Міністерство освіти України

Національний технічний університет "ХПІ"

кафедра "Стратегічного управління і системного аналізу”

**Звіт**

**Лабораторна робота 6**

з дисципліни "Основи Програмування"

Виконав: студент групи КН-1223А

.Джим І.Р

Перевірив:

Івашко А.В.

Харків 2023

Зміст

[**Завдання** 3](#_Toc151714937)

[**Мета роботи** 3](#_Toc151714938)

[**Завдання для виконання** 3](#_Toc151714939)

[**Рішення** 3](#_Toc151714940)

[**Завдання 1** 3](#_Toc151714941)

[**Завдання 2** 4](#_Toc151714942)

[**Завдання 3** 5](#_Toc151714943)

[**Код проекту** 6](#_Toc151714944)

[**Висновок** 10](#_Toc151714945)

# **Завдання**

# **Мета роботи**

Метою лабораторної роботи є отримання практичних навичок в роботі з умовним оператором і розгалуженими алгоритмами в мові ***C***.

# **Завдання для виконання**

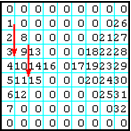
Заповнити сектори матриці, які лежать вліво і вправо від головної і побічної діагоналей, ЛПЧ, від лівого верхнього кута вниз – вправо. Залишок матриці заповнити нулями.

У двовимірному масиві (кожного разу працювати з даним масивом):

а) знайти максимальний в кожному рядку і поміняти його місцями з першим елементом цього ж рядка;

б) знайти мінімальний в кожному стовпці і поміняти його місцями з останнім елементом цього ж стовпця.

Перемножування двох матриць. Розмір матриць вводити з екрану. Здійснити перевірку можливість перемножування.



# **Рішення**

# **Завдання 1**

Цей код створює матрицю розміром 20x20 і заповнює її значеннями 0 або 1 в залежності від умов, заданих у внутрішньому циклі. Основна мета цього коду - генерація та виведення в консоль графічного зображення.

Давайте розглянемо код крок за кроком:

1. Задається розмірність матриці `rows` та `cols` (20 на 20).
2. Створюється двомірний масив `matrix` розміром 20x20.

3. Запускається зовнішній цикл для ітерації по рядках (i) від 0 до 19.

4. У внутрішньому циклі для кожного елементу матриці обчислюються значення `x` та `y` на основі поточного індексу (`i` та `j`).

5. Визначаються умови для заповнення кожного елементу матриці значеннями 0 або 1. Значення 0 призначається, якщо `(y >= x) && (y >= -x) && (y <= 1) && (y >= -1)` або `(y <= 1) && (y >= -1) && (y <= x) && (y <= -x)`. В інших випадках значення елементу матриці встановлюється в 1.

6. Після заповнення матриці виводяться її значення в консоль.

7. Завершується програма з використанням `system("pause")` та `system("cls")`. `system("pause")` призначений для тимчасового призупинення виконання програми, щоб користувач міг переглянути результати, а `system("cls")` використовується для очищення консолі.

Загалом, цей код генерує і відображає графічне зображення в консолі на основі вказаних умов для кожного елементу матриці.

# **Завдання 2**

Цей код створює динамічний двовимірний масив (матрицю) розміром 9x9, заповнює його значеннями за допомогою функції `FillMatrix`, виводить початкову матрицю за допомогою функції `PrintMatrix`, замінює перший елемент кожного рядка на максимальний елемент цього рядка за допомогою функції `SwapFirstAndMaxInEveryRow`, виводить матрицю після першої заміни, замінює останній елемент кожного рядка на мінімальний елемент цього рядка за допомогою функції `SwapLastAndMinInEveryRow`, і виводить матрицю після другої заміни.

Основні етапи роботи коду:

1. Створення динамічної матриці з вказаною кількістю рядків і стовпців.

2. Заповнення матриці значеннями за допомогою функції `FillMatrix`.

3. Виведення початкової матриці на екран за допомогою функції `PrintMatrix`.

4. Заміна першого елемента кожного рядка на максимальний елемент цього

рядка за допомогою функції `SwapFirstAndMaxInEveryRow`.

5. Виведення матриці після першої заміни на екран.

6. Заміна останнього елемента кожного рядка на мінімальний елемент цього рядка за допомогою функції `SwapLastAndMinInEveryRow`.

7. Виведення матриці після другої заміни на екран.

8. Очищення виділеної пам'яті для матриці.

9. Виклик команд `system("pause")` та `system("cls")`, які можуть використовуватися для тимчасового зупинення виконання програми та очищення екрану від попереднього виведення.

# **Завдання 3**

1. `fillMatrixRandom` функція:

- Призначення: Заповнює матрицю випадковими значеннями в діапазоні від 0 до 99.

- Параметри:

- `matrix[MAX\_SIZE][MAX\_SIZE]`: Матриця, яку треба заповнити.

- `rows`: Кількість рядків матриці.

- `cols`: Кількість стовпців матриці.

2. `multiplyMatrices` функція:

- Призначення: Множить дві матриці і зберігає результат в третій матриці.

- Параметри:

- `firstMatrix[MAX\_SIZE][MAX\_SIZE]`: Перша матриця для множення.

