

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 № 9 з дисципліни

«Алгоритми та структури даних-1. Основи алгоритмізації»

«Дослідження алгоритмів обходу масивів»

Варіант 27

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

Перевірів \_\_\_\_\_  
( прізвище, ім'я, по батькові)

Київ 2021

## Лабораторна робота 9

### Дослідження алгоритмів обходу масивів

**Мета** - дослідити алгоритми обходу масивів, 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. Обчислення змінної, що описана в п.1, згідно з варіантом (табл. 1).

27	Задано матрицю дійсних чисел $A[m,n]$ . При обході матриці по рядках знайти в ній перший максимальний елемент $X$ і його місцезнаходження. Обміняти знайдене значення $X$ з елементом останнього стовбця.
----	---

#### Постановка задачі

Згенерувати двовимірний масив, розмірність якого задає користувач, з довільними елементами дійсного типу. Далі потрібно виконати обхід матриці по рядках змійкою. В результаті обходу матриці потрібно знайти перший максимальний елемент і обміняти його місцями з елементом останнього стовбця.

#### Побудова математичної моделі

Змінна	Тип	Ім'я	Призначення
Матриця	double	arr1	Проміжні дані/Результат
Кількість рядків матриці	int	m	Проміжні дані
Кількість стовпців матриці	int	n	Проміжні дані
Індекси рядків елементів	int	ind1	Проміжні дані
Індекси стовпців елементів	int	ind2	Проміжні дані
Перший максимальний елемент	double	max	Результат
Лічильник i	int	i	Проміжні дані
Лічильник j	int	j	Проміжні дані
Змінна, яка тимчасово тримає значення елемента	double	temp	Проміжні дані

У функції `getInit()` ініціалізується матриця та заповнюється випадковими числами. Далі за допомогою функції `outputArr()` виводиться сформований масив, щоб можна було відслідкувати роботу програми. Функція `getMax()` знаходить перший максимальний елемент і назначає змінним індекси положення елемента в матриці. В функції `swarArr()` відбувається обхід матриці по рядкам. Далі ще один раз за допомогою функції `outputArr()` виводиться масив, але уже з внесеними змінами. І за допомогою масиву `deleteArr()` видаляється масив.

## **Розв'язання**

Програмні специфікації запишемо у псевдокодi, графічній формi у вигляді блок-схеми та у вигляді коду.

Крок 1. Визначення основних дій.

Крок 2. Ініціалізація матриці.

Крок 3. Визначення першого максимального елемента обходом матриці.

Крок 4. Зміна місцями максимального елемента та елемента з останнього стовпця.

Крок 5. Видалення масиву.

## **Псевдокод**

*Крок 5*

**початок**

введення m

введення n

ind1: = m

ind2: = n

getInit(double, int, int)

getMax(double, int&, int&)

swapArr(double, int, int, int, int)

outputArr(double, int, int, int, int)

deleteArr(double, int)

**кінець**

***Підпрограма 1:***

getInit(double, int, int)

**початок**

**повторити** для  $i = 0, i < m, i++$

arr1[i]: = new double[n]

**повторити** для  $j = 0, j < n, j++$

arr1[i][j] = round(((double)rand() / 1600 - 10) \* 100) / 100

**все повторити**

**все повторити**

**кінець**

***Підпрограма 2:***

getMax(double, int&, int&)

**початок**

max: = arr1[0][0]

**повторити** для  $i = 0, i < m, i++$

**повторити** для  $j = 0, j < n, j++$

**якщо** max < arr1[i][j] **то**

max: = arr1[i][j]

ind1: = i

ind2: = j

**все якщо**

**все повторити**

**все повторити**

m: = ind1

n: = ind2

**кінець**

### ***Підпрограма 3:***

swapArr(double, int, int, int, int)

**початок**

temp: = arr1[m - 1][n - 1]

arr1[m - 1][n - 1]: = arr1[ind1][ind2]

arr1[ind1][ind2]: = temp

**кінець**

### ***Підпрограма 4:***

outputArr(double, int, int, int, int)

**початок**

**повторити** для i = 0, i < m, i++

**повторити** для j = 0, j < n, j++

**якщо** arr1[i][j] == arr1[m - 1][n - 1] || arr1[i][j] ==  
arr1[ind1][ind2] **то**

Виведення: arr1[i][j]

**інакше**

Виведення: arr1[i][j]

**все якщо**

**все повторити**

**все повторити**

**кінець**

### ***Підпрограма 5:***

deleteArr(double, int)

**початок**

**повторити** для  $i = 0, i < m, i++$

Видалення: []arr1[i]

