Додаток 1

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 27

Виконав студент ІП-15, Пономаренко Маргарита Альбертівна
(шифр, прізвище, ім'я, по батькові)

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

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.

Індивідуальне завдання

Варіант 27

Завдання

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).

27	8 x 4	Дійсний	Із добутку від'ємних значень елементів рядків двовимірного масиву. Відсортувати обміном за зростанням.

Постановка задачі

Згенерувати двовимірний масив заданої розмірності 8 на 4 з довільними елементами дійсного типу. Для кожного рядка вирахувати добуток від'ємних значень елементів та створити з них одновимірний масив. У другому масиві відсортувати обміном за зростанням.

Побудова математичної моделі

Змінна	**Тип**	**Ім'я**	**Призначення**
Матриця	float	matrix	Проміжні дані
Масив	float	array	Проміжні дані/Результат
Кількість рядків матриці	int	row	Початкові дані
Кількість стовпців матриці	int	col	Початкові дані
Лічильник i	int	i	Проміжні дані
Лічильник j	int	j	Проміжні дані
Лічильник r	int	r	Проміжні дані
Лічильник n	int	n	Проміжні дані
Змінна, яка тимчасово тримає значення елемента	float	temp	Проміжні дані

У функції get_matrix() створюється масив 8 на 4 та заповнюється випадковими числами з діапазону (-10;10). Функція output_matrix() виводить двовимірний масив. В функції get_arr2() формується та виводиться другий масив. Він формується шляхом переобору елементів кожного рядка двовимірного масиву і знаходження добутку від'ємних чисел. Далі за допомогою функції sort_arr2() одновимірний масив сортується метод бульбашки за зростанням.

Розв'язання

Програмні специфікації запишемо у псевдокоді, графічній формі у вигляді блок-схеми та у вигляді коду.

Крок 1. Визначення основних дій.

Крок 2. Ініціалізація матриці

Крок 3. Ініціалізація масиву

Крок 4. Сортування та виведення масиву

Псевдокод

Крок 1

початок

row: = 8

col: = 4

Ініціалізація матриці та її виведення

Ініціалізація масиву

Сортування та виведення масиву

кінець

Крок 2

початок

row: = 8

col: = 4

get_matrix(const int row, const int col, int arr[8][4])

Ініціалізація масиву

Сортування та виведення масиву

кінець

Крок 3

початок

row: = 8

col: = 4

get_matrix(const int row, const int col, int arr[8][4])

get_arr2(const int row, const int col, int arr[8][4], int arr2[])

Сортування та виведення масиву

кінець

Крок 4

початок

row: = 8

col: = 4

get_matrix(const int row, const int col, int arr[8][4])

get_arr2(const int row, const int col, int arr[8][4], int arr2[])

sort_arr2(int arr2[])

кінець

Підпрограма 1:

get_matrix(const int row, const int col, int arr[8][4])

початок

повторити для i = 0, i < row, i++

повторити для j = 0, j < col, j++

arr[i][j] = rand() % 21 - 10

все повторити

все повторити

кінець

output_matrix(const int row, const int col, int arr[8][4])

початок

повторити для i = 0, i < row, i++

повторити для j = 0, j < col, j++

Виведення: arr[i][j]

все повторити

все повторити

кінець

Підпрограма 2:

get_arr2(const int row, const int col, int arr[8][4], int arr2[])

початок

k: = 0

mul: = 1

i: = 0

size: = 8

повторити поки i < row

повторити для j = 0, j < col, j++

якщо arr[i][j] < 0 **то**

k: = i

mul: = mul * arr[i][j]

arr2[k]: = mul

все якщо

якщо arr[i][j] >= 0 **то**

k: = i

arr2[k]: = mul

все якщо

все повторити

Виведення: arr2[k]

mul: = 1

i: = i + 1

кінець

Підпрограма 3:

void sort_arr2(int arr2[])