- `secondMatrix[MAX\_SIZE][MAX\_SIZE]`: Друга матриця для множення.

- `result[MAX\_SIZE][MAX\_SIZE]`: Матриця для зберігання результату множення.

- `rowsFirst`, `colsFirst`: Розміри першої матриці.

- `rowsSecond`, `colsSecond`: Розміри другої матриці.

3. `displayMatrix` функція:

- Призначення: Виводить матрицю на екран.

- Параметри:

- `matrix[MAX\_SIZE][MAX\_SIZE]`: Матриця для виведення.

- `rows`: Кількість рядків матриці.

- `cols`: Кількість стовпців матриці.

4. Task3 функція:

- Призначення: Виконує завдання з множення матриць.

- Читає розміри першої та другої матриць з консолі.

- Якщо кількість стовпців першої матриці не дорівнює кількість рядків другої, виводить повідомлення про помилку.

- Заповнює обидві матриці випадковими значеннями.

- Виводить на екран обидві матриці.

- Викликає `multiplyMatrices` для множення матриць та виводить результат.

- Використовує `system("pause")` та `system("cls")` для затримки та очищення консолі.

# **Код проекту**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream>

#include <iomanip>

#include <stdlib.h>

#include <Windows.h>

#include <random>

using namespace std;

void Task1();

void Task2();

void Task3();

int random(int from, int to) {

int result = rand() % (to - from + 1) + from;

return result;

}

void SwapElements(int\*\* matrix, int aRow, int aColumn, int bRow, int bColumn) {

int temp = matrix[aRow][aColumn];

matrix[aRow][aColumn] = matrix[bRow][bColumn];

matrix[bRow][bColumn] = temp;

}

void FillMatrix(int\*\* matrix, int rows, int columns) {

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

for (int j = 0; j < columns; j++) {

matrix[i][j] = random(10, 99);

}

}

}

void PrintMatrix(int\*\* matrix, int rows, int columns) {

HANDLE h = GetStdHandle(STD\_OUTPUT\_HANDLE);

SetConsoleTextAttribute(h, 15);

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

for (int j = 0; j < columns; j++) {

if (matrix[i][j] < 55) {

SetConsoleTextAttribute(h, 15);

}

else {

SetConsoleTextAttribute(h, 15);

}

cout << setw(3) << matrix[i][j];

}

cout << endl;

}

SetConsoleTextAttribute(h, 15);

}

void SwapFirstAndMaxInEveryRow(int\*\* matrix, int rows, int columns) {

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

int maxIndex = 0;

for (int j = 0; j < columns; j++) {

if (matrix[i][maxIndex] < matrix[i][j]) {

maxIndex = j;

}

}

SwapElements(matrix, i, 0, i, maxIndex);

}

}

void SwapLastAndMinInEveryRow(int\*\* matrix, int rows, int columns) {

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

int minIndex = 0;

for (int j = 0; j < columns; j++) {

if (matrix[i][minIndex] > matrix[i][j]) {

minIndex = j;

}

}

SwapElements(matrix, i, columns - 1, i, minIndex);

}

}

enum class MenuOptions { ClearConsole = -1, Exit = 0, Task1, Task2, Task3 };

void PrintMenuAndAskChoice(const HANDLE& h, MenuOptions& choice)

{

SetConsoleTextAttribute(h, 15);

printf("Виберіть програму:");

SetConsoleTextAttribute(h, 10);

SetConsoleTextAttribute(h, 15);

printf("\nЗаповнення частин матриці - 1");

SetConsoleTextAttribute(h, 10);

SetConsoleTextAttribute(h, 15);

printf("\nМаксимальні та мінімальні елементи масиву - 2");

SetConsoleTextAttribute(h, 10);

SetConsoleTextAttribute(h, 15);

printf("\nПеремножування двох матриць - 3");

SetConsoleTextAttribute(h, 10);

SetConsoleTextAttribute(h, 15);

printf("\nВихід - 0 \nПрограма: ");

SetConsoleTextAttribute(h, 6);

scanf("%i", &choice);

SetConsoleTextAttribute(h, 15);

}

int main()

{

srand(time(0));

setlocale(LC\_ALL, "Russian");

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

HANDLE h = GetStdHandle(STD\_OUTPUT\_HANDLE);

SetConsoleTextAttribute(h, 15);

MenuOptions choice;

while (true) {

PrintMenuAndAskChoice(h, choice);

switch (choice)

{

case MenuOptions::Task1:

Task1();

break;

case MenuOptions::Task2:

Task2();

break;

case MenuOptions::Task3:

Task3();

break;

case MenuOptions::Exit:

SetConsoleTextAttribute(h, 4);

cout << "Завершення...\n";

SetConsoleTextAttribute(h, 15);

system("pause");

system("cls");

return 0;

default:

SetConsoleTextAttribute(h, 4);

printf("Неправильне значення!\n");

SetConsoleTextAttribute(h, 15);

system("pause");

system("cls");

break;