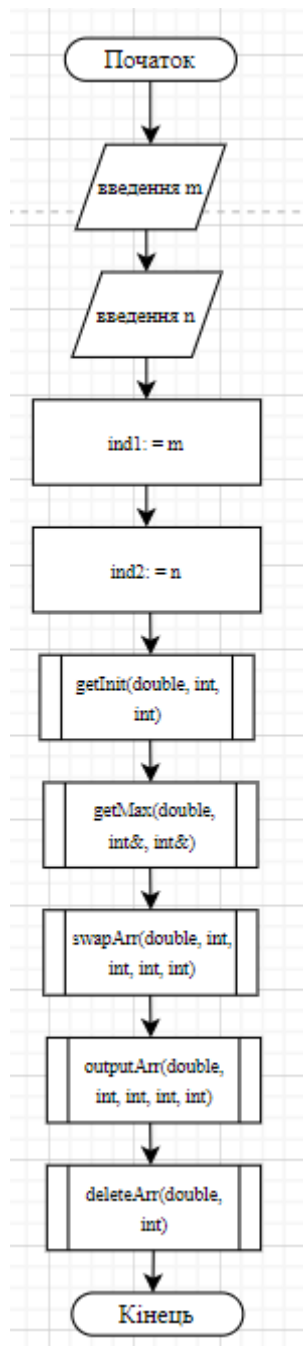
**все повторити**

Видалення: []arr1;

