

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 23

Виконав студент ПІ-13 Недельчев Євген Олександрович  
(шифр, прізвище, ім'я, по батькові)

Перевірила Вечерковська Анастасія Сергіївна  
(прізвище, ім'я, по батькові)

Київ 20211

## 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 23

№ варіанта	Розмірність	Тип даних	Обчислення значень елементів одновимірного масиву
23	8 x 4	Дійсний	Із добутку від'ємних значень елементів рядків двовимірного масиву. Відсортувати методом вставки за зростанням.

#### Постановка задачі

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

1. Опису змінної індексованого типу (двовимірний масив) згідно з варіантом.
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).

#### Побудова математичної моделі

<i>Змінна</i>	<i>Тип</i>	<i>Ім'я</i>	<i>Призначення</i>
Кількість рядків у двовимірному масиві	Цілий	ROWS	Константа
Кількість стовбців у двовимірному масиві	Цілий	COLUMNS	Константа
Перша змінна індексованого типу(двовимірний масив) розмірністю ROWSxCOLUMNS	Дійсний	array1[ROWS][COLUMNS]	Початкові дані
Друга змінна індексованого типу(одновимірний масив) з 8 символних значень	Дійсний	array2[ROWS]	Результат
Лічильник у циклах	Цілий	i	Лічильник
Лічильник у вкладеному циклі	Цілий	j	Лічильник
Добуток від'ємних значень елементів рядків двовимірного масиву array1	Дійсний	product_of_negative_elements	Проміжний результат

Кількість від'ємних елементів у рядку масива	Цілий	amount_of_negative_elements	Проміжний результат
Тимчасова змінна для обміну значень при сортуванні	Дійсний	temp	Буферна змінна

Таким чином математичне формулювання задачі зводиться до створення одновимірного масиву з ROWS елементів (оскільки рядків у двовимірному масиві ROWS) з добутку від'ємних елементів рядків двовимірного масиву. У випадку, якщо від'ємних елементів у рядку немає, добуток дорівнює нулю. Після ініціалізації одновимірного масиву відсортувати його за зростанням методом вставки.

Для сортування створимо функцію SortArray(double\* arr, int n), яка в якості параметрів приймає посилання на масив arr дійсних чисел та кількість елементів n цього масиву. Оскільки функція лише змінює переданий їй масив та не повертає жодних значень, то її тип – void.

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

### Розв'язання

Крок 1. Визначимо основні дії.

Крок 2. Ініціалізація двовимірного масиву array1

Крок 3. Ініціалізація одновимірного масиву array2 добутками від'ємних значень елементів рядків двовимірного масиву array1.

Крок 4. Сортування масиву array2 методом вставки за зростанням.

## Псевдокод

**функція** SortArray(double\* arr, int n)

**повторити** для i від 1 до n

        temp = arr[i]

        j = i - 1

**повторити поки** j >= 0 && arr[j] > temp

            arr[j+1] = arr[j]

            j--

**все повторити**

        arr[j+1] = temp

**все повторити**

**все функція**

**початок**

    ROWS = 8

    COLUMNS = 4

    array1[ROWS][COLUMNS]

**повторити** для i від 0 до ROWS

**повторити** для j від 0 до COLUMNS

            введення array1[i][j]

**все повторити**

**все повторити**

    array2[ROWS]

**повторити** для i від 0 до ROWS

        amount\_of\_negative\_elements = 0

        product\_of\_negative\_elements = 1

**повторити** для j від 0 до COLUMNS

**якщо** array1[i][j] < 0

                product\_of\_negative\_elements \*= array1[i][j]