}

}

}

const int MAX\_SIZE = 10;

void fillMatrixRandom(int matrix[MAX\_SIZE][MAX\_SIZE], int rows, int cols) {

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

for (int j = 0; j < cols; ++j) {

matrix[i][j] = rand() % 100;

}

}

}

void multiplyMatrices(int firstMatrix[MAX\_SIZE][MAX\_SIZE], int secondMatrix[MAX\_SIZE][MAX\_SIZE], int result[MAX\_SIZE][MAX\_SIZE], int rowsFirst, int colsFirst, int rowsSecond, int colsSecond) {

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

for (int j = 0; j < colsSecond; ++j) {

result[i][j] = 0;

for (int k = 0; k < colsFirst; ++k) {

result[i][j] += firstMatrix[i][k] \* secondMatrix[k][j];

}

}

}

}

void displayMatrix(int matrix[MAX\_SIZE][MAX\_SIZE], int rows, int cols) {

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

for (int j = 0; j < cols; ++j) {

cout << matrix[i][j] << " ";

}

cout << endl;

}

}

void Task3() {

const int MAX\_SIZE = 10;

int rowsFirst, colsFirst, rowsSecond, colsSecond;

cout << "Визначте параметри першої матриці: ";

cin >> rowsFirst >> colsFirst;

cout << "Визначте параметри другої матриці: ";

cin >> rowsSecond >> colsSecond;

if (colsFirst != rowsSecond) {

cout << "Неправильне значення!";

system("pause");

system("cls");

}

int firstMatrix[MAX\_SIZE][MAX\_SIZE], secondMatrix[MAX\_SIZE][MAX\_SIZE], result[MAX\_SIZE][MAX\_SIZE];

fillMatrixRandom(firstMatrix, rowsFirst, colsFirst);

fillMatrixRandom(secondMatrix, rowsSecond, colsSecond);

cout << "\nПерша матриця:\n";

displayMatrix(firstMatrix, rowsFirst, colsFirst);

cout << "\nДруга матриця:\n";

displayMatrix(secondMatrix, rowsSecond, colsSecond);

multiplyMatrices(firstMatrix, secondMatrix, result, rowsFirst, colsFirst, rowsSecond, colsSecond);

cout << "\nІтог множення:\n";

displayMatrix(result, rowsFirst, colsSecond);

system("pause");

system("cls");

}

void Task2() {

int rows = 9;

int columns = 9;

int\*\* matrix = new int\* [rows];

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

matrix[i] = new int[columns];

}

FillMatrix(matrix, rows, columns);

cout << "Початковий масив:\n";

PrintMatrix(matrix, rows, columns);

SwapFirstAndMaxInEveryRow(matrix, rows, columns);

cout << "Перша заміна:\n";

PrintMatrix(matrix, rows, columns);

SwapLastAndMinInEveryRow(matrix, rows, columns);

cout << "Друга заміна:\n";

PrintMatrix(matrix, rows, columns);

cout << endl;

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

delete[] matrix[i];

}

delete[] matrix;

system("pause");

system("cls");

}

void Task1() {

const int rows = 20;

const int cols = 20;

int matrix[rows][cols];

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

for (int j = 0; j < cols; ++j) {

double x = static\_cast<double>(j) / (cols - 1) \* 2 - 1;

double y = 1 - static\_cast<double>(i) / (rows - 1) \* 2;

if ((y >= x) && (y >= -x) && (y <= 1) && (y >= -1)) {

matrix[i][j] = 0;

}

else if ((y <= 1) && (y >= -1) && (y <= x) && (y <= -x)) {

matrix[i][j] = 0;

}

else {

matrix[i][j] = 1;

}

}

}

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

for (int j = 0; j < cols; ++j) {

std::cout << matrix[i][j] << " ";

}

std::cout << std::endl;

}

system("pause");

system("cls");

}

# **Висновок**

В ході виконання лабораторної роботи у середовищі Visual Studio були отримані важливі навички програмування та роботи з інтегрованим середовищем розробки. За допомогою мови програмування, яка була

використана в цій лабораторній роботі, вдалося створити програму, яка відповідає поставленим завданням.

Одним з ключових вмінь, які були розвинуті, є вміння розробляти і відлагоджувати код. Під час лабораторної роботи було необхідно виявити та виправити помилки в програмі, що сприяє покращенню розуміння процесу відлагодження.

Також, лабораторна робота дозволила ознайомитися з основними концепціями створення графічного інтерфейсу користувача, що є важливим аспектом розробки програм. Використання графічного інтерфейсу полегшило взаємодію з користувачем та зробило програму більш зручною та доступною.

Під час виконання лабораторної роботи було надано можливість вивчити та використовувати різноманітні бібліотеки та інструменти, які розширюють можливості Visual Studio. Це дозволяє покращити продуктивність та ефективність розробки.

Завдяки цій лабораторній роботі, ми також набули досвіду у спільній роботі над програмним проектом, що може бути корисним у майбутньому професійному житті.

У підсумку, виконання цієї лабораторної роботи дало нам можливість розширити свої знання та вміння в області програмування, відлагодження коду, створення графічного інтерфейсу та спільної роботи над проектами. Все це є важливими компонентами розвитку як професійного програміста.