Міністерство освіти і науки України

Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського"

Факультет інформатики та обчислювальної техніки

Кафедра автоматизованих систем обробки інформації і управління

Звіт

з лабораторної роботи № 8 з дисципліни «Основи програмування-1»

«Багатовимірні масиви»

Варіант 8

Виконав студент ІП-02 Глазунова Поліна Андріївна

(шифр, прізвище, ім'я, по батькові)

Перевірив Вітковська Ірина Іванівна

( прізвище, ім'я, по батькові)

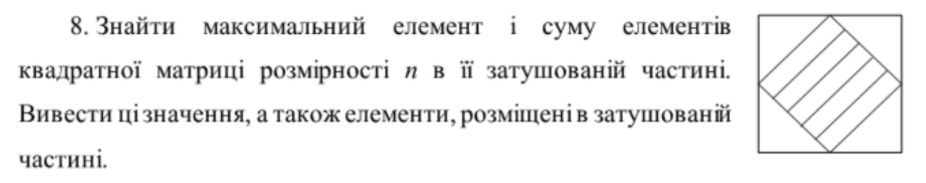
Київ 2020

**Лабораторна робота 8**

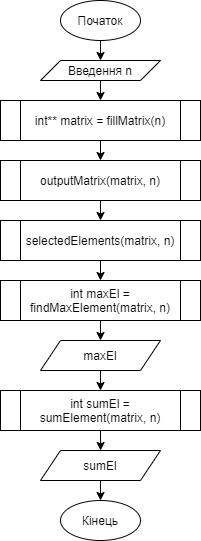
**Багатовимірні масиви**

**Мета** – опанувати технологію використання двовимірних масивів даних (матриць), навчитися розробляти алгоритми та програми із застосуванням матриць.

