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 лінійних алгоритмів ”
Варіант:12

Виконав студент: ІП-12 Єльчанінов Артем Юрійович
(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.

Варіант 12

Задача: Обчислити значення елементів одновимірного масиву із максимальних значень елементів рядків матриці. Відсортувати методом вставки за спаданням.

Розмір матриці: 6 x 4

Постановка задачі

Результатом розв’язку задачі є відсортований одновимірний масив методом вставки за спаданням.

Спершу заповнюємо матрицю випадково згенерованими значеннями. Далі слідує дія заповнення одновимірного масиву, вона відбується визначенням максимального значення в рядку матриці, і передачі цього значення в одновимірний масив. Останньою дією є сортування одновимірного масиву заповненого максимальними елементами рядків матриці за спаданням методом вставки. Після цього задача буде виконана.

Математична модель

Змінна	Тип	Ім’я	Призначення
Двовимірний масив (матриця) розміру 6 x 4	Цілий	matrix[6][4]	Проміжне дане
Одновимірний масив розміру 6	Цілий	array[6]	Вихідне дане
Лічильник для арифметичних цилів	Цілий	i	Проміжне дане
Додатковий лічильник для арифметичного циклу	Цілий	j	Проміжне дане
Фунція для виведення матриці	Відсутній (void)	ouput_matrix	Допоміжний алгоритм
Тимчасова зміння для зберігання значення максимального елемента в рядку матриці	Цілий	max_in_line	Проміжне дане
Функція для виведення елементів одновимірного масиву	Відсутній (void)	output_array	Допоміжний алгоритм

Тимчасова змінна для зберігання значення елемента сортованого масиву	Цілий	temp	Проміжне дане

Для генерації випадкових чисел будемо застосовувати функцію **rand( )**

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

Крок 1. Визначимо основні дії.

Крок 2. Деталізуємо дію заповнення матриці випадковими елементами.

Крок 3. Деталізуємо дію заповнення одновимірного масиву максимальними елементами рядків матриці.

Крок 4. Деталізуємо дію сортування елементів одновимірного масиву за спаданням за методом вставки.

Псевдокод алгоритму

Крок 1:

Початок

Перебір рядків матриці
Перебір стовпичків матриці
Заповнення матриці випадковими елементами
Виведення матриці
Перебір індексів елементів одновимірного масиву
Заповнення одновимірного масиву
Виведення одновимірного масиву
Перебір елементів одновимірного масиву
Сортування за спаданням одновимірного масиву
Виведення відсортованого одновимірного масиву

Кінець

Крок 2:

Початок
  для i **від** 0 **до** 6 **з кроком** 1
    Перебір стовпичків матриці
    Заповнення матриці випадковими елементами
  все повторити
  Виведення матриці
  Перебір індексів елементів одновимірного масиву
  Заповнення одновимірного масиву
  Виведення одновимірного масиву
  Перебір елементів одновимірного масиву
  Сортування за спаданням одновимірного масиву
  Виведення відсортованого одновимірного масиву
Кінець

Крок 3:

Початок
  для i **від** 0 **до** 6 **з кроком** 1
    для j **від** 0 **до** 4 **з кроком** 1
      Заповнення матриці випадковими елементами
    все повторити
  все повторити
  Виведення матриці
  Перебір індексів елементів одновимірного масиву
  Заповнення одновимірного масиву
  Виведення одновимірного масиву
  Перебір елементів одновимірного масиву
  Сортування за спаданням одновимірного масиву
  Виведення відсортованого одновимірного масиву
Кінець

Крок 4:

Початок
  для i **від** 0 **до** 6 **з кроком** 1
    для j **від** 0 **до** 4 **з кроком** 1
      matrix[i][j] := rand() % 100
    все повторити
  все повторити
  Виведення матриці
  Перебір індексів елементів одновимірного масиву
  Заповнення одновимірного масиву
  Виведення одновимірного масиву
  Перебір елементів одновимірного масиву
  Сортування за спаданням одновимірного масиву
  Виведення відсортованого одновимірного масиву
Кінець

Крок 5:

Початок
  для i **від** 0 **до** 6 **з кроком** 1
    для j **від** 0 **до** 4 **з кроком** 1
      matrix[i][j] := rand() % 100
    все повторити
  все повторити
  ouput_matrix(matrix)
  Перебір індексів елементів одновимірного масиву
  Заповнення одновимірного масиву
  Виведення одновимірного масиву
  Перебір елементів одновимірного масиву
  Сортування за спаданням одновимірного масиву
  Виведення відсортованого одновимірного масиву
Кінець

Крок 6:

Початок
  для i **від** 0 **до** 6 **з кроком** 1
    для j **від** 0 **до** 4 **з кроком** 1
      matrix[i][j] := rand() % 100
    все повторити
  все повторити
  ouput_matrix(matrix)
  для i **від** 0 **до** 6 **з кроком** 1
    Заповнення одновимірного масиву
  все повторити
  Виведення одновимірного масиву
  Перебір елементів одновимірного масиву
  Сортування за спаданням одновимірного масиву
  Виведення відсортованого одновимірного масиву
Кінець

Крок 7:

Початок
  для i **від** 0 **до** 6 **з кроком** 1
    для j **від** 0 **до** 4 **з кроком** 1
      matrix[i][j] := rand() % 100
    все повторити
  все повторити
  ouput_matrix(matrix)
  для i **від** 0 **до** 6 **з кроком** 1
    max_in_line := 0
    для j **від** 0 **до** 4 **з кроком** 1
      якщо max_in_line < matrix[i][j]
        то
          max_in_line := matrix[i][j]
      все якщо
    все повтрити
    array[i] := max_in_line
  все повторити
  Виведення одновимірного масиву
  Перебір елементів одновимірного масиву
  Сортування за спаданням одновимірного масиву
  Виведення відсортованого одновимірного масиву
Кінець

Крок 8:

Початок
  для i **від** 0 **до** 6 **з кроком** 1
    для j **від** 0 **до** 4 **з кроком** 1
      matrix[i][j] := rand() % 100
    все повторити
  все повторити
  ouput_matrix(matrix)
  для i **від** 0 **до** 6 **з кроком** 1
    max_in_line := 0
    для j **від** 0 **до** 4 **з кроком** 1
      якщо max_in_line < matrix[i][j]
        то
          max_in_line := matrix[i][j]
      все якщо
    все повтрити
    array[i] := max_in_line
  все повторити
  output_array(array)
  Перебір елементів одновимірного масиву
  Сортування за спаданням одновимірного масиву
  Виведення відсортованого одновимірного масиву
Кінець

