : counts) {

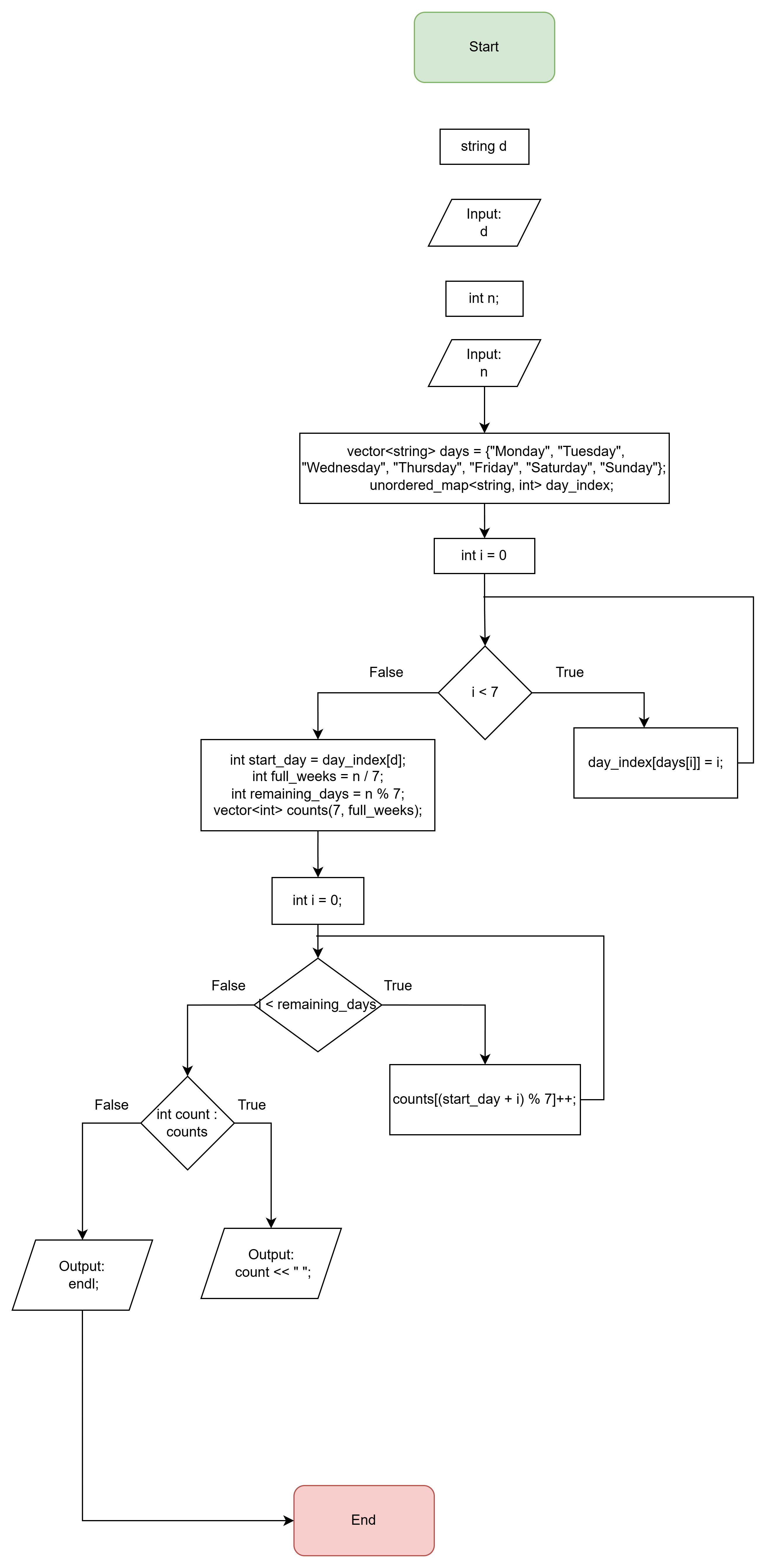
cout << count << " ";

}

cout << endl;

return 0;

}





**Pull Request:** [**LINK**](https://github.com/artificial-intelligence-department/ai_programming_playground_2024/pull/278)

**Висновок:**   
Я навчився використовувати базові алгоритми ітерації по масивами , також використав безпечне приведення типів в С++