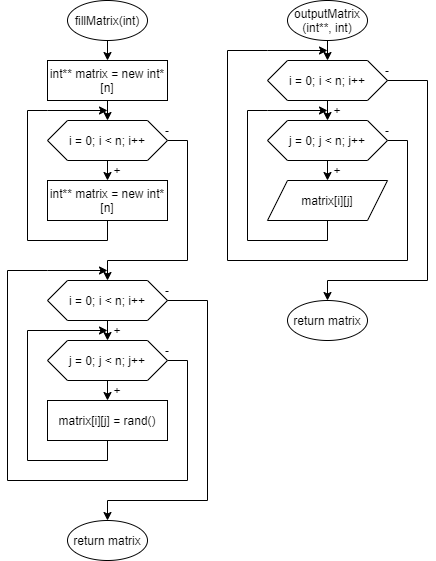
**Варіант 8**

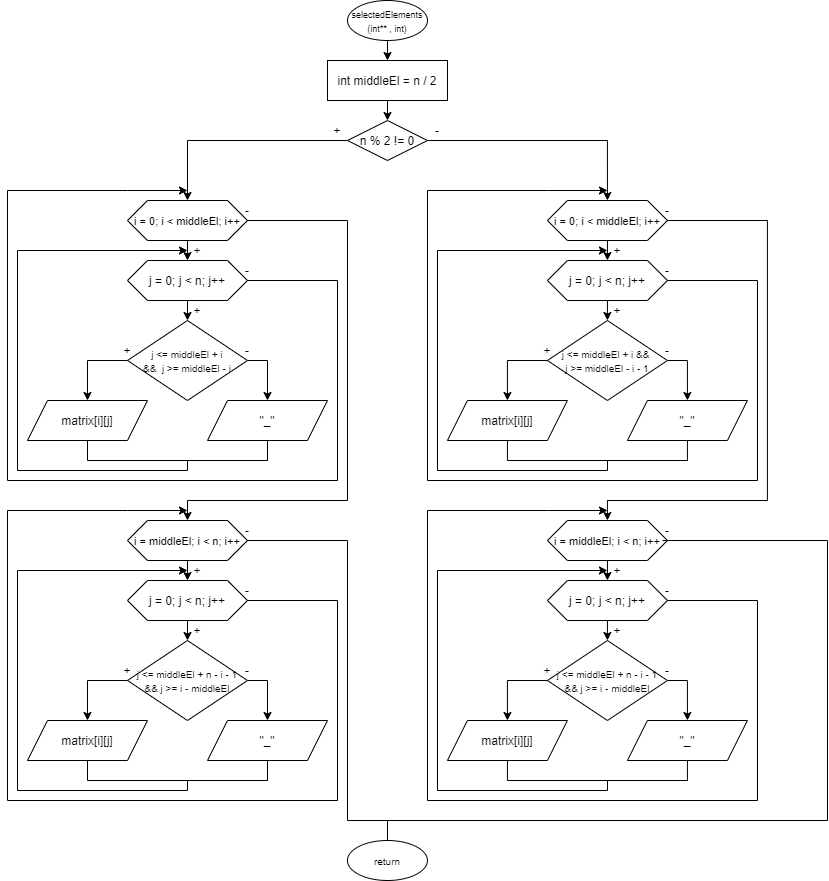


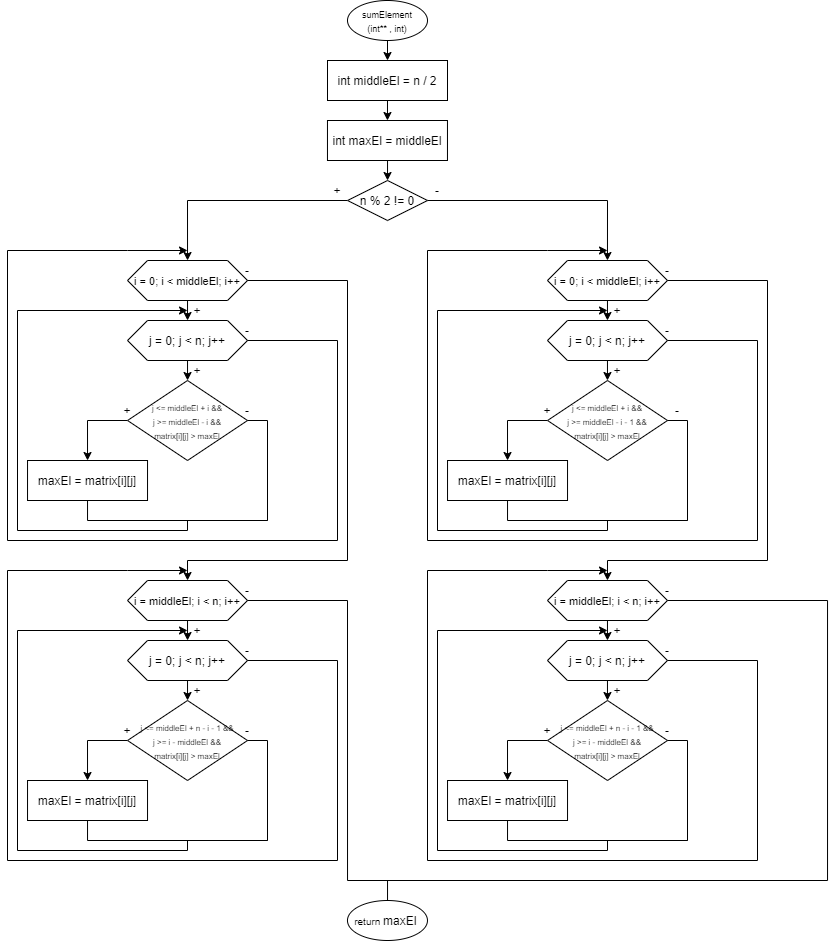
***Основна функція:***

******

***Підпрограми:***



**

**

**С++**

***//========= matrix\_func.h ============***

int\*\* fillMatrix(int n); // прототип функції для ініціювання двовимірного масиву (матриці n х n) випадковими числами

void outputMatrix(int\*\* matrix, int n); // прототип функції для виведення двовимірного масиву (матриці n х n)

void selectedElements(int\*\* matrix, int n); // прототип функції для виведення елементів, що розташовані в виділеній частині матриці

int findMaxElement(int\*\* matrix, int n); // прототип функції для пошуку максимального елементу в затушованій частині

int sumElement(int\*\* matrix, int n); // прототип функції для пошуку суми елементів в затушованій частині