**кінець**

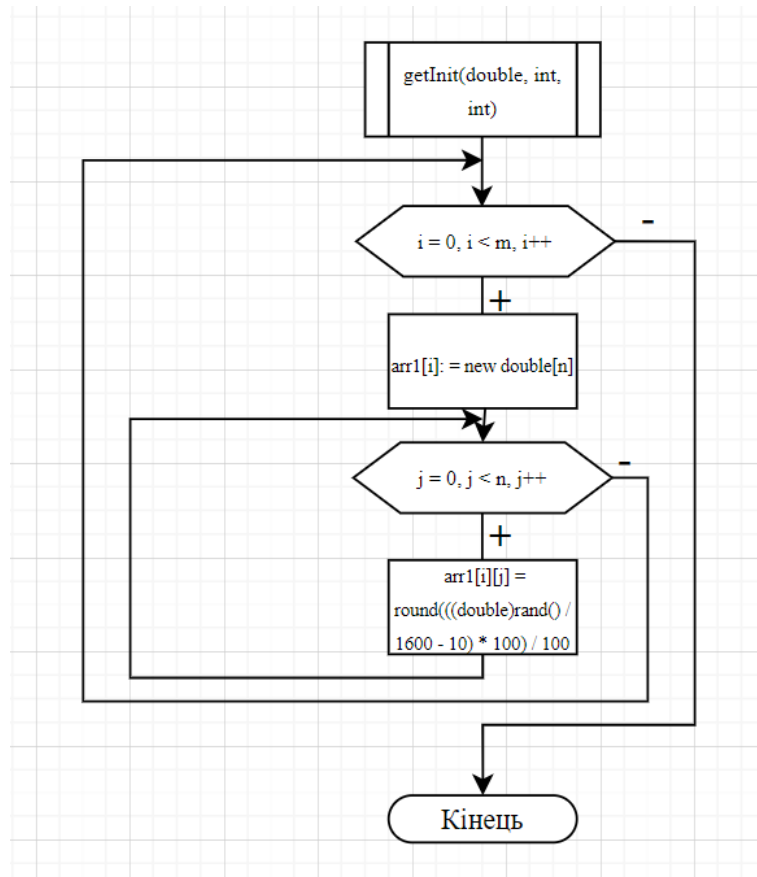
**Блок схема**

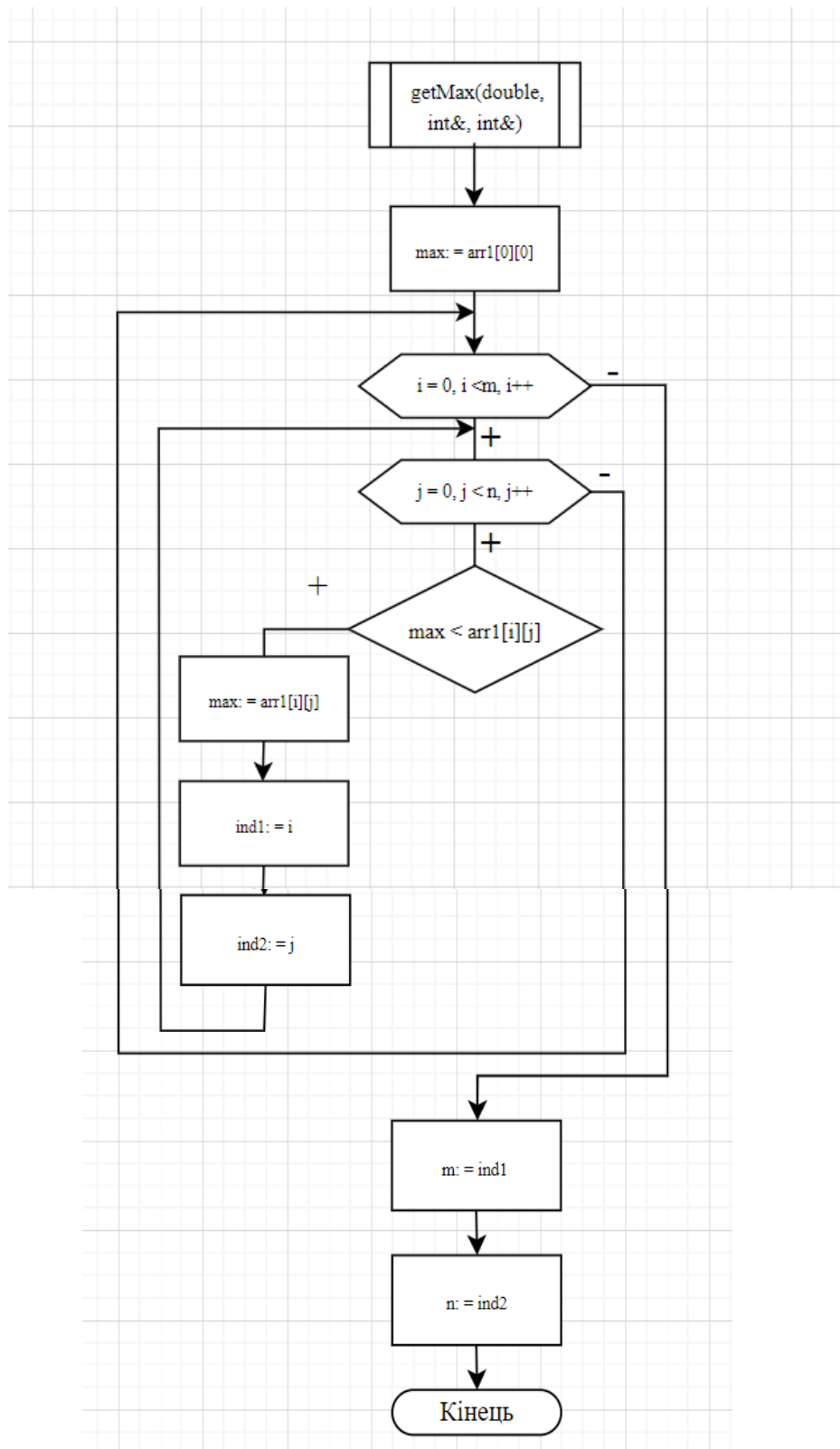
