

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 ПІ-11 Сідак Кирил Ігорович
(шифр, прізвище, ім'я, по батькові)

Перевірів _____
(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 упряdkувавши за збільшенням елементи цих діагоналей.

Постановка задачі:

Спочатку треба створити два динамічні двовимірні масиви розмірності n на t 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 упорядкувавши за збільшенням елементи цих діагоналей.*/
#include <iostream>
#include <ctime>
#include <iomanip>
using namespace std;
double random_double(int, int); // Функція для генерація випадкового дійсного числа з заданого проміжка
double** generate_matrix(int, int); // Функція для створення матриці заданої розмірності
void display_matrix(double**, int); // Функція для виведення матриці
void sort_diagonal(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 двовимірного 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(matrix_1, n);
sort_diagonal(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(double** matrix, int size) {
    double temp_num;
    for (int i = 0; i < size - 1; i++) {
        for (int j = i + 1; j < size; ++j) {
            if (matrix[i][i] > matrix[j][j]) {
                temp_num = matrix[i][i];
                matrix[i][i] = matrix[j][j];
                matrix[j][j] = temp_num;
            }
        }
    }
}

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;
}

```

The screenshot shows a C++ program running in a debugger. The program prompts the user to enter the size of square matrices (4). It then displays two matrices, Matrix 1 and Matrix 2. Matrix 1 is:

4	6.5	8.3	2.5
6.5	8.1	5.9	8.2
6.2	7.7	9.7	9.9
2.9	8.8	4.8	8.5

Matrix 2 is:

5.8	3.3	1.3	1.5
8.8	9.6	3.9	2.3
3.4	3.6	4.7	7.3
4.4	4.7	4.1	6.1

The program then shows the sorted main diagonals for both matrices. Matrix 1 with sorted main diagonal:

4	6.5	8.3	2.5
6.5	8.1	5.9	8.2
6.2	7.7	8.5	9.9
2.9	8.8	4.8	9.7

Matrix 2 with sorted main diagonal:

4.7	3.3	1.3	1.5
8.8	5.8	3.9	2.3
3.4	3.6	6.1	7.3
4.4	4.7	4.1	9.6

Next, it shows the main diagonal from matrix 2 for matrix 1:

4.7	6.5	8.3	2.5
6.5	5.8	5.9	8.2
6.2	7.7	6.1	9.9
2.9	8.8	4.8	9.6

Finally, it shows the main diagonal from matrix 1 for matrix 2:

4	3.3	1.3	1.5
8.8	8.1	3.9	2.3
3.4	3.6	8.5	7.3
4.4	4.7	4.1	9.7

Результат на C++:

Висновок

Отже, я дослідив двовимірні масиви, опанував технологію їх використання та написав алгоритм для заміни головних діагоналей двох двовимірних масивів,

The screenshot shows a C++ program running in a debugger. The program prompts the user to enter the size of square matrices (3). It then displays two matrices, Matrix 1 and Matrix 2. Matrix 1 is:

1.8	3.4	5
5.4	7.1	7.8
5.5	6.3	6.4

Matrix 2 is:

3.9	4.7	5
1.6	3	4.6
9.7	7.9	9.1

The program then shows the sorted main diagonals for both matrices. Matrix 1 with sorted main diagonal:

1.8	3.4	5
5.4	6.4	7.8
5.5	6.3	7.1

Matrix 2 with sorted main diagonal:

3	4.7	5
1.6	3.9	4.6
9.7	7.9	9.1

Next, it shows the main diagonal from matrix 2 for matrix 1:

3	3.4	5
5.4	3.9	7.8
5.5	6.3	9.1

Finally, it shows the main diagonal from matrix 1 for matrix 2:

1.8	4.7	5
1.6	6.4	4.6
9.7	7.9	7.1

Process finished with exit code 0

попередньо створивши алгоритм для сортування їх елементів за збільшенням, та отримав коректний результат.