

Міністерство освіти і науки України  
Національний технічний університет України «Київський політехнічний  
інститут імені Ігоря Сікорського»  
Факультет інформатики та обчислювальної техніки  
Кафедра інформатики та програмної інженерії

Звіт

З лабораторної роботи №8 з дисципліни

«Основи програмування-1.

Базові конструкції»

«Багатовимірні масиви»

Варіант 27

Виконав студент ІП-11 Савенко Олексій Андрійович

(шифр, прізвище, ім'я, по батькові)

Перевірів Вітковська І.І.

(прізвище, ім'я, по батькові)

Київ 2021

## Лабораторна робота 8

### Багатовимірні масиви

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

### Варіант 27

#### Індивідуальне завдання

27. Задані цілочисельні матриці  $A$  ( $n \times n$ ) та  $B$  ( $n \times n$ ). У кожній із них поміняти місцями мінімальні елементи рядка і відповідні елементи головної діагоналі. Побудувати матрицю  $Q$  ( $n \times n$ ), елементи якої обчислюються за формулою  $Q_{ij} = \max(A_{ij}, B_{ij})$ .

#### Постановка завдання

Створити дві матриці A,B розмірності (n x n) за рахунок динамічного двовимірного масиву, який у свою чергу буде створений за допомогою масиву покажчиків, за допомогою якого ми зможемо взаємодіяти з елементами одновимірних динамічних масивів, що являють собою рядки двовимірного масиву. Після створення виконати перестановку мінімального елемента рядка та відповідного діагонального елемента. На основі видозмінених матриць створити третю матрицю Q за допомогою порівняння відповідних елементів з двох матриць та знаходження максимального і внесення його як елемента до матриці Q. Після знаходження результату потрібно очистити пам'ять виділену для динамічних об'єктів.

#### Програмний код на мові C++:

```
#include <iostream>
#include <ctime>
using namespace std;

//Прототипізація функцій
void Generation(int**, int);
void Input(int**, int);
```

```

void Output(int**, int);
void Changer(int**, int);
void Creator(int**, int**, int**, int);
void Destruction(int**, int);

int main() {
    srand(time(NULL));

    int n;
    cout << "Enter amount of Matrix: "; cin >> n; //Введення розмірності матриці

    int** A = new int* [n]; //Створення динамічного масива покажчиків з n-
покажчиків
    Generation(A, n); //Створення динамічного масиву, введення
елементів у нього та їх виведення
    Input(A, n);
    cout << "Matrix A" << endl; Output(A, n);

    int** B = new int* [n]; //Створення динамічного масива покажчиків з n-
покажчиків
    Generation(B, n); Input(B, n); //Створення динамічного масиву, введення
елементів у нього та їх виведення
    cout << endl << "Matrix B" << endl; Output(B, n);

    Changer(A, n); Changer(B, n); //Видозмінення матриці зміна діагонального
елементу на мінімальний та навпаки
    cout << endl << "Matrix A after changes" << endl; Output(A, n);
    cout << endl << "Matrix B after changes" << endl; Output(B, n);

    int** Q = new int* [n]; //Створення динамічного масива покажчиків з n-
покажчиків
    Generation(Q, n); //Створення динамічного масиву, введення елементів на
основі більшого елементу з двох матриць у нього та їх виведення
    Creator(A, B, Q, n);
    cout << endl << "Matrix Q" << endl; Output(Q, n);

    //Звільнення ініціалізованої пам'яті для Динамічних Масивів
    Destruction(A, n);
    Destruction(B, n);
    Destruction(Q, n);
    return 0;
}

//Генерація динамічних одновимірних масивів, які стануть рядками двовимірного масива
void Generation(int** Matrix, int size) {
    for (int i = 0; i < size; i++) {
        Matrix[i] = new int[size];
    }
}

//Заповнення масива елементами
void Input(int** Matrix, int size) {
    for (int i = 0; i < size; i++) {
        for (int j = 0; j < size; j++) {
            Matrix[i][j] = rand() % 101;
        }
    }
}
}