***Основна програма:***

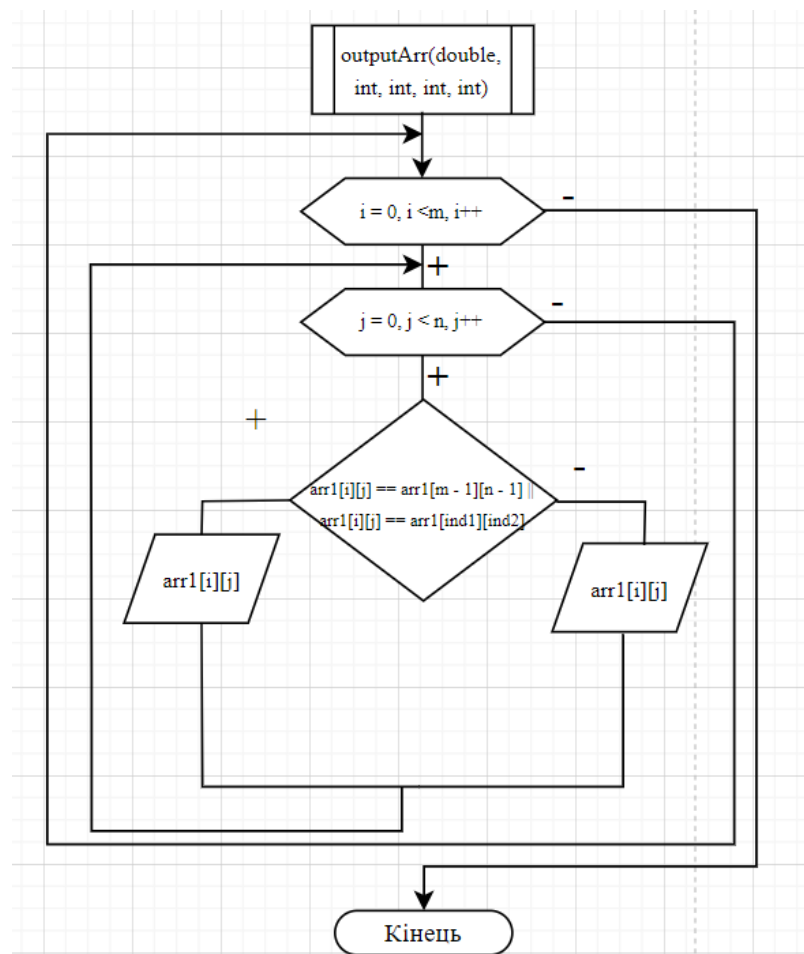
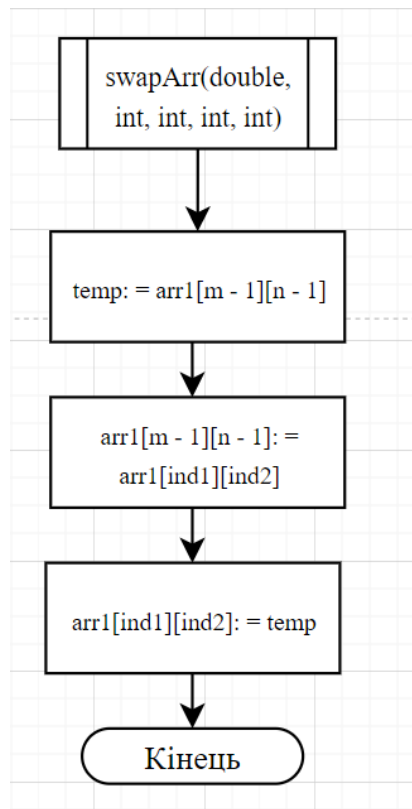


