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 № 1 з дисципліни
«Основи програмування-1.
Базові конструкції»

«Багатовимірні масиви»

Варіант 28

Виконав студент ІП-11 Сідак Кирил Ігорович
(шифр, прізвище, ім'я, по батькові)

Перевірив _____
(прізвище, ім'я, по батькові)

Київ 2021

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

Варіант 28

У двох заданих дійсних квадратних матрицях розмірності n поміняти місцями їх головні діагоналі, попередньо упрядкувавши за збільшенням елементи цих діагоналей.

Постановка задачі:

Спочатку треба створити два динамічні двовимірні масиви розмірності n на т та заповнити їх випадковими дійсними числами з певного проміжку, потім в даних масивах за допомогою алгоритму сортування упорядкувати за збільшенням елементи головних діагоналей та замінити кожний елемент головної діагоналі одного масиву відповідним елементом головної діагоналі іншого масиву. Після цього потрібно видалити з пам’яті обидва динамічні масиви.

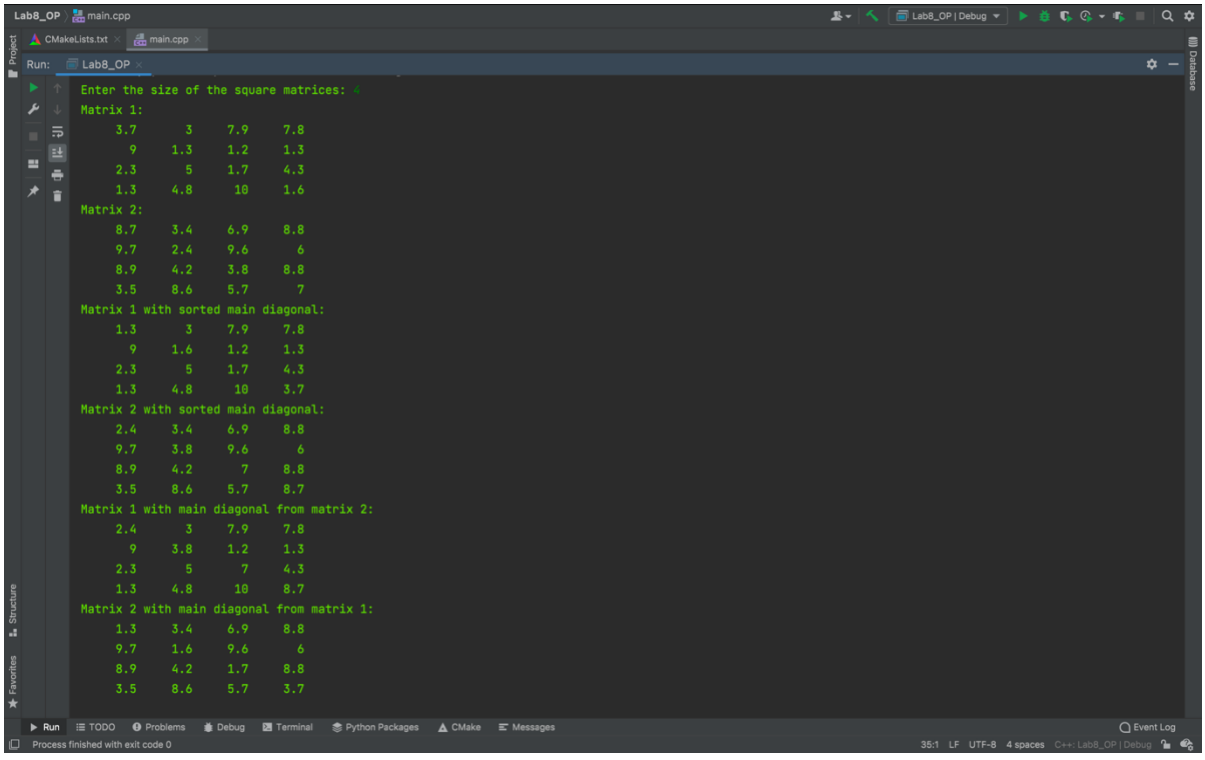
Програма на C++:

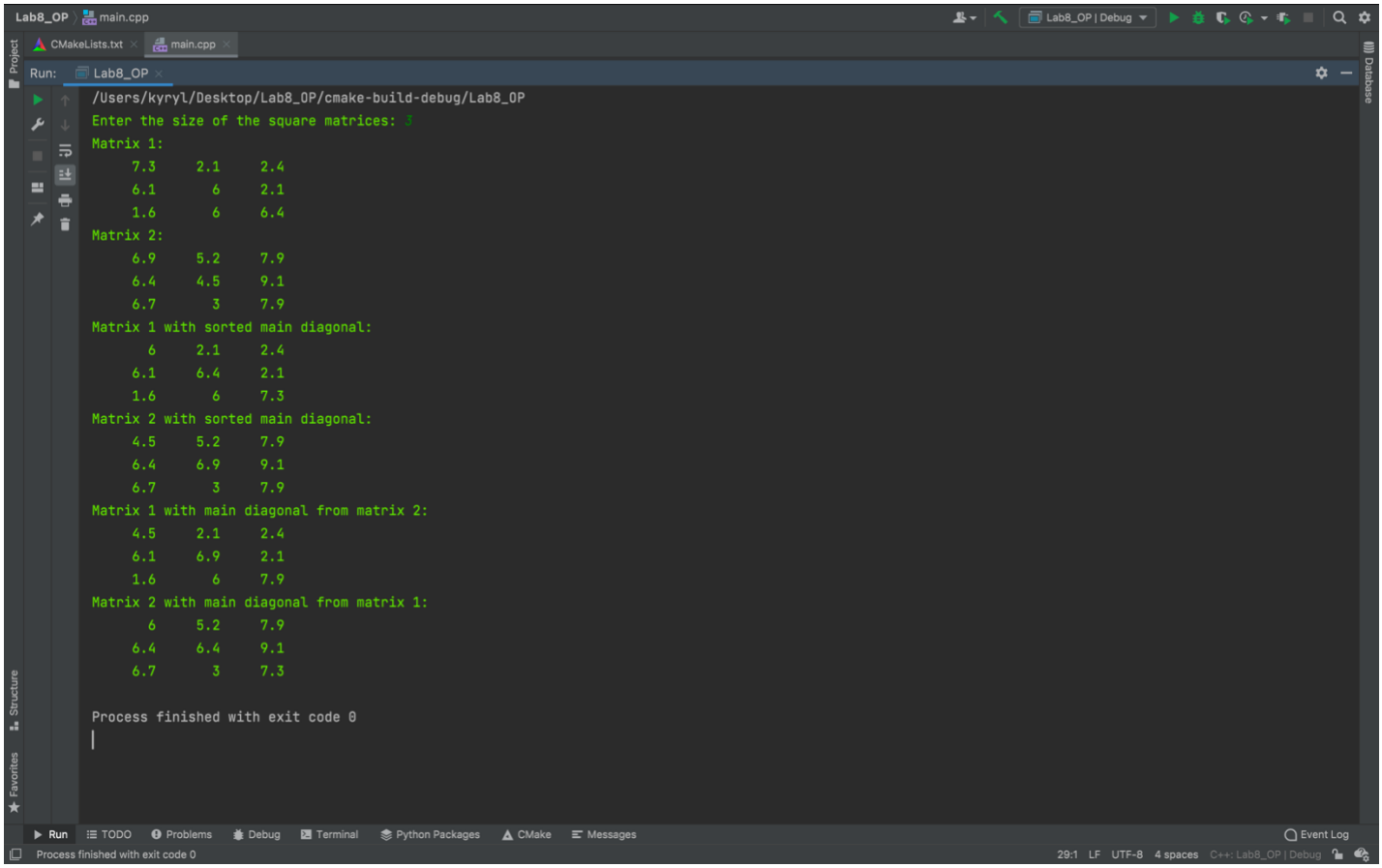
```
/* Варіант 28
У двох заданих дійсних квадратних матрицях розмірності n поміняти
місцями їх
 головні діагоналі, попередньо упрядкувавши за збільшенням
елементи цих діагоналей.*/
#include <iostream>
#include <ctime>
#include <iomanip>
using namespace std;
double random_double(int, int); // Функція для генерація
випадкового дісного числа з заданого проміжка
double** generate_matrix(int, int); // Фукнція для створення
матриці заданої розмірності
void display_matrix(double**, int); // Функція для вивдення
матриці
void sort_diagonal_asc(double**, int); // Функція для сортування
елементів головної діагоналі за збільшенням
void swap_diagonals(double**, double**, int); /* Функція для
заміни головної діагоналі однієї матриці відповідною
 діагоналлю іншої матриці*/
void delete_matrix(double**, int); // Функція для видалення
двомірного масиву з пам'яті

int main() {
    int n;
    double** matrix_1;
    double** matrix_2;
    srand(time(NULL));
    cout << "Enter the size of the square matrices: ";
    cin >> n;
    matrix_1 = generate_matrix(n, n);
    matrix_2 = generate_matrix(n, n);
    cout << "Matrix 1:\n";
```

```
    display_matrix(matrix_1, n);
    cout << "Matrix 2: \n";
    display_matrix(matrix_2, n);
    sort_diagonal_asc(matrix_1, n);
    sort_diagonal_asc(matrix_2, n);
    cout << "Matrix 1 with sorted main diagonal:\n";
    display_matrix(matrix_1, n);
    cout << "Matrix 2 with sorted main diagonal:\n";
    display_matrix(matrix_2, n);
    swap_diagonals(matrix_1, matrix_2, n);
    cout << "Matrix 1 with main diagonal from matrix 2:\n";
    display_matrix(matrix_1, n);
    cout << "Matrix 2 with main diagonal from matrix 1:\n";
    display_matrix(matrix_2, n);
    delete_matrix(matrix_1, n);
    delete_matrix(matrix_2, n);
    return 0;
}
double random_double(int min, int max) {
    double num;
    int rnd_of_max = rand()%(max + 1);