```

```

//Виведення масиву
void Output(int** Matrix, int size) {

    for (int i = 0; i < size; i++) {
        cout << endl;
        for (int j = 0; j < size; j++) {
            cout << "\t" << Matrix[i][j];
        }
    }

}

//Зміна елементів місцями у масиві - зміна діагонального елементу на мінімальний та
навпаки
void Changer(int** Matrix, int size) {
    int min, pos, diag;
    for (int i = 0; i < size; i++) {
        min = Matrix[i][0];
        diag = Matrix[i][i];
        for (int j = 0; j < size; j++) {
            if (Matrix[i][j] <= min) {
                min = Matrix[i][j];
                pos = j; //Запам'ятовування позиції мінімального елемента
            }
        }
        if (min != diag) {
            Matrix[i][i] = min;
            Matrix[i][pos] = diag;
        }
    }
}

//Створення масиву Q на основі двох перших масивів за допомогою порівняння та віднайдення
відповідного більшого елемента
void Creator(int** A, int** B, int** Q, int size) {
    for (int i = 0; i < size; i++) {
        for (int j = 0; j < size; j++) {
            if (A[i][j] >= B[i][j]) {
                Q[i][j] = A[i][j];
            }
            else {
                Q[i][j] = B[i][j];
            }
        }
    }
}

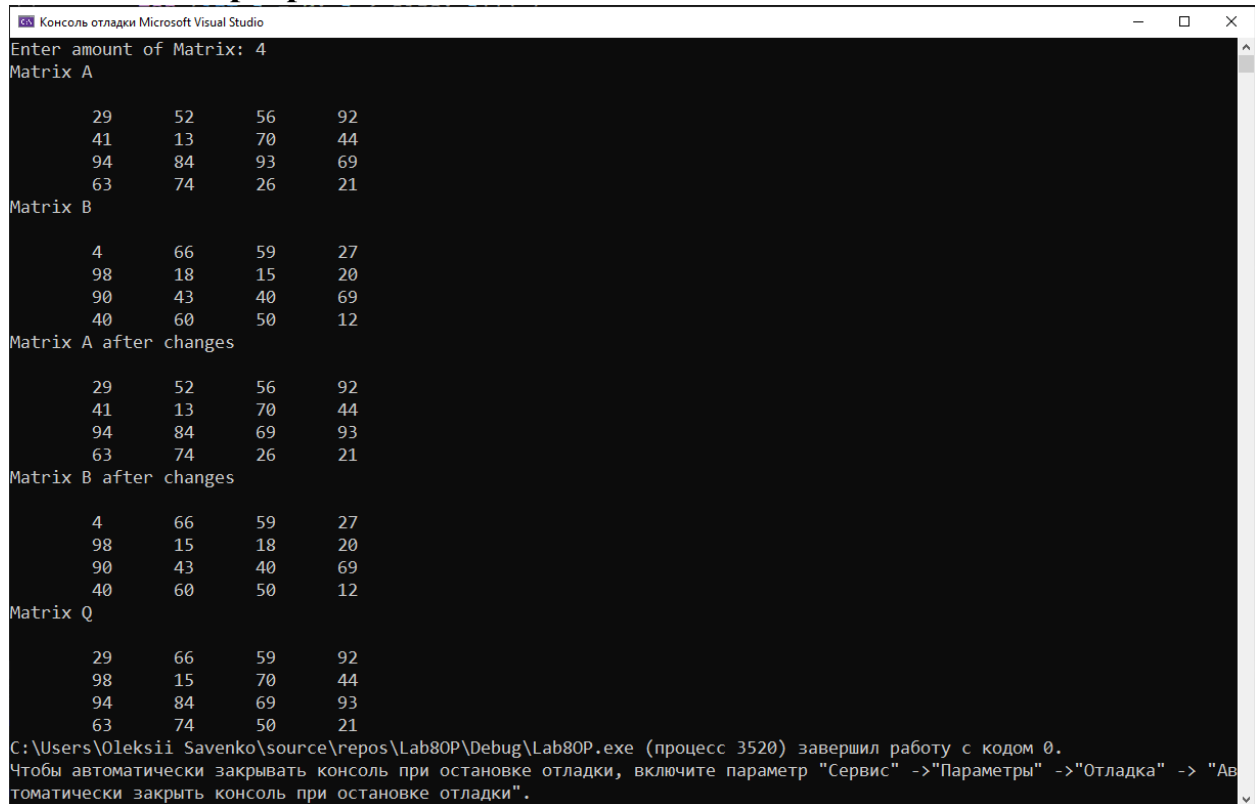
//Знищення матриці та вивільнення відповідної ділянки пам'яті, знищення відповідного
масива покажчиків
void Destruction(int** Matrix, int size) {

    for (int i = 0; i < size; i++) {
        delete[] Matrix[i];
    }
    delete[] Matrix;

}

```

## Виконання програми на мові C++:



```
Консоль отладки Microsoft Visual Studio
Enter amount of Matrix: 4
Matrix A
    29    52    56    92
    41    13    70    44
    94    84    93    69
    63    74    26    21
Matrix B
    4     66    59    27
    98    18    15    20
    90    43    40    69
    40    60    50    12
Matrix A after changes
    29    52    56    92
    41    13    70    44
    94    84    69    93
    63    74    26    21
Matrix B after changes
    4     66    59    27
    98    15    18    20
    90    43    40    69
    40    60    50    12
Matrix Q
    29    66    59    92
    98    15    70    44
    94    84    69    93
    63    74    50    21
C:\Users\Oleksii Savenko\source\repos\Lab80P\Debug\Lab80P.exe (процесс 3520) завершил работу с кодом 0.
Чтобы автоматически закрывать консоль при остановке отладки, включите параметр "Сервис" ->"Параметры" ->"Отладка" -> "Автоматически закрыть консоль при остановке отладки".
```

## Висновок

Отже, я опанував технологію використання двовимірних масивів даних(матриць), навчився розробляти алгоритми та програми із

застосуванням матриць. Для вирішення поставленого завдання було розроблено алгоритм та написано програмний код на мові програмування C++ з використанням динамічних двовимірних масивів. Для створення та використання масивів було створено масив покажчиків, де було вказано кількість рядків майбутнього двовимірного масива. Після чого у функції **Generation()**, було створено відповідну кількість одновимірних масивів з  $n$ -стовпцями, які стануть рядками матриці. За допомогою функції **Input()** матриці було заповнено елементами, а за допомогою функції **Output()** виведено. Функція **Changer()** поміняла місцями відповідні мінімальні елементи рядка та діагональні елементи, на основі видозмінених матриць за допомогою функції **Creator()** було створено матрицю  $Q$  за рахунок порівняння відповідних елементів матриць та вибору максимального з них як елемента для матриці  $Q$ , після отримання результату пам'ять виділену для динамічних об'єктів було очищено. Після чого було проведено випробування алгоритму та доведено його вірність. Отже, мій програмний код та алгоритм можна використовувати для вирішення завдань даного типу.