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
«Алгоритми та структури даних-1.
Основи алгоритмізації»

«Дослідження алгоритмів пошуку та сортування»

Варіант 6

Виконав студент ІП-13 Вдовиченко Станіслав Юрійович
(шифр, прізвище, ім'я, по батькові)

Перевірив Вєчерковська Анастасія Сергіївна
(прізвище, ім'я, по батькові)

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 6:

Розробити алгоритм та написати програму, яка складається з наступних дій:

1. Опису змінної індексованого типу (двовимірний масив) **розмірністю 7x5, тип даних - дійсний.**
2. Ініціювання змінної, що описана в п.1 даного завдання.
3. Створення нової змінної індексованого типу (одновимірний масив) та її ініціювання значеннями **із максимальних значень елементів рядків двовимірного масиву. Сортування методом бульбашки за спаданням.**

Постановка задачі: треба задати матрицю(двовимірний масив) розмірністю 7x5(7 рядків, 5 стовпчиків), заповнити її випадковими числами. За допомогою алгоритму пошуку заповнити третій масив максимальними значеннями кожного рядка матриці(так як рядків у матриці 7, то й розмірність одновимірного масиву буде також 7). Відсортувати отриманий масив методом бульбашки за спаданням.

Математична модель:

Змінна	**Тип**	**Ім’я**	**Призначення**
К-сть рядків матриці	Цілочисельний	rows	Вхідні дані
К-сть стовпців матриці	Цілочисельний	cols	Вхідні дані
Двовимірний масив	Дійсний	matrix	Проміжні дані
Одновимірний масив	Дійсний	array	Вихідні дані

Підпрограма **fillMatrix**

Змінна	**Тип**	**Ім’я**	**Призначення**
Двовимірний масив	Дійсний	matr	Вхідні дані
К-сть рядків	Цілочисельний	rs	Вхідні дані
К-сть стовпців	Цілочисельний	cl	Вхідні дані
Випадкове значення	Дійсний	random	Проміжні дані

Підпрограма **showMatrix**

Змінна	**Тип**	**Ім’я**	**Призначення**
Двовимірний масив	Дійсний	matr	Вхідні дані
К-сть рядків	Цілочисельний	rs	Вхідні дані
К-сть стовпців	Цілочисельний	cl	Вхідні дані

Підпрограма **fillArray**

Змінна	**Тип**	**Ім’я**	**Призначення**
Двовимірний масив	Дійсний	matr	Вхідні дані
Одновимірний масив	Дійсний	arr	Вхідні дані
К-сть рядків	Цілочисельний	rs	Вхідні дані
К-сть стовпців	Цілочисельний	cl	Вхідні дані
Максимальне значення	Дійсний	max	Проміжні дані

Підпрограма **showArray**

Змінна	**Тип**	**Ім’я**	**Призначення**
Одновимірний масив	Дійсний	arr	Вхідні дані
К-сть рядків	Цілочисельний	rs	Вхідні дані

Підпрограма **bubbleSort**

Змінна	**Тип**	**Ім’я**	**Призначення**
Одновимірний масив	Дійсний	arr	Вхідні дані
К-сть рядків	Цілочисельний	rs	Вхідні дані
Тимчасова змінна	Дійсний	temp	Проміжні дані

Таким чином математичне формулювання задачі зводиться до створення двовимірного масиву 7x5, заповнення його випадковими дійсними числами за допомогою підпрограми fillMatrix. Далі створюємо одновимірний масив, заповнюємо його максимальними значеннями рядків двовимірного масиву(використовуючи лінійний пошук: задаємо перше значення рядка як

максимальне значення, проходимо по рядку, і якщо якийсь елемент більший за максимум, то максимум замінюємо на значення цього елемента) за допомогою підпрограми fillArray. Розмір масиву буде визначатися кількістю рядків двовимірного масиву(в нашому випадку – 7). Далі отриманий масив відсортовуємо за допомогою методу бульбашки(підпрограма bubbleSort): у поданому наборі даних(в нашому випадку масив) порівнюються два сусідні елементи, якщо ж один з елементів є меншим за свого сусіда(сортування за спаданням), то ці два елементи міняються місцями(для цього в тілі підпрограми створюємо тимчасову змінну temp, яка буде приймати значення одного з елементів при обміні). Прохід продовжується доти, доки дані не будуть відсортовані. Складність алгоритму у найгіршому випадку – O(n^2).