                amount\_of\_negative\_elements++

**все якщо**

**все повторити**

**якщо** !amount\_of\_negative\_elements

            array2[i] = 0

**все якщо**

**інакше**

            array2[i] = product\_of\_negative\_elements

**все інакше**

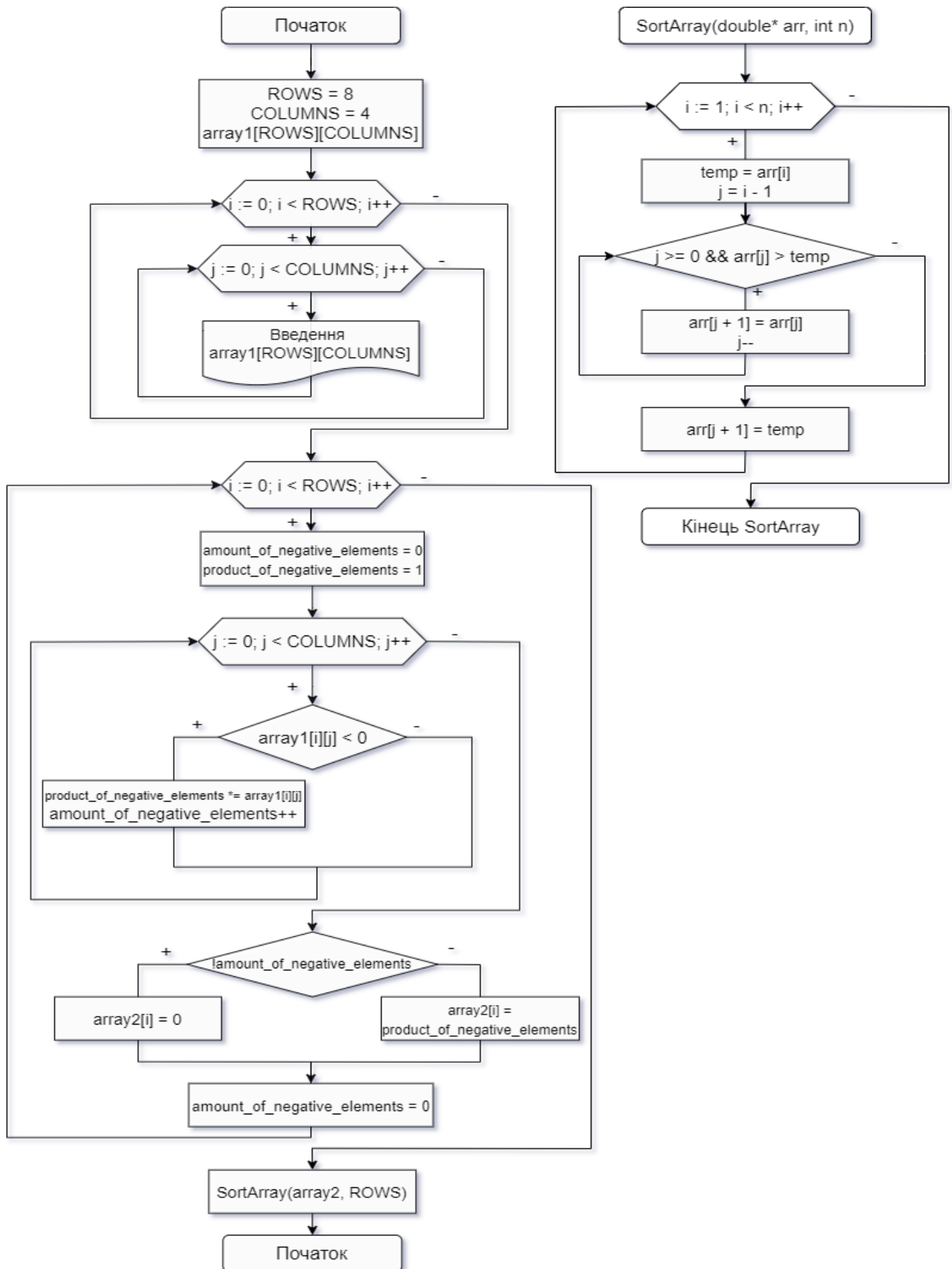
        amount\_of\_negative\_elements = 0

