Міністерство освіти і науки України
Національний технічний університет України «Київський політехнічний
інститут імені І. Сікорського»
Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт
з лабораторної роботи № 8
«Алгоритми та структури даних-1.
Основи алгоритмізації»

«Дослідження алгоритмів пошуку та сортування»

Варіант 2

**Виконав студент**: ІП-13 Бабашев Олексій Дмитрович

**Перевірив**: Вечерковська Анастасія Сергіївна

Київ 2021

Лабораторна робота 8

Дослідження алгоритмів пошуку та сортування

**Мета** – дослідити алгоритми пошуку та сортування, набути практичних навичок використання цих алгоритмів під час складання програмних специфікацій.

Варіант 2

Розробити алгоритм та написати програму, яка складається з наступних дій: 1. Опису змінної індексованого типу (двовимірний масив) згідно з варіантом (табл. 1). 2. Ініціювання змінної, що описана в п.1 даного завдання. 3. Створення нової змінної індексованого типу (одновимірний масив) та її ініціювання значеннями, що обчислюються згідно з варіантом (табл. 1).

| | | | |
|---|---|---|---|
| **2** | 6 x 5 | Дійсний | Із середнього арифметичного значення елементів рядків двовимірного масиву. Відсортувати обміном за спаданням. |

# 1) Постановка задачі:

Згенерувати матрицю(двомірний масив) з випадкових дійсних значень. Обчислити середні арифметичні значення її рядків та сформувати з них масив, сформований масив відсортувати за спаданням.

# 2) Побудова математичної моделі:

Таблиця змінних:

| Змінна | Тип | Назва | Призначення |
|---|---|---|---|
| Двомірний масив | Дійсний | arr | Вихідне дане |
| Одномірний масив | Дійсний | arrRes | Вихідне дане |
| Константа кількість рядків рядків матриці то розмір результуючого масиву | Цілий | ROW | Проміжне дане |
| Константа кількість стовпців матриці | Цілий | COL | Проміжне дане |
| Параметр функції кількість рядків | Цілий | row | Проміжне дане |
| Параметр функції кількість стовпців | Цілий | col | Проміжне дане |
| Параметр функції двомірний масив | Дійсний | mat | Проміжне дане |
| Параметр функції одномірний масив | Дійсний | array | Проміжне дане |
| Середнє арифметичне елементів рядків матриці | Дійсний | res | Проміжне дане |
| Змінна для зміни змінних місцями у сортуванні бульбашки | Дійсний | temp | Проміжне дане |
| Лічильник | Цілий | i | Проміжне дане |
| Лічильник | Цілий | j | Проміжне дане |

Таблиця функцій:

| Назва | Синтаксис | Опис |
|---|---|---|
| matrix | matrix(mat[][5], row, col) | Ініціалізація матриці та заповнення рандомними елементами |
| unsortedArrRes | unsortedArrRes(mat[][5], array[], row, col) | Ініціалізація результуючого масиву та заповнення його середніми арифметичними значеннями рядків матриці |
| sortedArrRes | sortedArrRes(array[], row) | Сортування за спаданням результуючого масиву методом бульбашки |

Математичне формулювання задачі зводиться до ініціалізації матриці та заповнення рандомними значеннями, розрахунку середніх арифметичних значень рядків матриці та заповненнями ними одномірного результуючого масиву, сортування цього масиву методом бульбашки.

Розв'язання:

Крок 1. Визначимо основні дії.

Крок 2. Деталізуємо ініціалізацію матриці та заповнення рандомними значеннями.

Крок 3. Деталізуємо розрахунок середніх арифметичних значень рядків матриці та заповненнями ними одномірного результуючого масиву.

Крок 4. Деталізуємо сортування результуючого масиву методом бульбашки.

**3)Псевдокод:**

**Початок**

ROW = 6
COL = 5
***matrix*** (arr, ROW, COL)
***unsortedArrRes*** (arr, arrRes, ROW, COL)
***sortedArrRes*** (arrRes, ROW)
**Кінець**

**Підпрограми:**

***matrix*** (mat[][5], row, col)
    **Повторити** i = 0 , i<row, i++
        **Повторити** j = 0 , j<col, j++
            mat[i][j] = rand()
            Вивести mat[i][j]
        **Все повторити**
    **Все повторити**
**Кінець *matrix()***

***unsortedArrRes***(mat[][5], array[], row, col)
    res = 0
    **Повторити** i = 0 , i<row, i++
        **Повторити** j = 0 , j<col, j++
            res += mat[i][j];
        **Все повторити**
        res /= col;
        array[i] = res;
        Вивести array[i]
    **Все повторити**
**Кінець *unsortedArrRes()***

