

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 12

Виконала студентка ІП-15 Коваленко Марія Олександрівна  
(шифр, прізвище, ім'я, по батькові)  
Перевірила Вечерковська Анастасія Сергіївна  
( прізвище, ім'я, по батькові)

### Лабораторна робота 7

#### 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 12

#### Завдання

12. Задано множину послідовностей значень  $A[6,4]$ . Створити масив з максимальних значень елементів рядків двовимірного масиву. Відсортувати методом вставки за зростанням.

#### Постановка задачі

Задамо матрицю розмірністю 6 на 4

Створимо масив з найбільших елементів рядків матриці

Відсортуємо масив методом вставки за зростанням

#### Побудова математичної моделі

*Складемо таблицю імен змінних.*

Змінна	Тип	Ім'я	Призначення
лічильник	цілий	i	проміжні дані
лічильник	цілий	k	проміжні дані
розмірність матриці i масиву	цілий	m	проміжні дані
розмірність матриці	цілий	n	проміжні дані
вихідна матриця	цілий	matrix	проміжні дані
вихідний масив	цілий	array	вихідні дані
лічильник	цілий	t	проміжні дані
додаткова змінна для сортування	цілий	key	проміжні дані

також будемо використовувати функцію  $\text{rand}(a,b)$ , яка повертає рандомне ціле число у проміжку від  $a$  до  $b$

### Розв'язання

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

Крок 1. Визначимо основні дії;

Крок 2. Деталізуємо ініціалізування матриці

Крок 3. Деталізуємо знаходження необхідного масиву

Крок 4. Деталізуємо сортування необхідного масиву

### Псевдокод

початок

**m=6**

**n=4**

**повторити** для i від 0 до m

**повторити** для k від 1 до n

matrix[i, k]:=rand(1,1000);

**кінець циклу**

**кінець циклу**

**повторити** для i від 0 до m

array[i]=0;

**повторити** для k від 1 до n

**якщо** matrix[i,k]>array[i]

**то** array[i]:= matrix[i,k]

**кінець циклу**

**кінець циклу**

**повторити** для k від 1 до m

key = array[k]

t= k-1

array[k]=0;

**поки**(t>= 0 && array[t] > key)

**повторити**

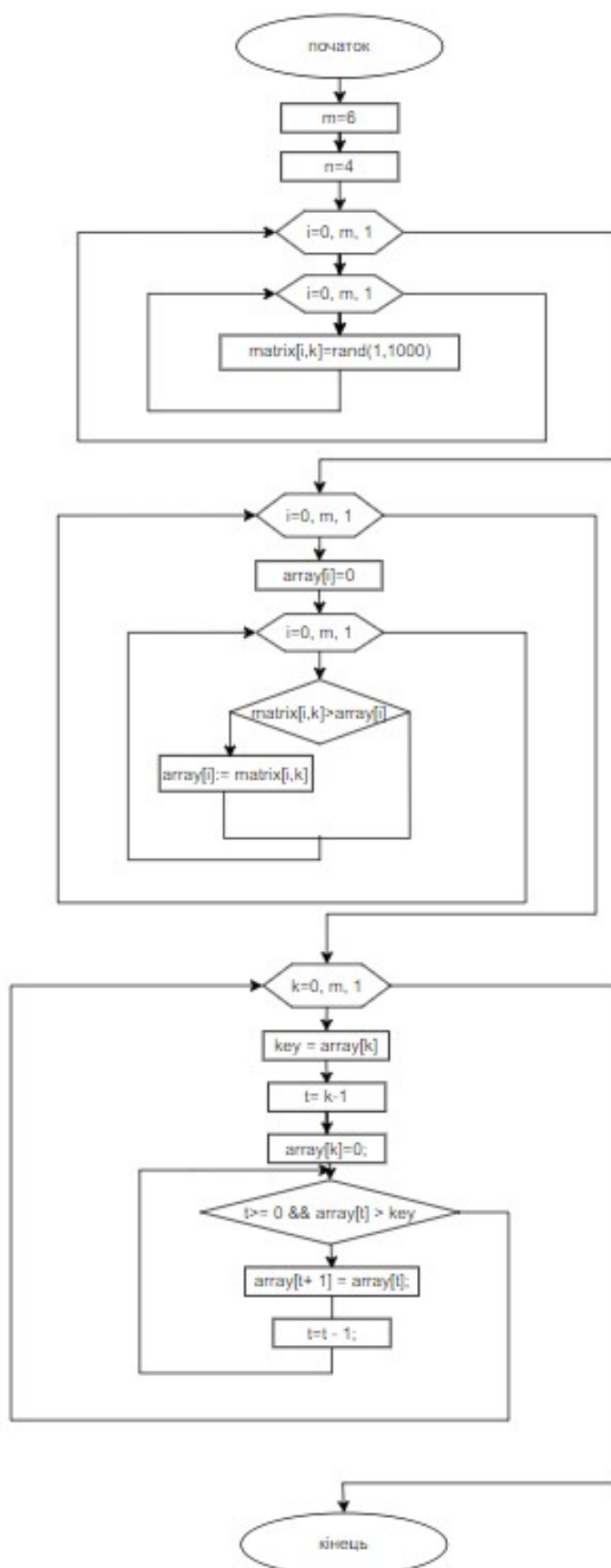
array[t+ 1] = array[t];