Для генерування випадкових чисел використаємо функція rand(), яка буде генерувати дійсні числа з діапазону [-10;10].
Програмні специфікації запишемо у псевдокоді та графічній формі у вигляді блок-схеми.
Крок 1. Визначимо основні дії.
Крок 2. Ініціювання двовимірного масиву.
Крок 3. Ініціювання одновимірного масиву.
Крок 4. Заповнення двовимірного масиву, його вивід.
Крок 5. Заповнення одновимірного масиву, його вивід.
Крок 6. Сортування одновимірного масиву, його вивід.

Псевдокод
Процедура
fillMatrix(matr,rs,cl)

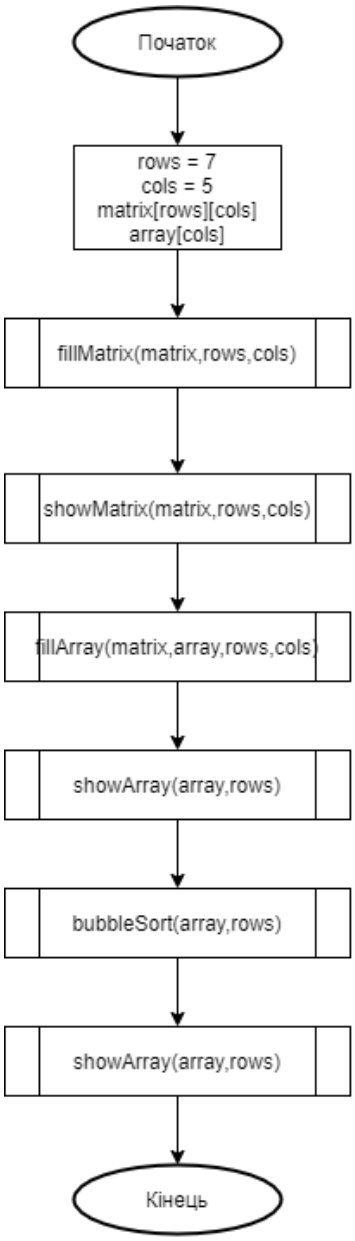
```
        повторити від 0 до rs з кроком 1
            повторити від 0 до cl з кроком 1
                matr[i][j] = rand(-10,10)
            все повторити
        все повторити
Все процедура

Процедура
    showMatrix(matr,rs,cl)
        повторити від 0 до rs з кроком 1
            повторити від 0 до cl з кроком 1
                Виведення matr[i][j]
            все повторити
        все повторити
Все процедура

Процедура
    fillArray(matr,arr,rs,cl)
        повторити від 0 до rs з кроком 1
            max = matr [i][0]
            повторити від 0 до cl з кроком 1
                якщо (matr[i][j] >= max)
                    то
                        max = matr[i][j]
                        arr[i]  = max
                все якщо
            все повторити
        все повторити
Все процедура

Процедура
    showArray(arr,rs)
        повторити від 0 до rs з кроком 1
            Виведення arr[i]
```