***//========= matrix\_func.cpp ============***

#include<iostream>

#include <cstdlib>

#include <ctime>

#include <cmath>

#include <iomanip>

using namespace std;

//=========Визначення функцій============

int\*\* fillMatrix(int n) // функція для ініціювання двовимірного масиву (матриці n х n) випадковими числами

{

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

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

matrix[i] = new int[n];

}

srand(time(NULL));

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

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

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

}

}

return matrix;

}

void outputMatrix(int\*\* matrix, int n) // функція для виведення двовимірного масиву (матриці n х n)

{

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

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

cout << setw(5) << matrix[i][j] << " ";

}

cout << endl;

}

}

void selectedElements(int\*\* matrix, int n) // функція для виведення елементів, що розташовані в виділеній частині матриці

{

int middleEl = n / 2;

if (n % 2 != 0) // перевірка на парність

{

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

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

if (j <= middleEl + i && j >= middleEl - i) cout << setw(5) << matrix[i][j] << " ";

else cout << setw(5) << "\_" << " ";

}

cout << endl;

}

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

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

if (j <= middleEl + n - i - 1 && j >= i - middleEl) cout << setw(5) << matrix[i][j] << " ";

else cout << setw(5) << "\_" << " ";

}

cout << endl;

}

}

else {

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

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

if (j <= middleEl + i && j >= middleEl - i - 1) cout << setw(5) << matrix[i][j] << " ";

else cout << setw(5) << "\_" << " ";

}

cout << endl;

}

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

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

if (j <= middleEl + n - i - 1 && j >= i - middleEl) cout << setw(5) << matrix[i][j] << " ";

else cout << setw(5) << "\_" << " ";

}

cout << endl;

}

}

}

int findMaxElement(int\*\* matrix, int n) // функція для пошуку максимального елементу в затушованій частині

{

int middleEl = n / 2;

int maxEl = middleEl;

if (n % 2 != 0) // перевірка на парність

{

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

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

if (j <= middleEl + i && j >= middleEl - i && matrix[i][j] > maxEl) maxEl = matrix[i][j];

}

}

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

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

if (j <= middleEl + n - i - 1 && j >= i - middleEl && matrix[i][j] > maxEl) maxEl = matrix[i][j];

}

}

}

else {

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

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

if (j <= middleEl + i && j >= middleEl - i - 1 && matrix[i][j] > maxEl) maxEl = matrix[i][j];

}

}

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

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

if (j <= middleEl + n - i - 1 && j >= i - middleEl && matrix[i][j] > maxEl) maxEl = matrix[i][j];

}

}

}

return maxEl;

}

int sumElement(int\*\* matrix, int n) // функція для пошуку суми елементів в затушованій частині

{

int middleEl = n / 2;

int sum = 0;

if (n % 2 != 0) // перевірка на парність

{

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

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

if (j <= middleEl + i && j >= middleEl - i) sum += matrix[i][j];

}

}

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

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

if (j <= middleEl + n - i - 1 && j >= i - middleEl) sum += matrix[i][j];

}

}

}

else {

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

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

if (j <= middleEl + i && j >= middleEl - i - 1) sum += matrix[i][j];

}

}

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

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

if (j <= middleEl + n - i - 1 && j >= i - middleEl) sum += matrix[i][j];

}

}

}

return sum;

}

***//========= основний файл============***

#include<iostream>

#include "matrix\_func.h"

#include "matrix\_func.cpp"

using namespace std;

//=========Головна функція============

int main() {

int n; // розмір квадратної матриці

cout << "Enter the number of rows and columns: ";

cin >> n;

int\*\* matrix = fillMatrix(n); // двовимірний масив (квадратна матриця)

cout << "Matrix: " << endl;

outputMatrix(matrix, n); // виведення початкової матриці

cout << endl;

cout << "Selected elements: " << endl;

selectedElements(matrix, n); // виведення елементів, розташованих в затушованій частині

int maxEl = findMaxElement(matrix, n); // максимальний елемент затушованої частині

cout << "Max element:" << maxEl << endl;

int sumEl = sumElement(matrix, n); // сума елементів в затушованій частині

cout << "Sum of selected elements:" << sumEl << endl;

system("pause");

}

**Відеокопія результату:**