Крок 9:

Початок
  для i **від** 0 **до** 6 **з кроком** 1
    для j **від** 0 **до** 4 **з кроком** 1
      matrix[i][j] := rand() % 100
    все повторити
  все повторити
  ouput_matrix(matrix)
  для i **від** 0 **до** 6 **з кроком** 1
    max_in_line := 0
    для j **від** 0 **до** 4 **з кроком** 1
      якщо max_in_line < matrix[i][j]
        то
          max_in_line := matrix[i][j]
      все якщо
    все повтрити
    array[i] := max_in_line
  все повторити
  output_array(array)
  для i **від** 1 **до** 6 **з кроком** 1
    Сортування за спаданням одновимірного масиву
  все повторити
  Виведення відсортованого одновимірного масиву
Кінець

Крок 10:

Початок
  для i **від** 0 **до** 6 **з кроком** 1
    для j **від** 0 **до** 4 **з кроком** 1
      matrix[i][j] := rand() % 100
    все повторити
  все повторити
  ouput_matrix(matrix)
  для i **від** 0 **до** 6 **з кроком** 1
    max_in_line := 0
    для j **від** 0 **до** 4 **з кроком** 1
      якщо max_in_line < matrix[i][j]
        то
          max_in_line := matrix[i][j]
      все якщо
    все повтрити
    array[i] := max_in_line
  все повторити
  output_array(array)
  для i **від** 1 **до** 6 **з кроком** 1
    temp := array[i]
    j := i - 1
    поки j >= 0 && array[j] < temp
      повторити
        array[j+1] := array[j]
        array[j] := temp
        j--
      все повторити
  все повторити
  Виведення відсортованого одновимірного масиву
Кінець

Крок 11:

```
Початок
  для i від 0 до 6 з кроком 1
    для j від 0 до 4 з кроком 1
      matrix[i][j] := rand() % 100
    все повторити
  все повторити
  ouput_matrix(matrix)
  для i від 0 до 6 з кроком 1
    max_in_line := 0
    для j від 0 до 4 з кроком 1
      якщо  max_in_line < matrix[i][j]
        то
          max_in_line := matrix[i][j]
      все якщо
    все повтрити
    array[i] := max_in_line
  все повторити
  output_array(array)
  для i від 1 до 6 з кроком 1
    temp := array[i]
    j := i - 1
    поки  j >= 0  &&  array[j] < temp
      повторити
        array[j+1] := array[j]
        array[j] := temp
        j--
      все повторити
  все повторити
  output_array(array)
Кінець
```

ouput_matrix(matrix[6][4]):

для i **від** 0 **до** 6 **з кроком** 1
для j **від** 0 **до** 4 **з кроком** 1
Виведення matrix[i][j] “\t”
все повторити
Виведення “\n”
все повторити
Кінець

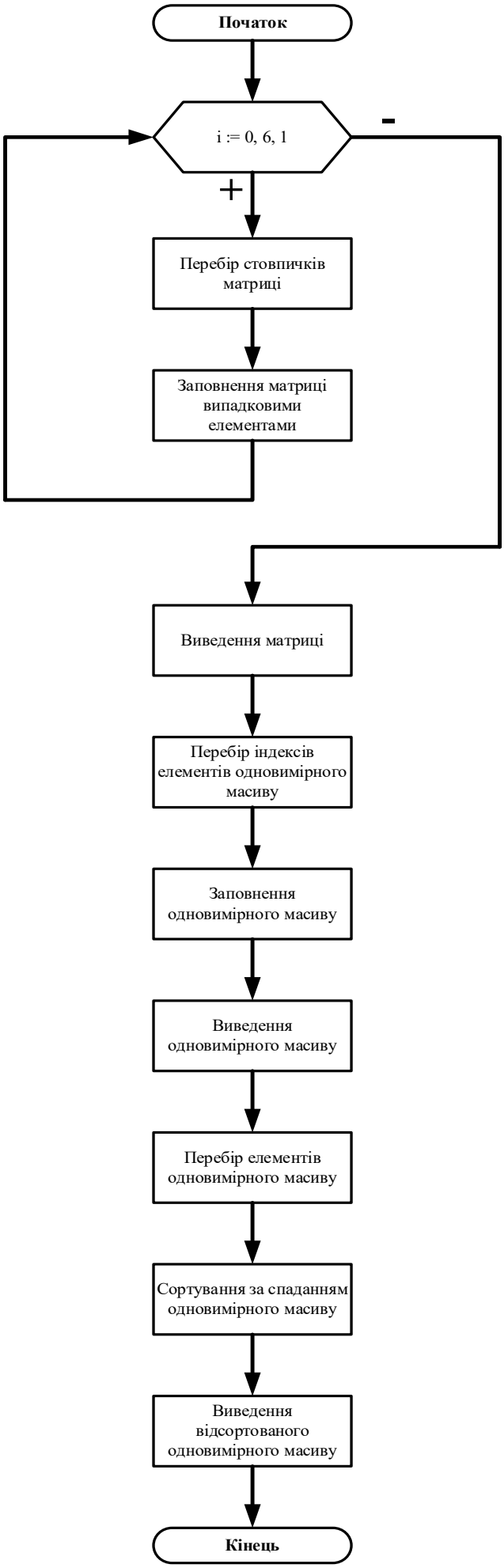
output_array(array[6]):

для i **від** 0 **до** 6 **з кроком** 1
Виведення array[i] “\t”
все повторити
Виведення “\n\n”
Кінець