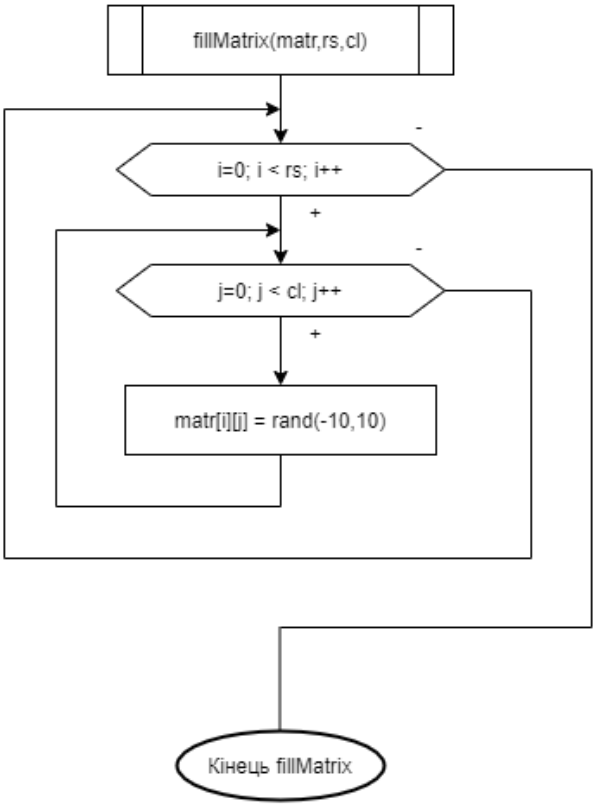
все повторити
Все процедура

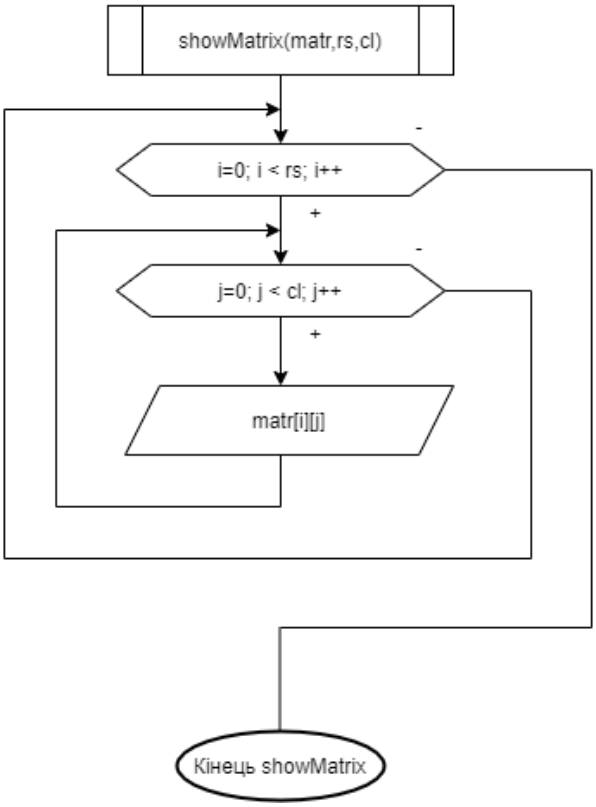
Процедура
bubbleSort(arr,rs)
повторити від 0 до rs з кроком 1
повторити від 0 до rs – 1 з кроком 1
якщо (arr[j] < arr[j+1])
то
temp = arr[j]
arr[j] = arr[j+1]
arr[j+1] = temp
все якщо
все повторити
все повторити
Все процедура

