**Міністерство Освіти І НАУКИ України**

**Національний університет "Львівська політехніка"**

**КНІТ**

Кафедра **ПЗ**

### ЗВІТ

До лабораторної роботи № 4

**З дисципліни:** *“Алгоритми та структури даних”*

**На тему:** *“Метод швидкого сортування”*

**Лектор:**

доц. каф. ПЗ

Коротєєва Т.О.

**Виконала:**

ст. групи ПЗ-26

Матолінець Л. А.

**Прийняв:**

асистент каф. ПЗ

Симець І.І.

« \_\_\_\_ » \_\_\_\_\_\_\_\_ 2022 р.

∑= \_\_\_\_\_ .

Львів – 2022

**Тема роботи:** Метод швидкого сортування**.**

**Мета роботи**: Вивчити [алгоритм](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=472&eid=19102&displayformat=dictionary" \o "Глосарій: Алгоритм) швидкого сортування. Здійснити програмну реалізацію [алгоритм](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=472&eid=19102&displayformat=dictionary" \o "Глосарій: Алгоритм)у швидкого сортування. Дослідити швидкодію [алгоритм](http://vns.lp.edu.ua/mod/glossary/showentry.php?courseid=472&eid=19102&displayformat=dictionary" \o "Глосарій: Алгоритм)у швидкого сортування.

**TЕОРЕТИЧНІ ВІДОМОСТІ**

***Покроковий опис роботи алгоритму швидкого сортування.***

**Алгоритм QS** Задано одновимірний масив елементів *R1 ,R2 ,…,Rn.* Даний алгоритм реорганізує масив у висхідному порядку таким чином, що при виборі опорного елемента відбувається сортування, щоб всі елементи зліва були менші, справа – більші.

QS1. Ініціалізація: low = 1, high = n;

QS2. Перевірка: if low > high перехід на QS8

QS3. Ініціалізація: pivot = R[high], i = low - 1, j = low;

QS4. Пошук: якщо R[i] < pivot -> i ++ ->QS5;

QS5: R[i] <-> R[j];

QS6: low = 1, high = i - 1 -> QS2;

QS7: low = j, high = n -> QS2;

QS8: кінець рекурсії.

**ІНДИВІДУАЛЬНЕ ЗАВДАННЯ**

**Варіант 4**

Задано матрицю дійсних чисел. Впорядкувати (переставити) її стовпці за зростанням значень їх перших елементів.

**ХІД ВИКОНАННЯ**

Метод швидкого сортування:

using System;

using System.Diagnostics;

using System.Threading;

class GFG

{

static void swap(int[] arr, int i, int j)

{

int temp = arr[i];

arr[i] = arr[j];

arr[j] = temp;

}

static int partition(int[] arr, int low, int high)

{

int pivot = arr[high];

int i = low - 1;

for (int j = low; j <= high - 1; j++)

{

if (arr[j] < pivot)

{

i++;

swap(arr, i, j);

}

}

swap(arr, i + 1, high);

return (i + 1);

}

static void quickSort(int[] arr, int low, int high)

{

if (low < high)

{

int pi = partition(arr, low, high);

quickSort(arr, low, pi - 1);

quickSort(arr, pi + 1, high);

}

}

static void printArray(int[] arr, int size)

{

for (int i = 0; i < size; i++)

Console.Write(arr[i] + " ");

Console.WriteLine();

}

public static void Main()

{

int Min = -50000;

int Max = 50000;

Console.WriteLine("Enter the size of your array: ");

int length = Convert.ToInt32(Console.ReadLine());

int[] unsortedArr = new int[length];

Random randNum = new Random();

for (int i = 0; i < length; i++)

{

unsortedArr[i] = randNum.Next(Min, Max);

}

Stopwatch stopwatch = new Stopwatch();

stopwatch.Start();

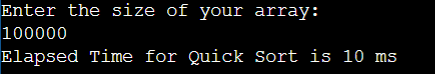
quickSort(unsortedArr, 0, length-1);

stopwatch.Stop();

Console.WriteLine("Elapsed Time for Quick Sort is {0} ms", stopwatch.ElapsedMilliseconds);

}

}



*Рис.1 Результат виконання програми*

Вихідний код виконання індивідуального завдання:

using System;

using System.Collections.Generic;

using System.Numerics;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

namespace SortingApp

{

/// <summary>

/// Interaction logic for MainWindow.xaml

/// </summary>

///

public class MyArrayOfArrays

{

protected MyArray[] array = new MyArray[0];

public MyArrayOfArrays() { }

public MyArray[] getList() { return array; }

public void addElement(MyArray item)

{

Array.Resize(ref array, array.Length + 1);

array[array.Length - 1] = item;

}

public void setArray(MyArray[] arr)

{

array = arr;

}

}

public partial class MainWindow : Window

{

MyArrayOfArrays matrix = new MyArrayOfArrays();

int rows\_amount;

int columns\_amount;

public MainWindow()

{

InitializeComponent();

}

private void Button\_Click(object sender, RoutedEventArgs e)

{

rows\_amount = System.Convert.ToInt32(rows\_count.Text);

columns\_amount = System.Convert.ToInt32(columns\_count.Text);

}

int index = 0;

private void addArray\_Click(object sender, RoutedEventArgs e)

{

MyArray array = new MyArray();

String arrField = column\_field.Text;

String[] strings = arrField.Split(" ; ");

foreach (var item in strings)