***sortedArrRes***(array[], row)

**Повторити** i = 0 , i<row, i++

**Повторити** j = 0 , j<col, j++

**Якщо** array[j] > array[j + 1]

temp = array[j + 1]
array[j + 1] = array[j]
array[j] = temp

**Все якщо**

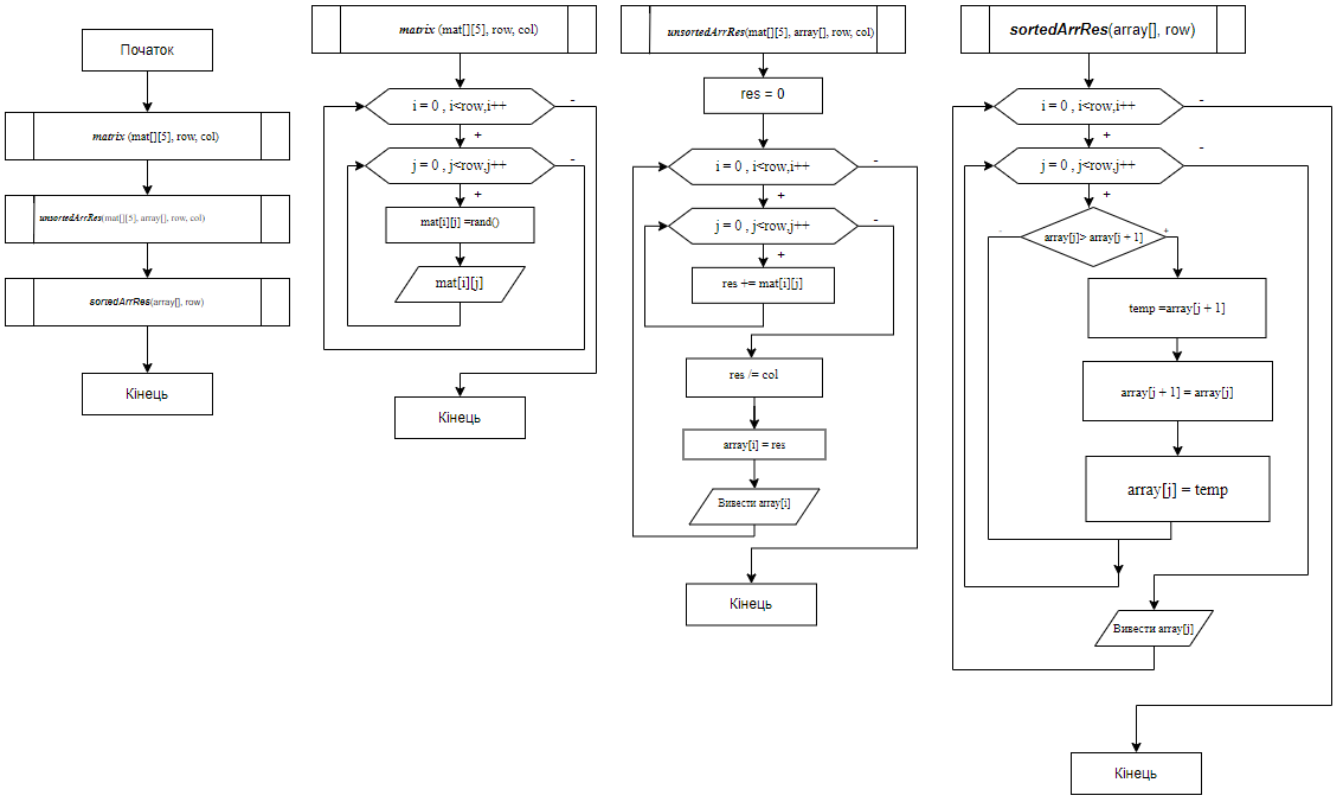
**Все повторити**

Вивести array[j]