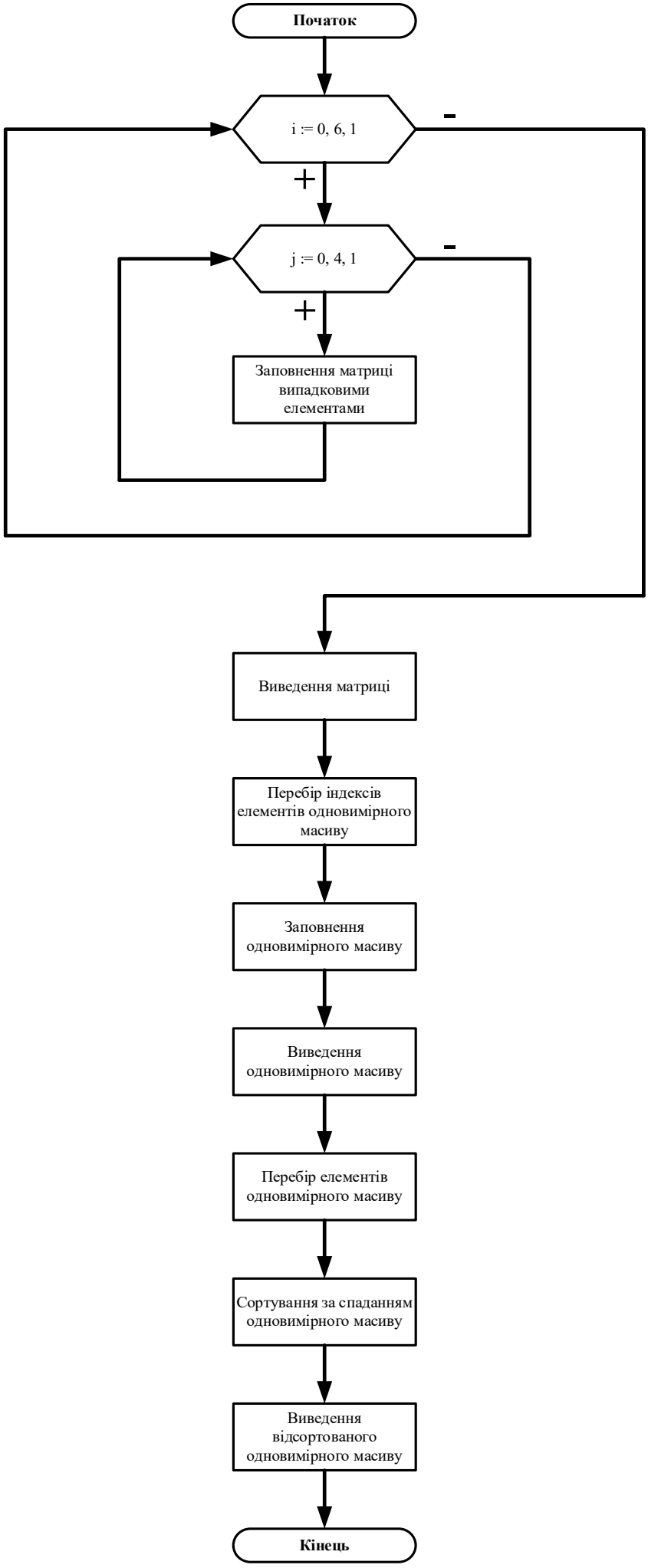
Блок-схема

Крок 1:

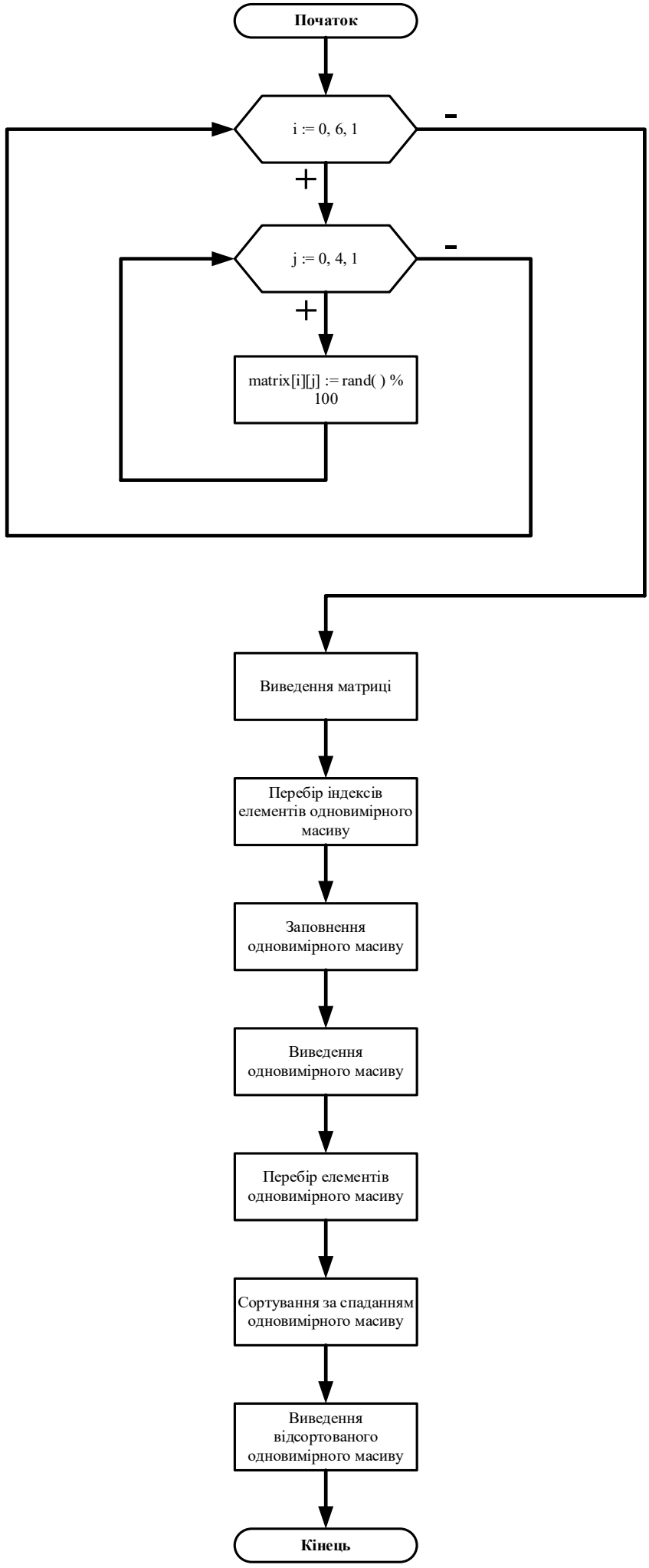
Крок 2:



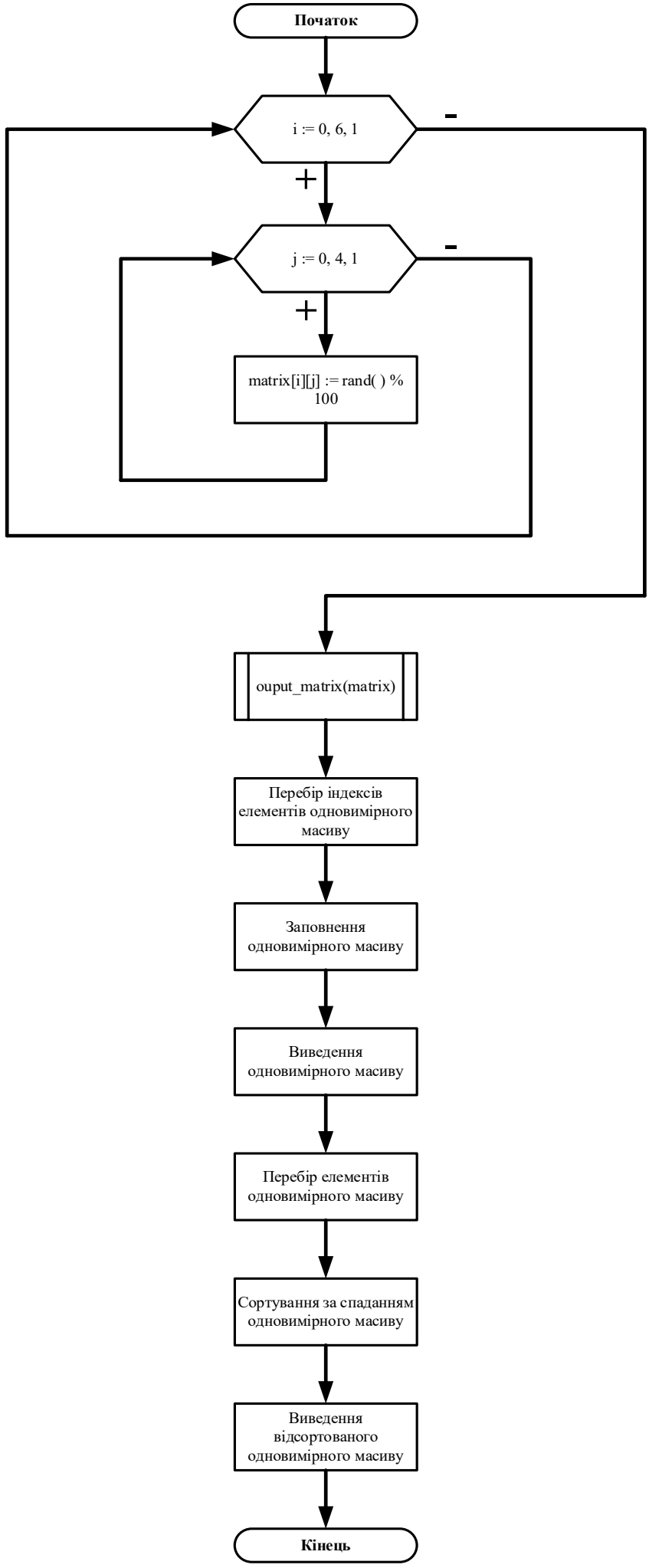
Крок 3:


Початок
i := 0, 6, 1
-
+
j := 0, 4, 1
-
+
Заповнення матриці випадковими елементами
Виведення матриці
Перебір індексів елементів одновимірного масиву
Заповнення одновимірного масиву
Виведення одновимірного масиву
Перебір елементів одновимірного масиву
Сортування за спаданням одновимірного масиву
Виведення відсортованого одновимірного масиву
Кінець

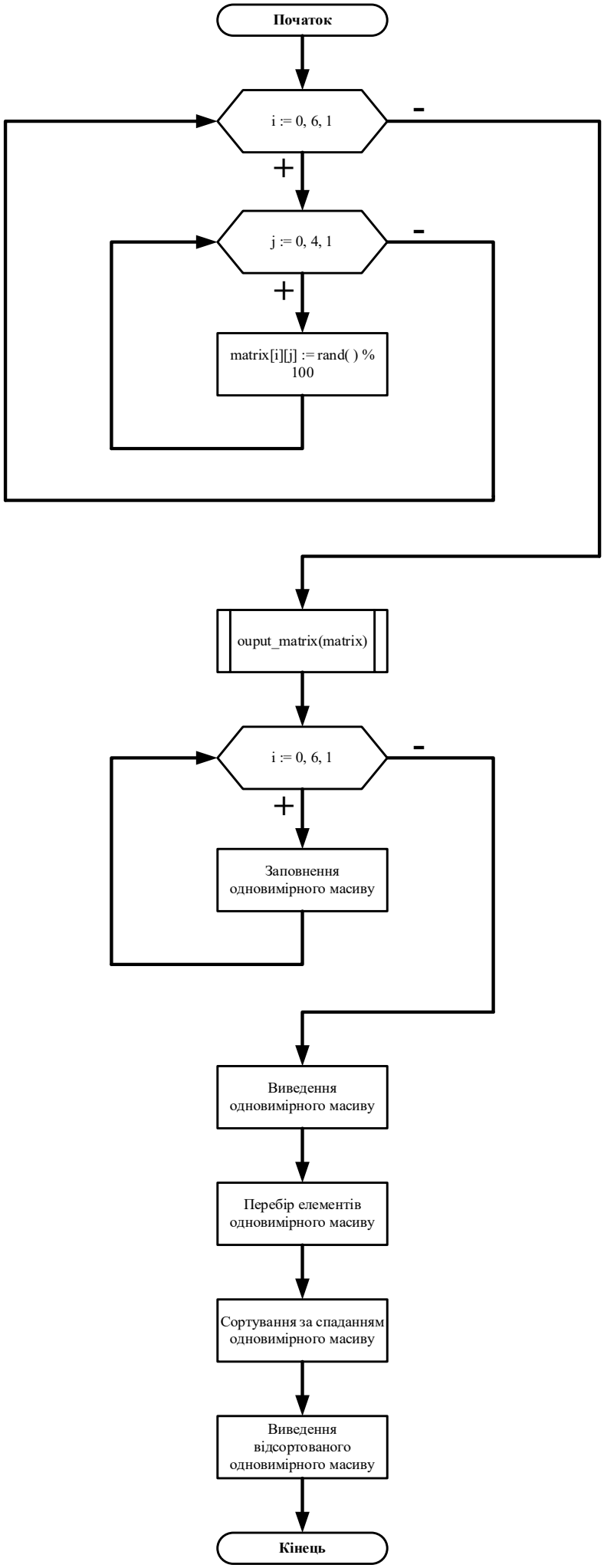
Крок 4:


Початок
i := 0, 6, 1
-
+
j := 0, 4, 1
-
+
matrix[i][j] := rand( ) % 100
Виведення матриці
Перебір індексів елементів одновимірного масиву
Заповнення одновимірного масиву
Виведення одновимірного масиву
Перебір елементів одновимірного масиву
Сортування за спаданням одновимірного масиву
Виведення відсортованого одновимірного масиву
Кінець

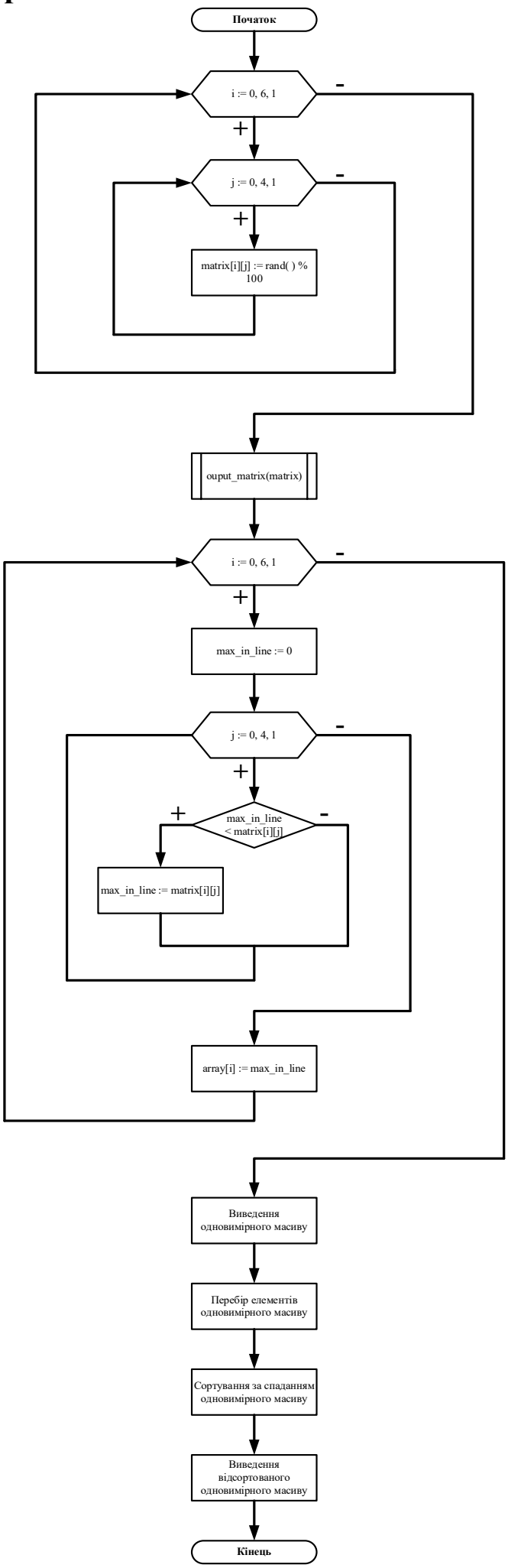
Крок 5:


Початок
i := 0, 6, 1
-
+
j := 0, 4, 1
-
+
matrix[i][j] := rand( ) % 100
ouput_matrix(matrix)
Перебір індексів елементів одновимірного масиву
Заповнення одновимірного масиву
Виведення одновимірного масиву
Перебір елементів одновимірного масиву
Сортування за спаданням одновимірного масиву
Виведення відсортованого одновимірного масиву
Кінець

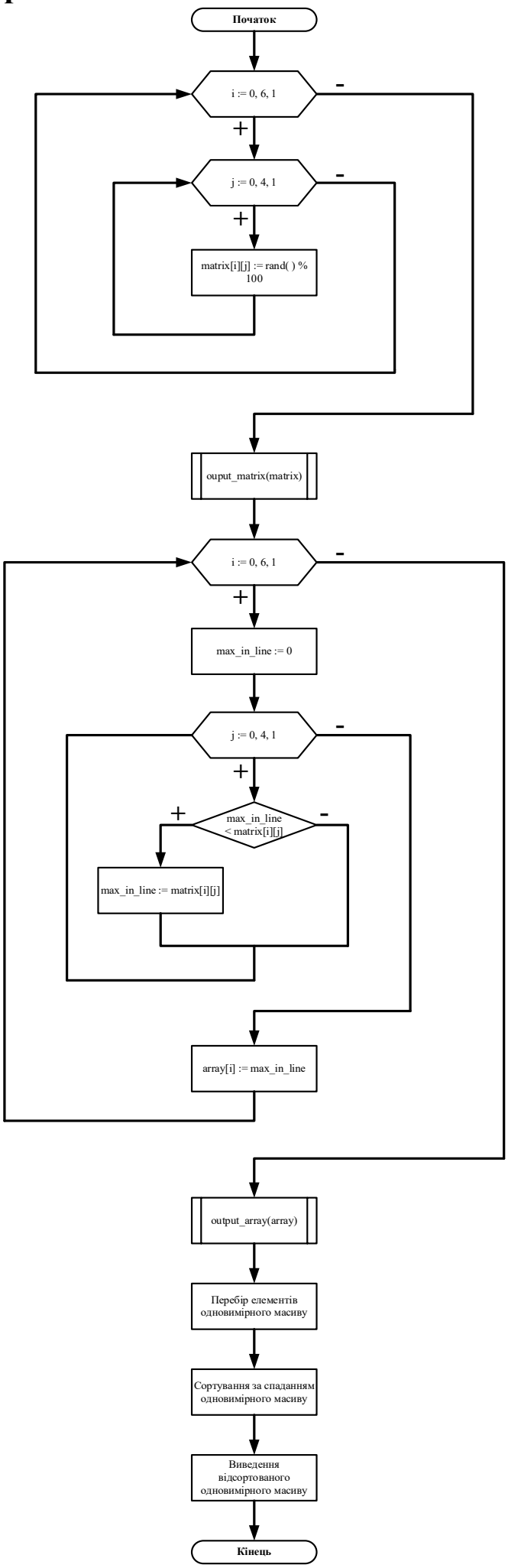
Крок 6:


Початок
i := 0, 6, 1
-
+
j := 0, 4, 1
-
+
matrix[i][j] := rand( ) % 100
ouput_matrix(matrix)
i := 0, 6, 1
-
+
Заповнення одновимірного масиву
Виведення одновимірного масиву
Перебір елементів одновимірного масиву
Сортування за спаданням одновимірного масиву
Виведення відсортованого одновимірного масиву
Кінець