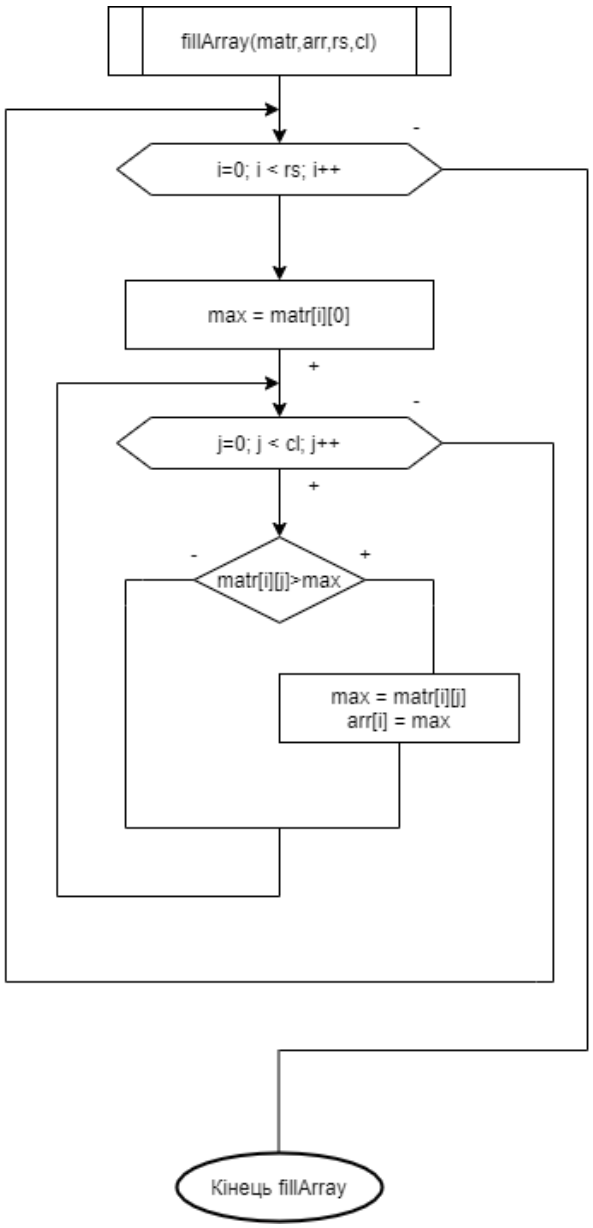
Початок
rows = 7
cols = 5
matrix[rows][cols]
array[rows]
fillMatrix(matrix, rows, cols)
showMatrix(matrix, rows, cols)
fillArray(matrix, array, rows, cols)
showArray(array, rows)
bubbleSort(array, rows)
showArray(array, rows)
Кінець