{

double x = Convert.ToDouble(item);

array.addElement(x);

}

matrix.addElement(array);

}

private void showArray\_Click(object sender, RoutedEventArgs e)

{

}

private void Array\_Click(object sender, RoutedEventArgs e)

{

matrix\_view.Text = String.Empty;

String str = "";

for (int i = 0; i < rows\_amount; i++)

{

for (int j = 0; j < columns\_amount; j++)

{

MyArray arr = matrix.getList()[j];

str += arr.getList()[i] + " ";

}

str += "\n";

}

matrix\_view.Text += str;

}

static void swap(MyArrayOfArrays arr, int i, int j)

{

MyArray[] arr1 = arr.getList();

MyArray temp = arr1[i];

arr1[i] = arr1[j];

arr1[j] = temp;

arr.setArray(arr1);

}

static int partition(MyArrayOfArrays arr, int low, int high)

{

MyArray pivot = arr.getList()[high];

int i = (low - 1);

for (int j = low; j <= high - 1; j++)

{

if (arr.getList()[j].getList()[0] < pivot.getList()[0])

{

i++;

swap(arr, i, j);

}

}

swap(arr, i + 1, high);

return (i + 1);

}

static void quickSort(MyArrayOfArrays arr, int low, int high)

{

if (low < high)

{

int pi = partition(arr, low, high);

quickSort(arr, low, pi - 1);

quickSort(arr, pi + 1, high);

}

}

private void Sort\_Click(object sender, RoutedEventArgs e)

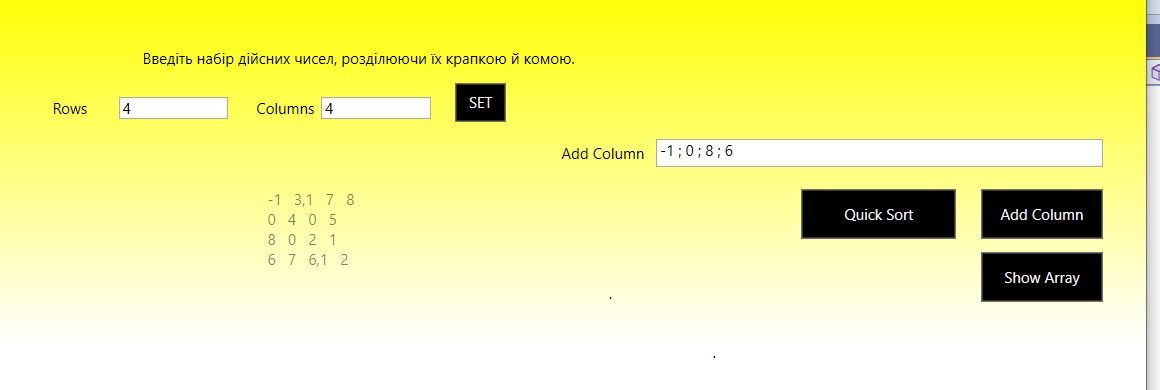
{

quickSort(matrix, 0, columns\_amount - 1);

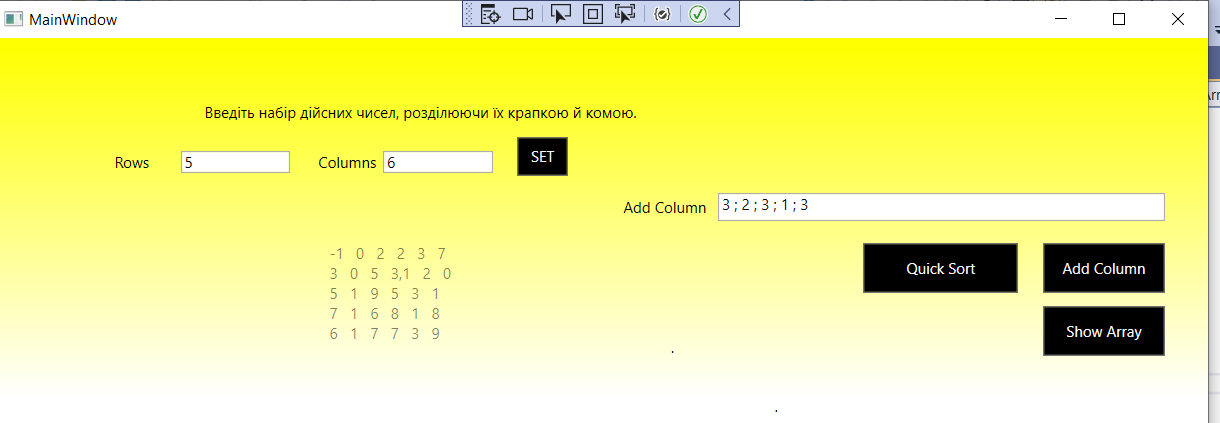
}

}

}



*Рис. 1 Результат сортування матриці з 16ти елементів*



*Рис. 2 Результат сортування матриці з 30ти елементів*

**Висновок**

На цій лабораторній роботі я ознайомилася з методом сортування Quick Sort, швидкодія якого є доволі великою в порівнянні з методами, застосованими раніше. Середня складність алгоритму – O(n\*log(n)). В результаті роботи я виконала індивідуальне завдання, в якому за допомогою Quick Sort посортувала матрицю за першими елементами стовпців та реалізувала алгоритм сортування для звичайних цілих чисел.