**все повторити**

    SortArray(array2, ROWS)

**кінець**

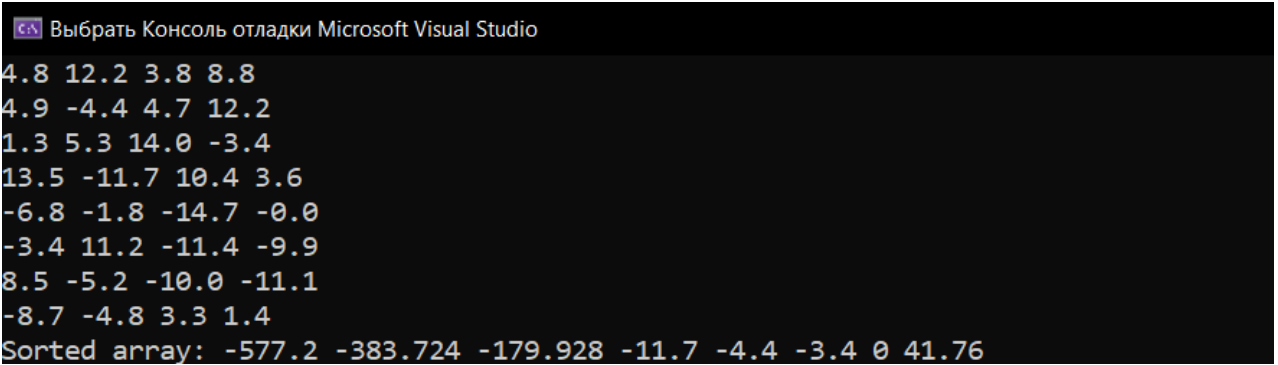
## Блок-схема



## Код програми

```
1  #include <iostream>
2
3  void SortArray(double* arr, int n) {
4      int j; double temp;
5      for (int i = 1; i < n; i++) {
6          temp = arr[i];
7          j = i - 1;
8          while (j >= 0 && arr[j] > temp) {
9              arr[j + 1] = arr[j];
10             j--;
11         }
12         arr[j + 1] = temp;
13     }
14 }
15
16 int main() {
17     const int ROWS = 8;
18     const int COLUMNS = 4;
19     double array1[ROWS][COLUMNS];
20     for (int i = 0; i < ROWS; i++) {
21         for (int j = 0; j < COLUMNS; j++) {
22             std::cin >> array1[i][j];
23         }
24     }
25     double array2[ROWS];
26     for (int i = 0; i < ROWS; i++) {
27         int amount_of_negative_elements = 0;
28         double product_of_negative_elements = 1;
29         for (int j = 0; j < COLUMNS; j++) {
30             if (array1[i][j] < 0) {
31                 product_of_negative_elements *= array1[i][j];
32                 amount_of_negative_elements++;
33             }
34         }
35         if (!amount_of_negative_elements) array2[i] = 0;
36         else array2[i] = product_of_negative_elements;
37     }
38     SortArray(array2, ROWS);
39     std::cout << "Sorted array: ";
40     for (int i = 0; i < ROWS; i++) {
41         std::cout << array2[i] << ' ';
42     }
43 }
```

## Тестування алгоритму



```
Выбрать Консоль отладки Microsoft Visual Studio
4.8 12.2 3.8 8.8
4.9 -4.4 4.7 12.2
1.3 5.3 14.0 -3.4
13.5 -11.7 10.4 3.6
-6.8 -1.8 -14.7 -0.0
-3.4 11.2 -11.4 -9.9
8.5 -5.2 -10.0 -11.1
-8.7 -4.8 3.3 1.4
Sorted array: -577.2 -383.724 -179.928 -11.7 -4.4 -3.4 0 41.76
```

## Висновки

Під час виконання цієї лабораторної роботи я дослідив алгоритми пошуку та сортування, набув практичних навичок використання цих алгоритмів під час складання програмних специфікацій.