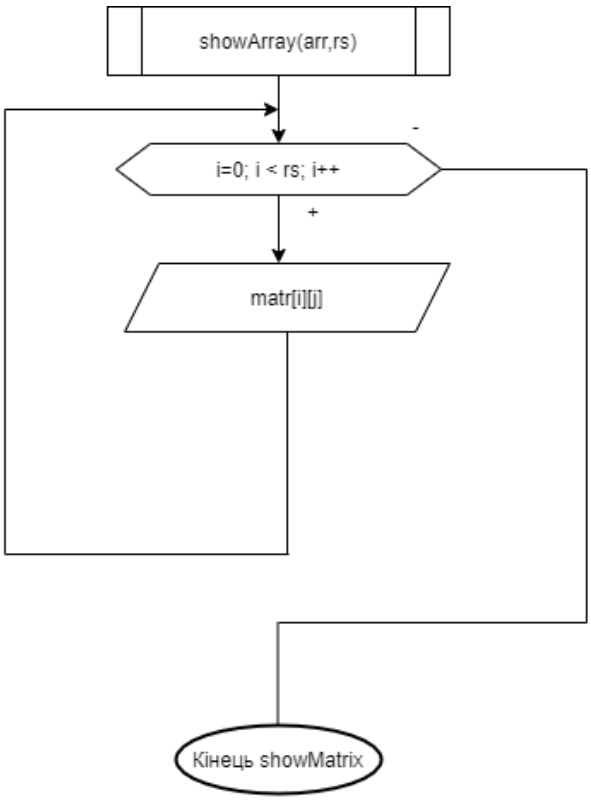
Блок-схема

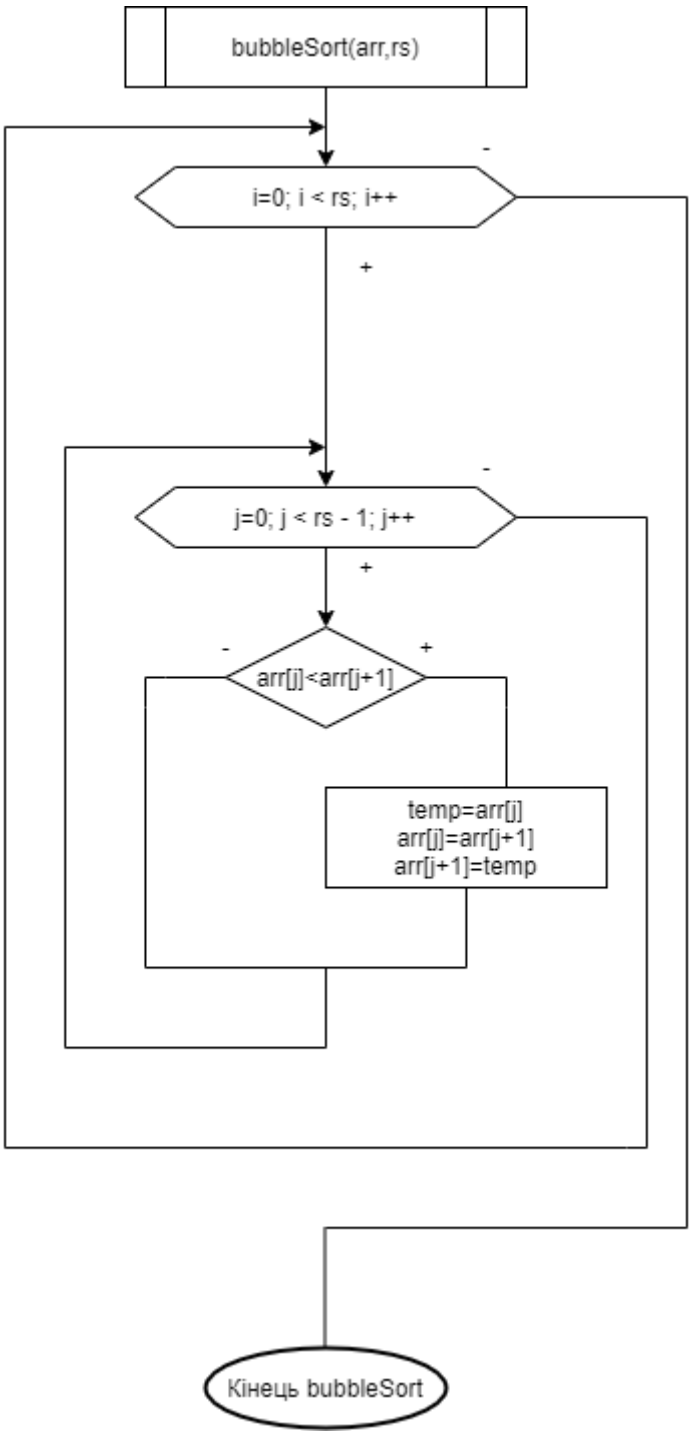

Початок
rows = 7
cols = 5
matrix[rows][cols]
array[cols]
fillMatrix(matrix,rows,cols)
showMatrix(matrix,rows,cols)
fillArray(matrix,array,rows,cols)
showArray(array,rows)
bubbleSort(array,rows)
showArray(array,rows)
Кінець


fillMatrix(matr,rs,cl)
i=0; i < rs; i++
-
+
j=0; j < cl; j++
-
+
matr[i][j] = rand(-10,10)
Кінець fillMatrix


showMatrix(matr,rs,cl)
i=0; i < rs; i++
-
+
j=0; j < cl; j++
-
+
matr[i][j]
Кінець showMatrix


fillArray(matr,arr,rs,cl)
i=0; i < rs; i++
-
max = matr[i][0]
+
j=0; j < cl; j++
-
+
matr[i][j]>max
-
+
max = matr[i][j]
arr[i] = max
Кінець fillArray


showArray(arr,rs)
i=0; i < rs; i++
-
+
matr[i][j]
Кінець showMatrix


bubbleSort(arr,rs)
i=0; i < rs; i++
-
+
j=0; j < rs - 1; j++
-
+
arr[j]<arr[j+1]
-
+
temp=arr[j]
arr[j]=arr[j+1]
arr[j+1]=temp
Кінець bubbleSort