початок

size: = 8

повторити для n = 0; n < size - 1; n++

повторити для r = 0; r < size - n - 1; r++

якщо arr2[r] > arr2[r + 1] **то**

temp: = arr2[r]

arr2[r]: = arr2[r + 1]

arr2[r + 1]: = temp

все якщо

все повторити

все повторити

кінець

void output_arr2(int arr2[])

початок

size: = 8

повторити для r = 0; r < size; r++

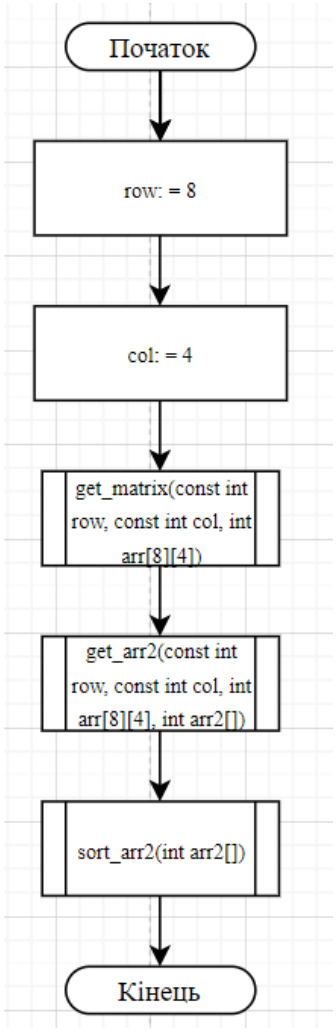
Виведення: arr2[r]