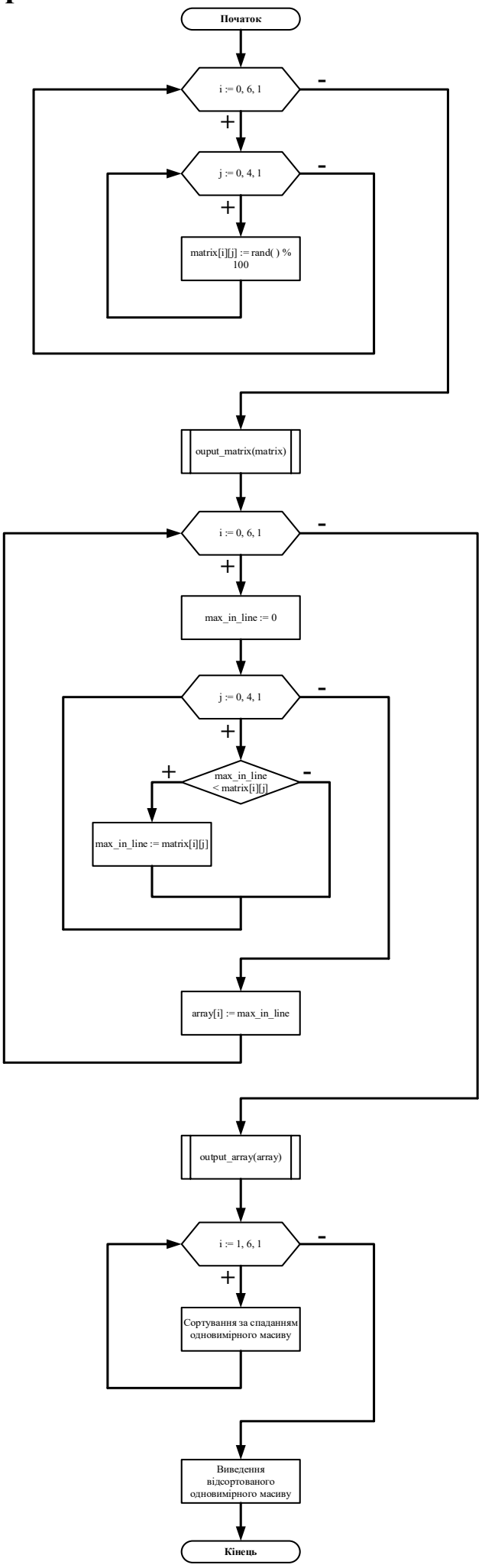
Крок 7:


Початок
i := 0, 6, 1
-
+
j := 0, 4, 1
-
+
matrix[i][j] := rand( ) % 100
ouput_matrix(matrix)
i := 0, 6, 1
-
+
max_in_line := 0
j := 0, 4, 1
-
+
+
max_in_line < matrix[i][j]
-
max_in_line := matrix[i][j]
array[i] := max_in_line
Виведення одновимірного масиву
Перебір елементів одновимірного масиву
Сортування за спаданням одновимірного масиву
Виведення відсортованого одновимірного масиву
Кінець

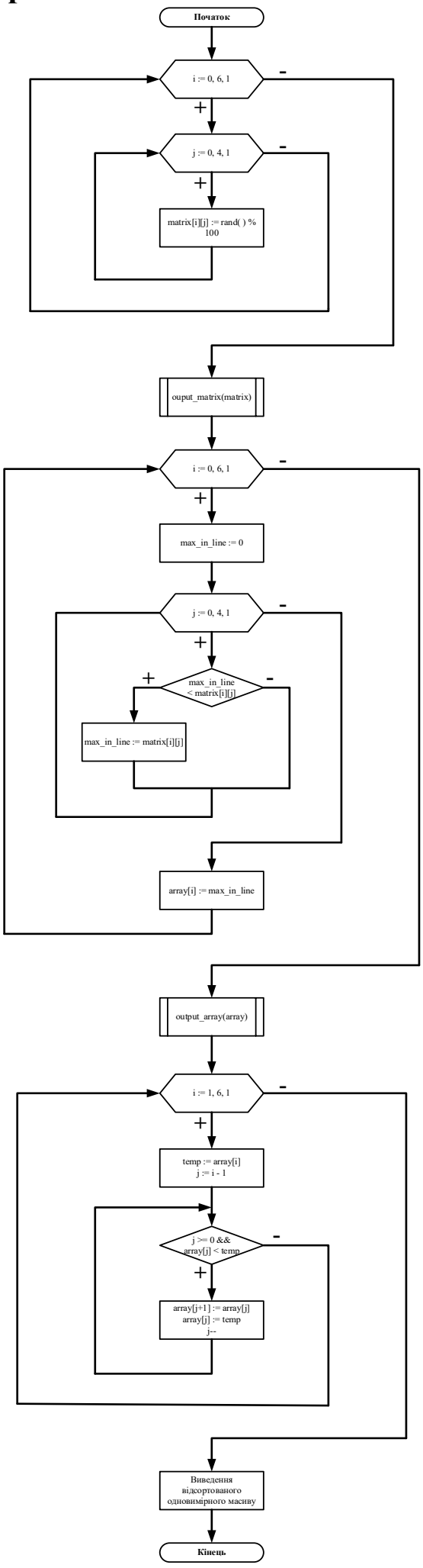
Крок 8:


Початок
i := 0, 6, 1
-
+
j := 0, 4, 1
-
+
matrix[i][j] := rand( ) % 100
ouput_matrix(matrix)
i := 0, 6, 1
-
+
max_in_line := 0
j := 0, 4, 1
-
+
+
max_in_line < matrix[i][j]
-
max_in_line := matrix[i][j]
array[i] := max_in_line
output_array(array)
Перебір елементів одновимірного масиву
Сортування за спаданням одновимірного масиву
Виведення відсортованого одновимірного масиву
Кінець