Код програми

```cpp
#include <iostream>
#include <cstdlib>
#include <ctime>
#include <iomanip>
using namespace std;

void fillMatrix(double matr[7][5] ,int,int);
void showMatrix(double matr[7][5],int,int);
void fillArray(double matr[7][5], double arr[],int,int);
void showArray(double arr[],int);
void bubbleSort(double arr[],int);
int main() {
    srand( _Seed: time( _Time: NULL));
    const int ROWS = 7;
    const int COLS = 5;
    double matrix[ROWS][COLS];
    double array[ROWS]{};
    fillMatrix(matrix, ROWS, COLS);
    cout << "The matrix is: " << endl;
    showMatrix(matrix, ROWS, COLS);
    fillArray(matrix, array, ROWS, COLS);
    cout << "The array is: ";
    showArray(array,ROWS);
    bubbleSort(array,ROWS);
    cout << "Sorted array: ";
    showArray(array,ROWS);
    return 0;
}
```

```
void fillMatrix(double matr[7][5],int rs, int cl){
    double random;
    for (int i = 0; i < rs; ++i) {
        for (int j = 0; j < cl; ++j) {
            random = rand()%201-100;
            matr[i][j] = random / 10;
        }
    }
}
void showMatrix(double matr[7][5], int rs, int cl){
    for (int i = 0; i < rs; ++i) {
        for (int j = 0; j < cl; ++j) {
         cout << setw( n: 5) << matr[i][j];
        }
        cout << endl;
    }
}
void fillArray(double matr[7][5], double arr[], int rs, int cl){
    double max;
    for (int i = 0; i < rs; ++i) {
        max = matr[i][0];
        for (int j = 0; j < cl; ++j) {
            if(matr[i][j] >= max){
                max = matr[i][j];
                arr[i] = max;
            }
        }
    }
}
```

```
void showArray(double arr[], int rs){
    for (int i = 0; i < rs; ++i) {
        cout << setw( n: 5) << arr[i];
    }
    cout << endl;
}
void bubbleSort(double arr[], int rs) {
      double temp;
      for (int i = 0; i < rs; ++i) {
          for (int j = 0; j < rs - 1; ++j) {
              if(arr[j] < arr [j+1]) {
                  temp = arr[j];
                  arr[j] = arr[j + 1];
                  arr[j + 1] = temp;
              }
          }
      }
 }
```

Тестування програми

```
The matrix is:
 -8.2 -1.5  6.7 -3.6 -3.8
  2.9 -6.8 -7.6 -0.1   -2
   -8 -1.8  -10  0.1  9.8
  1.4 -2.5   -1  2.6 -1.2
  0.9 -4.8  6.6 -6.5 -2.3
 -7.4  8.5    4  7.8  1.4
  9.9 -1.4    6 -4.3  8.6
The array is:   6.7  2.9  9.8  2.6  6.6  8.5  9.9
Sorted array:   9.9  9.8  8.5  6.7  6.6  2.9  2.6

Process finished with exit code 0
```

```
The matrix is:
 -1.3 -5.6  8.9  -10  5.6
  6.5 -4.1 -3.3 -3.2 -7.3
    3 -9.8  3.5  9.6 -6.1
  4.3  6.7 -6.5  7.5  7.4
 -9.9 -1.5 -3.6  8.9  6.9
   -3  6.9  5.8  5.2  -10
 -0.6 -1.5 -8.9  9.2   -9
The array is:   8.9  6.5  9.6  7.5  8.9  6.9  9.2
Sorted array:   9.6  9.2  8.9  8.9  7.5  6.9  6.5

Process finished with exit code 0
```

```
The matrix is:
  6.8  7.8  0.6    0 -0.3
   -7 -2.7 -3.5  3.9  1.6
   -5 -0.2  6.9  3.1  9.9
  7.4  7.7  0.5  0.5 -2.5
    3  2.3  0.7 -4.4  7.2
  2.5  5.2  3.7 -0.2 -8.3
  3.5  9.6 -0.1 -2.9  8.8
The array is:   7.8  3.9  9.9  7.7  7.2  5.2  9.6
Sorted array:   9.9  9.6  7.8  7.7  7.2  5.2  3.9

Process finished with exit code 0
```

Висновки

Під час виконання даної лабораторної роботи я дослідив алгоритми пошуку та сортування, набув практичних навичок використання цих алгоритмів під час складання програмних специфікацій. Розробив алгоритм для розв’язання поставленої задачі, побудував математичну модель, псевдокод, блок-схему. Написав код програми, протестував алгоритм і переконався в його правильності.