все повторити

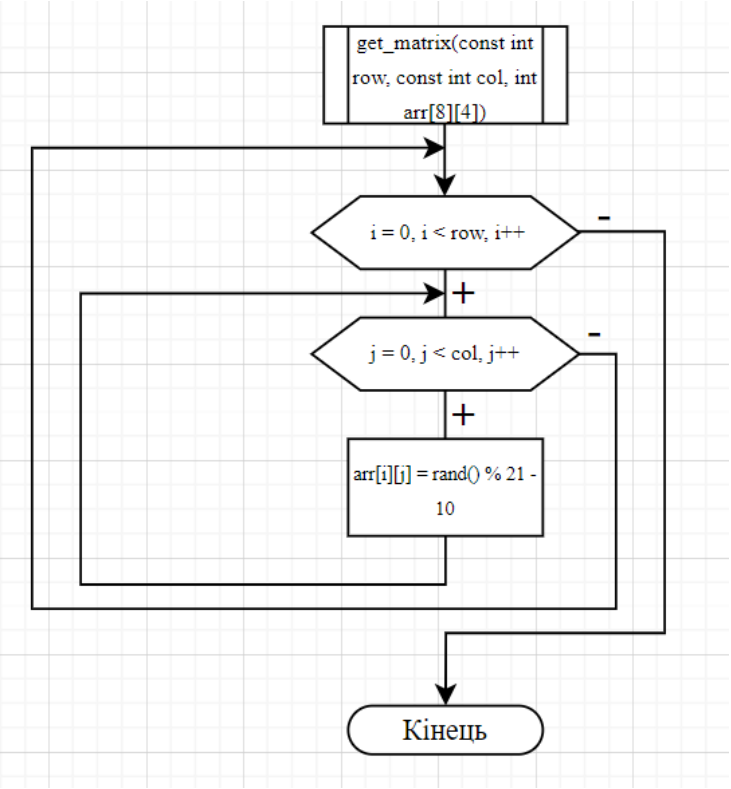
кінець

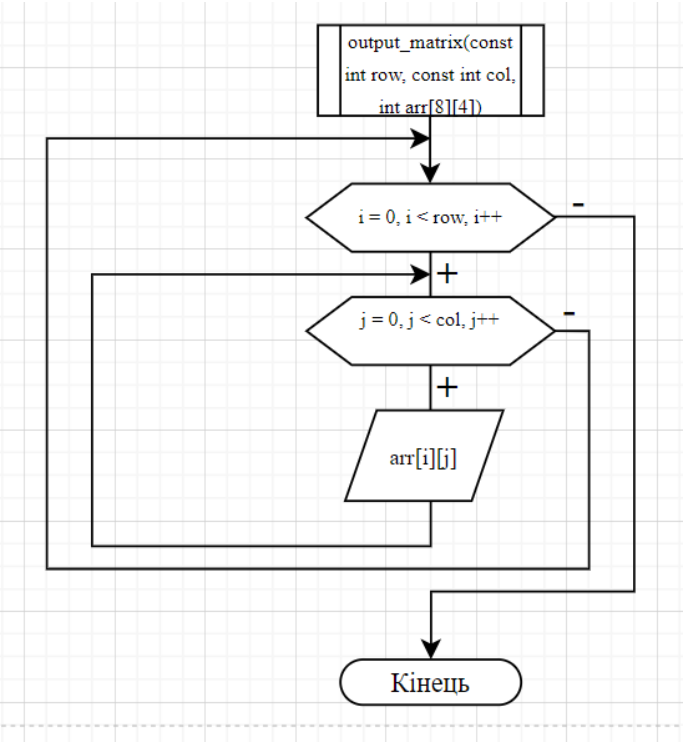
Блок схема

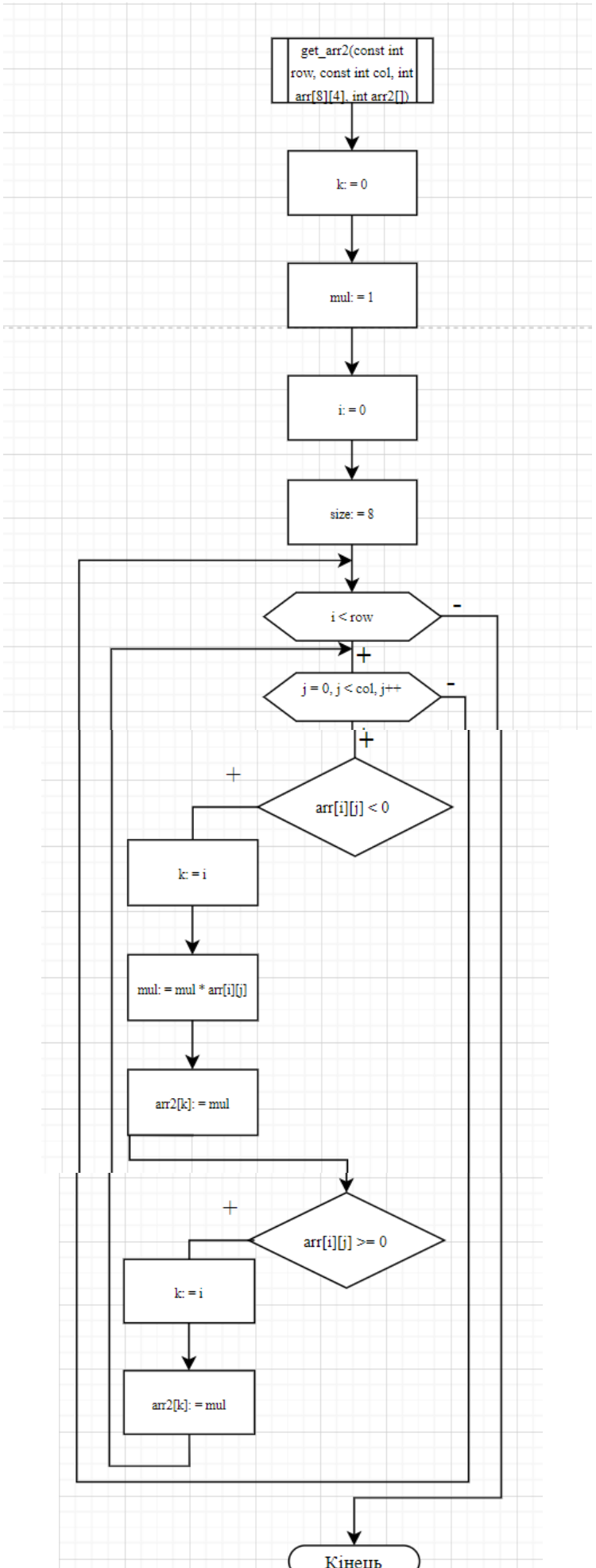
Основна програма:

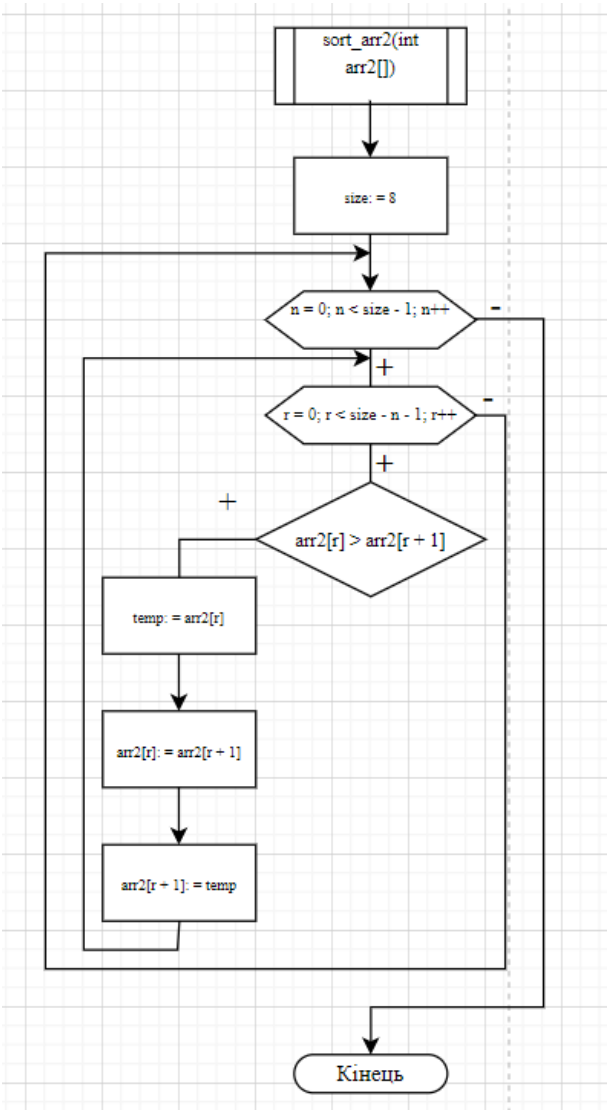


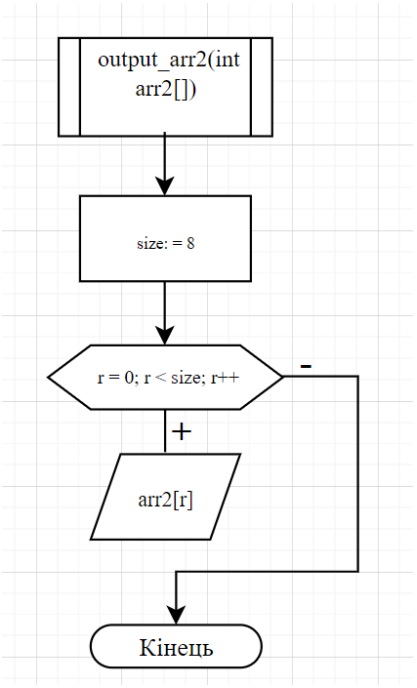
Підпрограми:


get_matrix(const int row, const int col, int arr[8][4])
i = 0, i < row, i++
-
+
j = 0, j < col, j++
-
+
arr[i][j] = rand() % 21 - 10
Кінець