t=t - 1;

**все повторити**

кінець  
кінець циклу

Блок-схема



### Код

```
using System;

namespace lab8
{
    Символ: 0
    class Program
    {
        Символ: 0
        static void Main(string[] args)
        {
            int m = 6;
            int n = 4;

            int[,] matrix = new int[m, n];
            Random rnd = new Random();
            for (int i = 0; i < m; i++)
            {
                for (int k = 0; k < n; k++)
                {
                    matrix[i, k] = rnd.Next(1, 100);
                }
            }

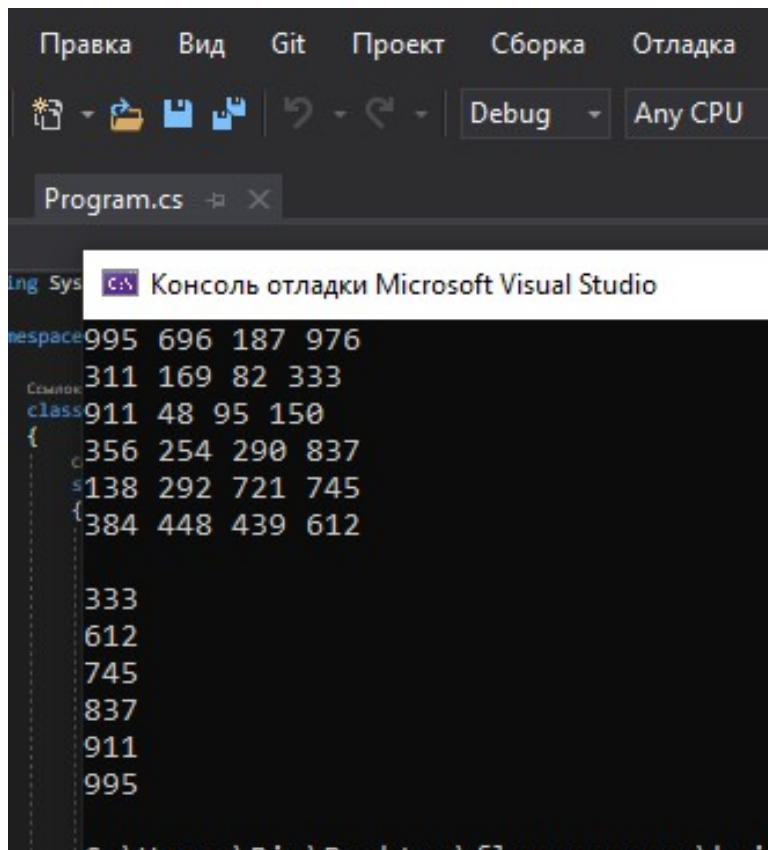
            int[] array = new int[m];

            for (int i = 0; i < m; i++)
            {
                array[i] = 0;
                for (int k = 0; k < n; k++)
                {
                    if (matrix[i, k] > array[i])
                    {
                        array[i] = matrix[i, k];
                    }
                }
            }

            int key, l;
            for (int k = 1; k < m; k++)
            {
                key = array[k];
                l = k - 1;
                while ((l >= 0) && (array[l] > key))
                {
                    array[l + 1] = array[l];
                    l = l - 1;
                }
                array[l + 1] = key;
            }
        }
    }
}
```

### Тестування

(для зручності пізніше я ввела у код виведення матриці та відсортованого масиву її найбільших елементів)



The screenshot shows the Visual Studio IDE with the 'Program.cs' file open. The console output displays a 4x4 matrix of numbers, followed by a list of the top 10 elements sorted in descending order.

```
995 696 187 976
311 169 82 333
911 48 95 150
356 254 290 837
138 292 721 745
384 448 439 612

333
612
745
837
911
995
```

### Висновок:

Ми дослідили алгоритми пошуку та сортування, набули практичних навичок використання цих алгоритмів під час складання програмних специфікацій.