

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(двовірний 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
Двовірний масив	Дійсний	arr	Вихідне дане
Одновірний масив	Дійсний	arrRes	Вихідне дане
Константа кількість 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 двовірний масив	Дійсний	mat	Проміжне дане
Параметр функції одновірний 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 випадковими 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 заповнення 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 заповнення 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**  $i = 0, i < \text{row}, i++$

**Повторити**  $j = 0, j < \text{col}, j++$

**Якщо**  $\text{array}[j] > \text{array}[j + 1]$

$\text{temp} = \text{array}[j + 1]$

$\text{array}[j + 1] = \text{array}[j]$

$\text{array}[j] = \text{temp}$

**Все якщо**

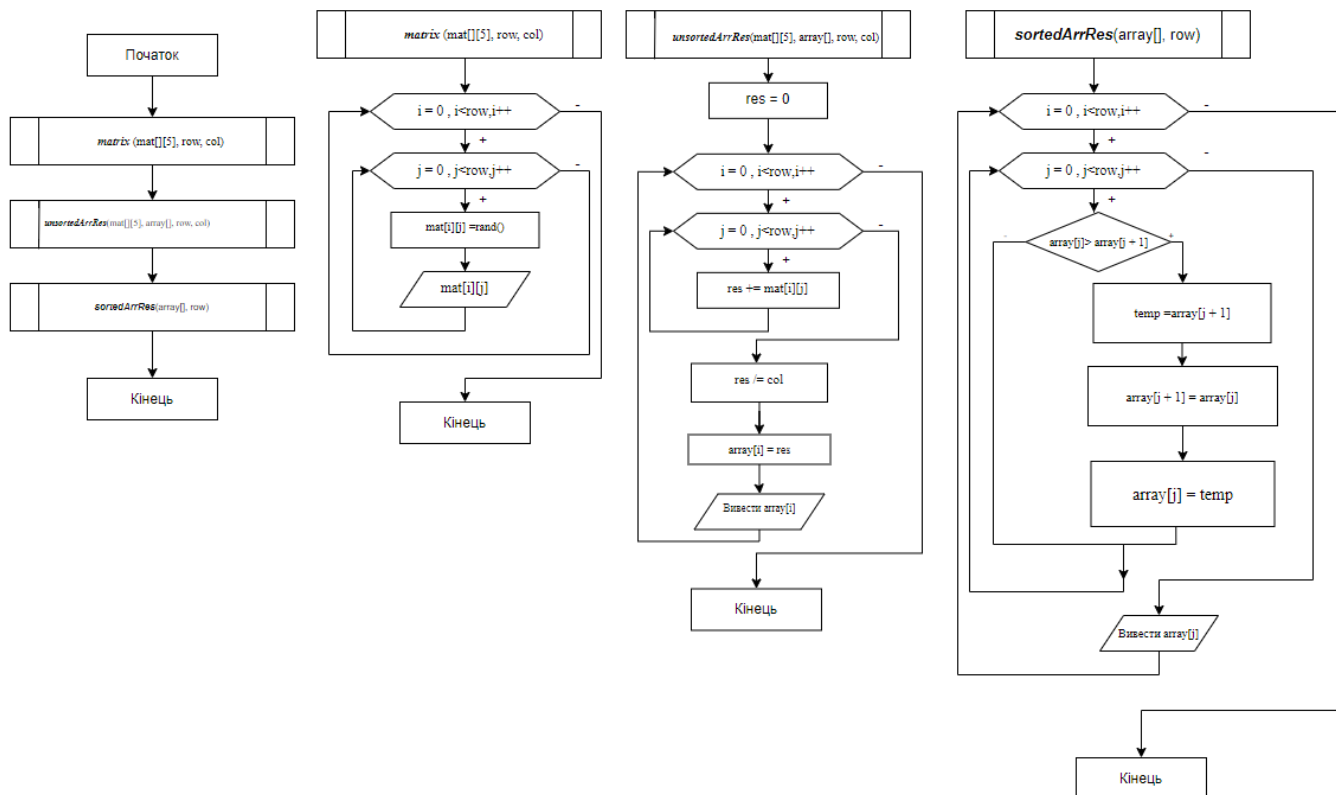
**Все повторити**

Вивести  $\text{array}[j]$

**Все повторити**

**Кінець** *sortedArrRes*()

#### 4)Блок-схема:



#### 5)Код c++:

## Project1

```
1  #include <iostream>
2  #include <iomanip>
3
4
5  using namespace std;
6
7  void matrix(double[][5], int, int);
8  void unsortedArrRes(double[][5], double[], int, int);
9  void sortedArrRes(double[], int);
10
11 int main()
12 {
13
14     srand(time(NULL));
15     const int ROW = 6;
16     const int COL = 5;
17
18     double arrRes[ROW];
19     double arr[ROW][COL];
20
21
22     cout << "Matrix:\n";
23     matrix(arr, ROW, COL);
24
25     cout << "\nUnsortedArrRes:\n";
26     unsortedArrRes(arr, arrRes, ROW, COL);
27
28     cout << "\nArrRes:\n";
29     sortedArrRes(arrRes, ROW);
30     cout << endl;
31 }
32
33 void matrix(double mat[][5], int row, int col) {
34     for (int i = 0; i < row; i++) {
35         for (int j = 0; j < col; j++) {
36             mat[i][j] = ((double)rand()) / RAND_MAX * 20 - 1 + 1 + (-10);
37             cout << setw(12) << mat[i][j];
38         }
39         cout << endl;
40     }
41 }
42
43 void unsortedArrRes(double mat[][5], double array[], int row, int col) {
44     double res = 0;
45     for (int i = 0; i < row; i++) {
46         for (int j = 0; j < col; j++) {
47             res += mat[i][j];
48         }
49         res /= col;
50         array[i] = res;
51         cout << setw(12) << array[i];
52     }
53 }
54
55 void sortedArrRes(double array[], int row) {
56     int i, j;
57     double temp;
58     for (i = 0; i < row; i++) {
59         for (j = 0; j < row - i - 1; j++){
60             if (array[j] > array[j + 1]) {
61                 temp = array[j + 1];
62                 array[j + 1] = array[j];
63                 array[j] = temp;
64             }
65         }
66         cout << setw(12) << array[j];
67     }
68 }
```

```

#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:

```
Microsoft Visual Studio Debug Console

Matrix:
    2.61818    -2.06519    1.52135    -2.6249    -5.69933
    7.08609    -3.02042    -4.2552    4.52559    1.6953
   -6.56728    -2.85684    -6.74306    1.13681    4.95895
   -7.25944    -3.24198    -2.70486    3.88043    -8.3636
  -0.464797    -8.76766    -6.72781    -4.17524    -8.05292
   -7.14469    -9.54527    2.22327    6.60756    4.22102

UnsortedArrRes:
   -1.24998    0.956279    -1.82303    -3.90249    -6.41818    -2.01126
ArrRes:
    0.956279    -1.24998    -1.82303    -2.01126    -3.90249    -6.41818
```

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