    double fraction = (double)(rnd_of_max) / max;
    num = min + rand()%(max - min) + fraction;
    return num;
}

double** generate_matrix(int rows, int columns) {
    double** matrix = new double*[rows];
    for (int i=0; i < rows; i++) {
        matrix[i] = new double[columns];
    }
    for (int i=0; i < rows; i++) {
        for (int j = 0; j < columns; j++) {
            matrix[i][j] = random_double(1, 10);
        }
    }
    return matrix;
}

void display_matrix(double** matrix, int size) {
    for (int i = 0; i < size; i++) {
        for (int j = 0; j < size; j++) {
            cout << setw(8) << matrix[i][j];
        }
        cout << '\n';
    }
}

void sort_diagonal_asc(double** matrix, int size) {
    int i;
    double temp_num;
    bool sorted, swapped;
    sorted = false;
    i = 0;
    while (i < size - 1 && !sorted){
        swapped = false;
```

```
        for (int j = 0; j < size - 1 - i; ++j) {
            if (matrix[j][j] > matrix[j+1][j+1]) {
                temp_num = matrix[j][j];
                matrix[j][j] = matrix[j+1][j+1];
                matrix[j+1][j+1] = temp_num;
                swapped = true;
            }
        }
        if (!swapped) {
            sorted = true;
        }
        i++;
    }
}

void swap_diagonals(double** matrix_1, double** matrix_2, int
size) {
    double temp_num;
    for (int i = 0; i < size; i++) {
        temp_num = matrix_1[i][i];
        matrix_1[i][i] = matrix_2[i][i];
        matrix_2[i][i] = temp_num;
    }
}

void delete_matrix(double** matrix, int size) {
    for (int i = 0; i < size; i++) {
        delete[] matrix[i];
    }
    delete[] matrix;
}
```

Результат на C++:





Висновок

Отже, я дослідив двовимірні масиви, опанував технологію їх використання та написав алгоритм для заміни головних діагоналей двох двовимірних масивів, попередньо створивши алгоритм для сортування їх елементів за збільшенням, та отримав коректний результат.