**Підпрограми:**









**Kod:**

```
ASD-Lab9 (Глобальная об.  
1  #include <iostream>  
2  #include <string>  
3  using namespace std;  
4  
5  void getInit(double, int, int);  
6  double getMax(double, int&, int&);  
7  void swapArr(double, int, int, int, int);  
8  void outputArr(double, int, int, int, int);  
9  void deleteArr(double, int);  
10  
11  int main() {  
12  
13      srand(time(NULL));  
14      int m, n;  
15      cout << "Input row number:" << endl;  
16      cin >> m;  
17      cout << "Input col number:" << endl;  
18      cin >> n;  
19      int ind1 = m, ind2 = n;  
20      double arr1 = new double* [m];  
21      getInit(arr1, m, n);  
22      cout << "Matrix:\n";  
23      outputArr(double, int, int, int, int);  
24      double max = getMax(arr1, ind1, ind2);  
25      cout << "\nMax element: " << max << "\nFinal matrix:\n";  
26      swapArr(arr1, m, n, ind1, ind2);  
27      outputArr(double, int, int, int, int);  
28      deleteArr(arr1, m);  
29  }
```

```

30
31 void getInit(double** arr1, int m, int n) {
32     for (int i = 0; i < m; i++) {
33         arr1[i] = new double[n];
34         for (int j = 0; j < n; j++) {
35             arr1[i][j] = round(((double)rand() / 1600 - 10) * 100) / 100;
36         }
37     }
38 }
39
40 double getMax(double arr1, int& m, int& n) {
41     double max;
42     max = arr1[0][0];
43     int ind1, ind2;
44     for (int i = 0; i < m; i++) {
45         for (int j = 0; j < n; j++) {
46             if (max < arr1[i][j]) {
47                 max = arr1[i][j];
48                 ind1 = i;
49                 ind2 = j;
50             }
51         }
52     }
53     m = ind1;
54     n = ind2;
55     return max;
56 }

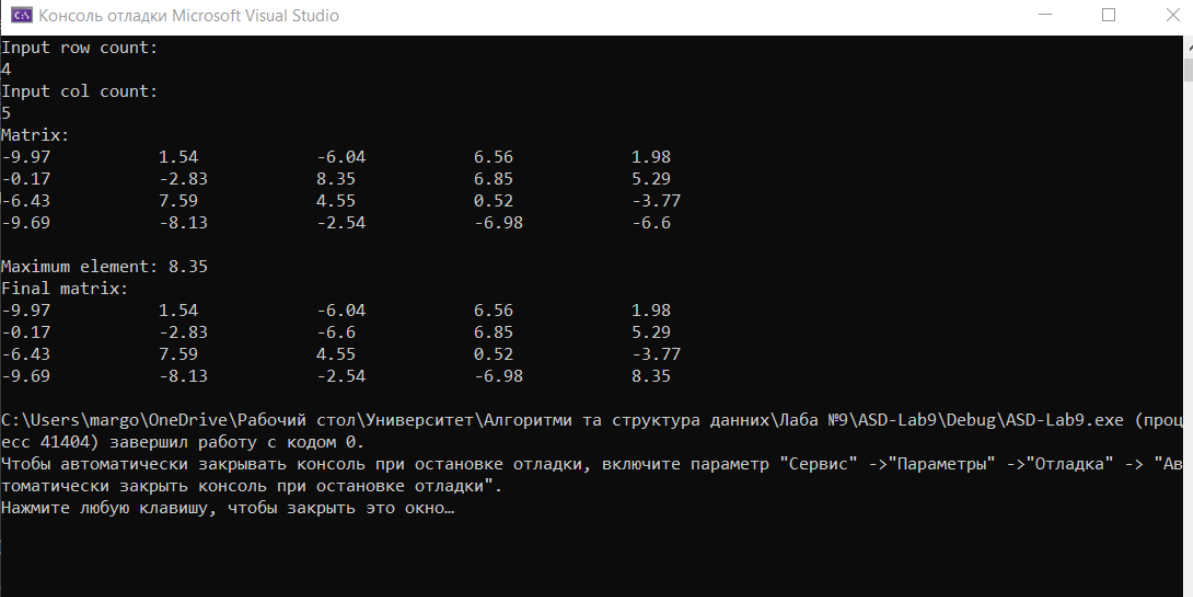
```

```

58 void swapArr(double** arr1, int m, int n, int ind1, int ind2) {
59     double temp;
60     temp = arr1[m - 1][n - 1];
61     arr1[m - 1][n - 1] = arr1[ind1][ind2];
62     arr1[ind1][ind2] = temp;
63 }
64
65 void outputArr(double** arr1, int m, int n, int ind1, int ind2) {
66     for (int i = 0; i < m; i++) {
67         for (int j = 0; j < n; j++) {
68             if (arr1[i][j] == arr1[m - 1][n - 1] || arr1[i][j] == arr1[ind1][ind2]) {
69                 cout << arr1[i][j] << "\t\t";
70             }
71             else {
72                 cout << arr1[i][j] << "\t\t";
73             }
74         }
75         cout << "\n";
76     }
77 }
78
79 void deleteArr(double** arr1, int m) {
80     for (int i = 0; i < m; i++) {
81         delete[] arr1[i];
82     }
83     delete[] arr1;
84 }

```

### Тестування:



```
Консоль отладки Microsoft Visual Studio
Input row count:
4
Input col count:
5
Matrix:
-9.97      1.54      -6.04      6.56      1.98
-0.17      -2.83      8.35      6.85      5.29
-6.43      7.59      4.55      0.52      -3.77
-9.69      -8.13      -2.54      -6.98      -6.6

Maximum element: 8.35
Final matrix:
-9.97      1.54      -6.04      6.56      1.98
-0.17      -2.83      -6.6      6.85      5.29
-6.43      7.59      4.55      0.52      -3.77
-9.69      -8.13      -2.54      -6.98      8.35

C:\Users\margo\OneDrive\Рабочий стол\Университет\Алгоритми та структура даних\Лаба №9\ASD-Lab9\Debug\ASD-Lab9.exe (процесс 41404) завершил работу с кодом 0.
Чтобы автоматически закрывать консоль при остановке отладки, включите параметр "Сервис" -> "Параметры" -> "Отладка" -> "Автоматически закрыть консоль при остановке отладки".
Нажмите любую клавишу, чтобы закрыть это окно...
```

### Висновки

Ми дослідити алгоритми обходу масивів, набути практичних навичок використання цих алгоритмів під час складання програмних специфікацій. В результаті було отримано алгоритм, що генерує двовимірний масив, розмірність якого задає користувач, з довільними елементами дійсного типу. Далі виконує обхід матриці по рядках змійкою. В результаті обходу матриці знаходить перший максимальний елемент і обмінює його місцями з елементом останнього стовбця.