output_matrix(const int row, const int col, int arr[8][4])
i = 0, i < row, i++
-
+
j = 0, j < col, j++
-
+
arr[i][j]
Кінець


get_arr2(const int row, const int col, int arr[8][4], int arr2[])
k: = 0
mul: = 1
i: = 0
size: = 8
i < row
-
+
j = 0, j < col, j++
-
+
+
arr[i][j] < 0
k: = i
mul: = mul * arr[i][j]
arr2[k]: = mul
+
arr[i][j] >= 0
k: = i
arr2[k]: = mul
Кінець


sort_arr2(int arr2[])
size: = 8
n = 0; n < size - 1; n++
-
+
r = 0; r < size - n - 1; r++
-
+
+
arr2[r] > arr2[r + 1]
temp: = arr2[r]
arr2[r]: = arr2[r + 1]
arr2[r + 1]: = temp
Кінець


output_arr2(int arr2[])
size: = 8
r = 0; r < size; r++
-
+
arr2[r]
Кінець

Код:

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

void get_matrix(const int row, const int col, int arr[8][4]) {
    for (int i = 0; i < row; i++) {
        for (int j = 0; j < col; j++) {
            arr[i][j] = rand() % 21 - 10;
        }
    }
}

void output_matrix(const int row, const int col, int arr[8][4]) {
    cout << "Matrix: " << endl;
    for (int i = 0; i < row; i++) {
        for (int j = 0; j < col; j++) {
            cout << setw(4) << arr[i][j] << " ";
        }
        cout << endl;
    }
}
```

```
23
24  void get_arr2(const int row, const int col, int arr[8][4], int arr2[]) {
25      int k = 0, mul = 1, i = 0, size = 8;
26      cout << endl << "Unsorted array: ";
27      while (i < row) {
28          for (int j = 0; j < col; j++) {
29              if (arr[i][j] < 0) {
30                  k = i;
31                  mul = mul * arr[i][j];
32                  arr2[k] = mul;
33              }
34              if (arr[i][j] >= 0) {
35                  k = i;
36                  arr2[k] = mul;
37              }
38          }
39          cout << arr2[k] << " ";
40          mul = 1;
41          i = i + 1;
42      }
43  }
44
```

```
45  void sort_arr2(int arr2[]) {
46      int size = 8;
47      for (int n = 0; n < size - 1; n++)
48      {
49          for (int r = 0; r < size - n - 1; r++)
50          {
51              if (arr2[r] > arr2[r + 1])
52              {
53                  int temp = arr2[r];
54                  arr2[r] = arr2[r + 1];
55                  arr2[r + 1] = temp;
56              }
57          }
58      }
59  }
60
```

```
61  void output_arr2(int arr2[]) {
62      cout << endl << "Sorted array: ";
63      int size = 8;
64      for (int r = 0; r < size; r++) {
65          cout << arr2[r] << " ";
66      }
67      cout << endl;
68  }
69
70  int main() {
71      const int row = 8, col = 4;
72      int arr[row][col];
73      int arr2[8];
74      get_matrix(row, col, arr);
75      output_matrix(row, col, arr);
76      get_arr2(row, col, arr, arr2);
77      sort_arr2(arr2);
78      output_arr2(arr2);
79  }
```

Тестування:

```
Matrix:
  -9   -6   -1    9
  -2    0    0   -1
   5    0   -8    9
  10   -6   10   -3
  -7    5    6    6
   7    4    2   -1
  -8   -5   -5    3
  -9    9   -5  -10

Unsorted array: -54 2 -8 18 -7 -1 -200 -450
Sorted array: -450 -200 -54 -8 -7 -1 2 18
```

Висновки

Ми дослідили методи послідовного пошуку у впорядкованих і невпорядкованих послідовностях та набули практичних навичок їх використання під час складання програмних специфікацій. В результаті було отримано алгоритм, що ініціалізує матрицю випадковими дійсними числами, а також ініціалізує масив за даною умовою та сортує цей масив.