**Все повторити**

**Кінець *sortedArrRes()***

**4)Блок-схема:**



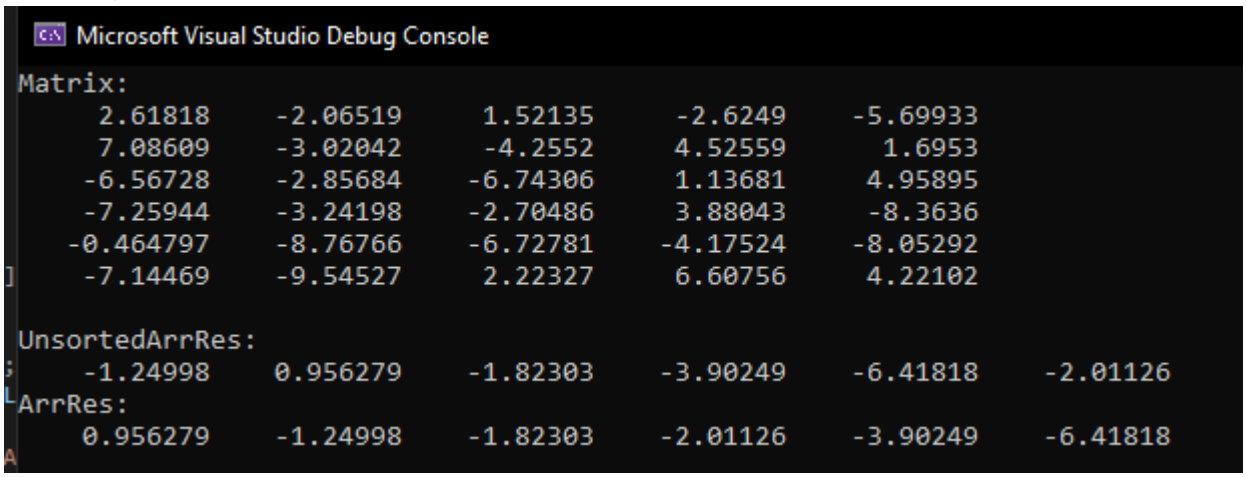

**5)Код c++:**

Project1

```cpp
#include <iostream>
#include <iomanip>


using namespace std;

void matrix(double[][5], int, int);
void unsortedArrRes(double[][5], double[], int, int);
void sortedArrRes(double[], int);

int main()
{

    srand(time(NULL));
    const int ROW = 6;
    const int COL = 5;

    double arrRes[ROW];
    double arr[ROW][COL];


    cout << "Matrix:\n";
    matrix(arr, ROW, COL);

    cout << "\nUnsortedArrRes:\n";
    unsortedArrRes(arr, arrRes, ROW, COL);

    cout << "\nArrRes:\n";
    sortedArrRes(arrRes, ROW);
    cout << endl;
}

void matrix(double mat[][5], int row, int col) {
    for (int i = 0; i < row; i++) {
        for (int j = 0; j < col; j++) {
            mat[i][j] = ((double)rand()) / RAND_MAX * 20 - 1 + 1 + (-10);
            cout << setw(12) << mat[i][j];
        }
        cout << endl;
    }
}

void unsortedArrRes(double mat[][5], double array[], int row, int col) {
    double res = 0;
    for (int i = 0; i < row; i++) {
        for (int j = 0; j < col; j++) {
            res += mat[i][j];
        }
        res /= col;
        array[i] = res;
        cout << setw(12) << array[i];
    }
}

void sortedArrRes(double array[], int row) {
    int i, j;
    double temp;
    for (i = 0; i < row; i++) {
        for (j = 0; j < row - i - 1; j++){
            if (array[j] > array[j + 1]) {
                temp = array[j + 1];
                array[j + 1] = array[j];
                array[j] = temp;
            }
        }
        cout << setw(12) << array[j];
    }
}
```

```cpp
#include <iostream>
#include <iomanip>


using namespace std;

void matrix(double[][5], int, int);
void unsortedArrRes(double[][5], double[], int, int);
void sortedArrRes(double[], int);

int main()
{

    srand(time(NULL));
    const int ROW = 6;
    const int COL = 5;

    double arrRes[ROW];
    double arr[ROW][COL];


    cout << "Matrix:\n";
    matrix(arr, ROW, COL);

    cout << "\nUnsortedArrRes:\n";
    unsortedArrRes(arr, arrRes, ROW, COL);

    cout << "\nArrRes:\n";
    sortedArrRes(arrRes, ROW);
    cout << endl;
}

void matrix(double mat[][5], int row, int col) {
    for (int i = 0; i < row; i++) {
        for (int j = 0; j < col; j++) {
            mat[i][j] = ((double)rand()) / RAND_MAX * 20 - 1 + 1 + (-10);
            cout << setw(12) << mat[i][j];
        }
        cout << endl;
    }
}

void unsortedArrRes(double mat[][5], double array[], int row, int col) {
    double res = 0;
    for (int i = 0; i < row; i++) {
        for (int j = 0; j < col; j++) {
            res += mat[i][j];
        }
        res /= col;
        array[i] = res;
        cout << setw(12) << array[i];
    }
}

void sortedArrRes(double array[], int row) {
    int i, j;
    double temp;
    for (i = 0; i < row; i++) {
        for (j = 0; j < row - i - 1; j++){
            if (array[j] > array[j + 1]) {
                temp = array[j + 1];
                array[j + 1] = array[j];
                array[j] = temp;
            }
        }
        cout << setw(12) << array[j];
    }
}
```

## 6)Тестування:




**7)Висновок:** дослідив алгоритми пошуку та сортування, набув практичних навичок використання цих алгоритмів під час складання програмних специфікацій.