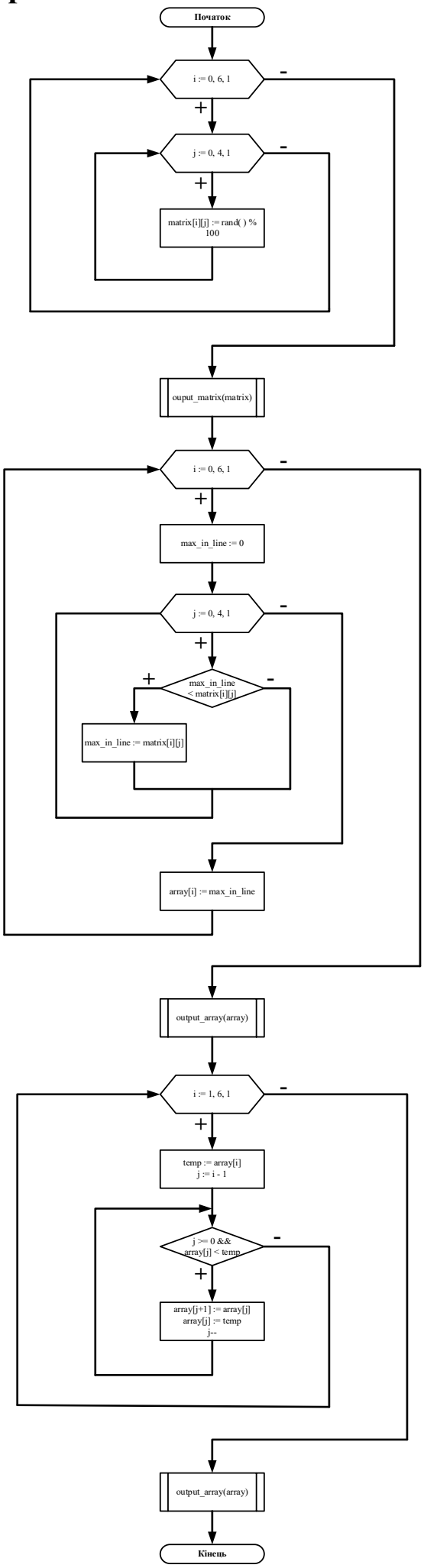
Крок 9:


Початок
i := 0, 6, 1
-
+
j := 0, 4, 1
-
+
matrix[i][j] := rand( ) % 100
ouput_matrix(matrix)
i := 0, 6, 1
-
+
max_in_line := 0
j := 0, 4, 1
-
+
+
max_in_line < matrix[i][j]
-
max_in_line := matrix[i][j]
array[i] := max_in_line
output_array(array)
i := 1, 6, 1
-
+
Сортування за спаданням одновимірного масиву
Виведення відсортованого одновимірного масиву
Кінець

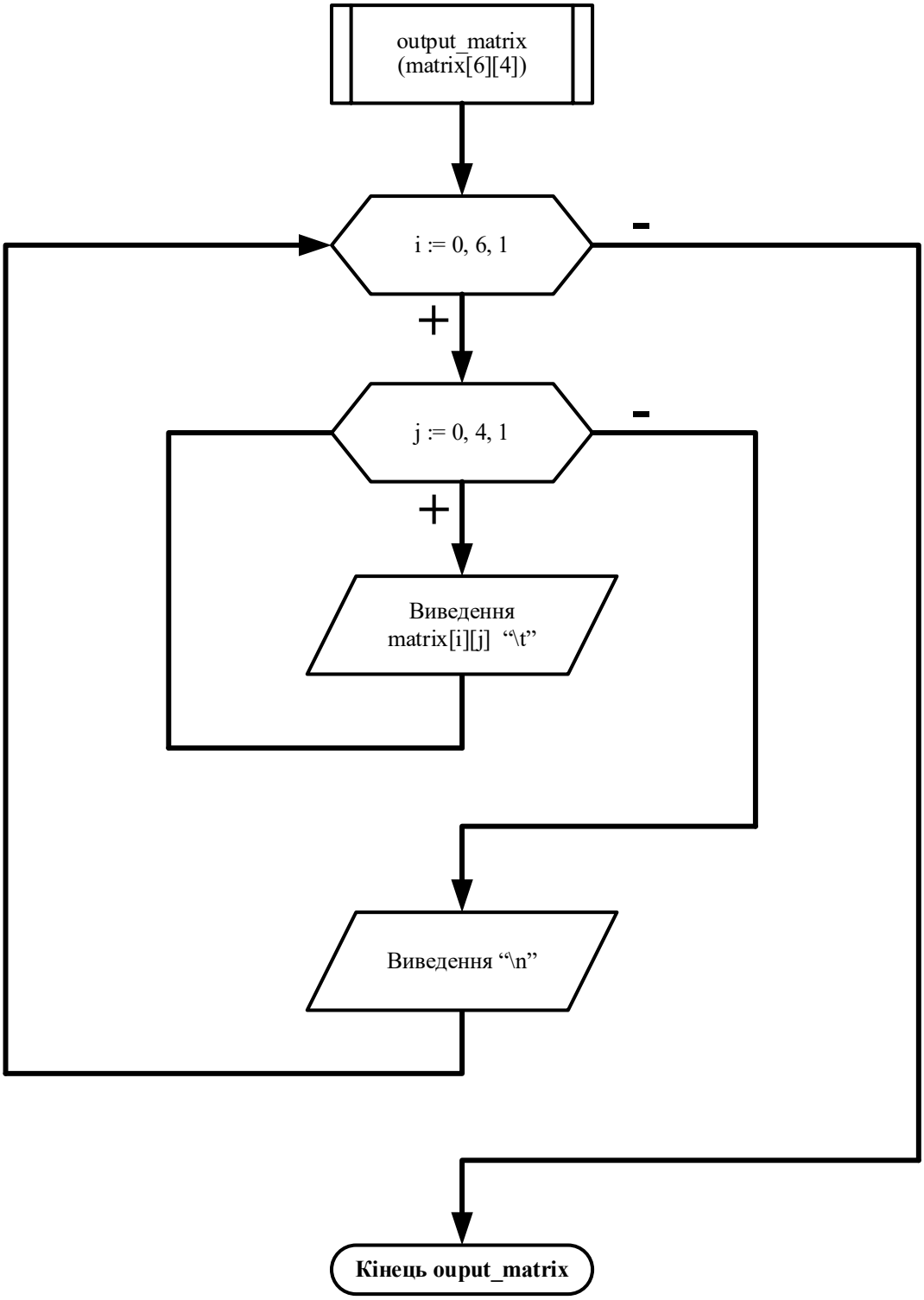
Крок 10:


Початок
i := 0, 6, 1
-
+
j := 0, 4, 1
-
+
matrix[i][j] := rand( ) % 100
ouput_matrix(matrix)
i := 0, 6, 1
-
+
max_in_line := 0
j := 0, 4, 1
-
+
+
max_in_line < matrix[i][j]
-
max_in_line := matrix[i][j]
array[i] := max_in_line
output_array(array)
i := 1, 6, 1
-
+
temp := array[i]
j := i - 1
j >= 0 && array[j] < temp
-
+
array[j+1] := array[j]
array[j] := temp
j--
Виведення відсортованого одновимірного масиву
Кінець

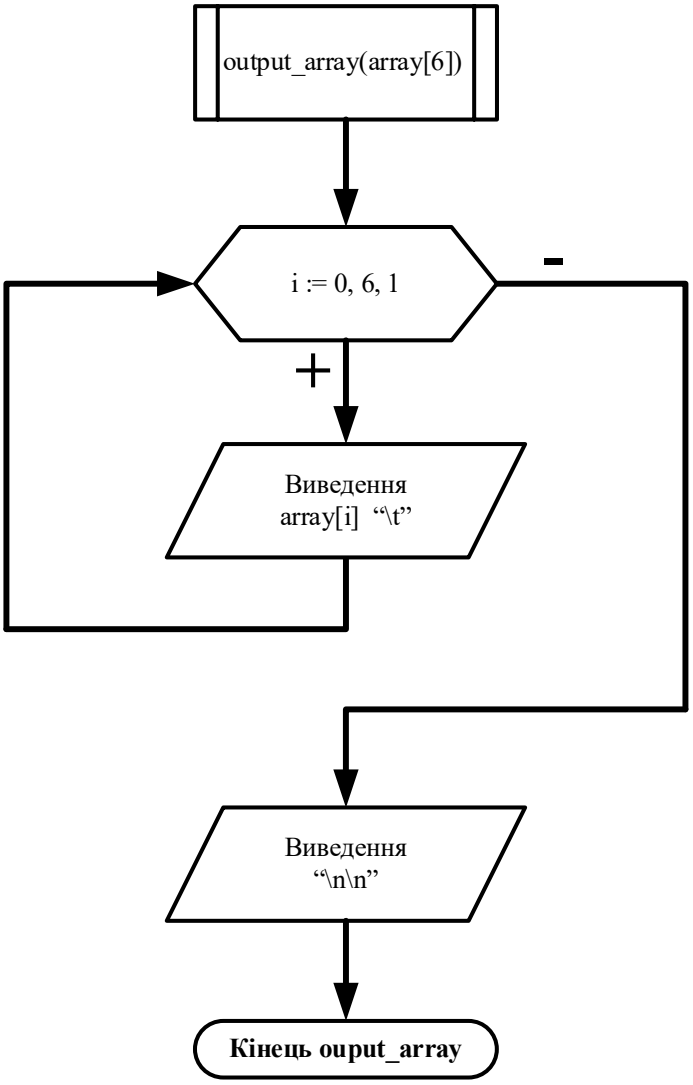
Крок 11:


Початок
i := 0, 6, 1
j := 0, 4, 1
matrix[i][j] := rand( ) % 100
ouput_matrix(matrix)
i := 0, 6, 1
max_in_line := 0
j := 0, 4, 1
max_in_line < matrix[i][j]
max_in_line := matrix[i][j]
array[i] := max_in_line
output_array(array)
i := 1, 6, 1
temp := array[i]
j := i - 1
j >= 0 && array[j] < temp
array[j+1] := array[j]
array[j] := temp
j--
output_array(array)
Кінець

ouput_matrix(matrix[6][4]):



output_array(array[6]):



Код програми на мові C++ :

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

void output_matrix(int matrix[6][4]);
void output_array(int array[6]);

int main() {
	srand(time(NULL));
	int matrix[6][4];
	int array[6];
	for (int i = 0; i < 6; i++) {
		for (int j = 0; j < 4; j++) {
			matrix[i][j] = rand() % 100;
		}
	}
	output_matrix(matrix);
	int max_in_line;
	for (int i = 0; i < 6; i++) {
		max_in_line = 0;
		for (int j = 0; j < 4; j++) {
			if (max_in_line < matrix[i][j]) {
				max_in_line = matrix[i][j];
			}
		}
		array[i] = max_in_line;
	}
	cout << "array: " << endl;
	output_array(array);
	int temp, j;
	for (int i = 1; i < 6; i++) {
		temp = array[i];
		j = i - 1;
		while (j >= 0 && array[j] < temp) {
			array[j + 1] = array[j];
			array[j] = temp;
			j--;
		}
	}
	cout << "newArray: " << endl;
	output_array(array);
	system("pause");
	return 0;
}

void output_matrix(int matrix[6][4]) {
	cout << "matrix: " << endl;
	for (int i = 0; i < 6; i++) {
		for (int j = 0; j < 4; j++) {
			cout << matrix[i][j] << "\t";
		}
		cout << "\n";
	}
	cout << endl;
```

```
}

void output_array(int array[6]) {
	for (int i = 0; i < 6; i++) {
		cout << array[i] << "\t";
	}
	cout << "\n\n";
}
```

Тестування програми:

```
matrix:
50	26	13	95
20	5	79	56
57	57	67	18
8	19	47	43
90	57	4	81
66	12	93	5

array:
95	79	67	47	90	93

newArray:
95	93	90	79	67	47
```

```
matrix:
26      16      6       94
51      41      11      56
47      41      48      96
60      85      7       28
2       11      59      61
11      83      93      45

array:
94      56      96      85      61      93

newArray:
96      94      93      85      61      56
```

```
matrix:
16      31      52      10
4       15      25      82
47      2       70      51
94      32      1       98
50      98      36      4
50      73      85      8

array:
52      82      70      98      98      85

newArray:
98      98      85      82      70      52
```

Висновок.

У результаті лабораторної роботи було розроблено математичну модель, що відповідає постановці задачі; псевдокод та блок-схеми, які пояснюють логіку алгоритму. Було набуто практичного новичок у використанні алгоритмів пошуку та сортування та їх інтерпретації у блок-схеми і псевдокод.

Алгоритм був випробуваний 3 рази. І в кожному з них була сгенерована матриця, заповнений масив максимальними елементами рідків матриці. У підсумку, було отримано правильно відсортований за спаданням одновимірний масив. Таким чином, було доведено вірність складеного алгоритму. Отже, його можна застосовувати для обчислення значення елементів одновимірного масиву із максимальних значень елементів рядків матриці та сортування цього одновимірного масиву методом вставки за спаданням.