МІНІСТЕРСТВО

ОСВІТИ ТА НАУКИ УКРАЇНИ

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Кафедра інформаційних систем та технологій

Звіт

з лабораторної роботи № 1

«Класи та об’єкти. Конструктори та деструктори. Модифікатори.»

з дисципліни

«Програмування – 2. ООП»

Варіант №21

Перевірив:

доц. Корнага Ярослав Ігорович

Виконала:

Студентка ІС-13, ФІОТ

Росновська Ольга

Київ 2022

Завдання 1

Створити клас с атрибутами та конструктором. У методі main() ініціалізувати створення екземплярів класу та продемонструвати роботу його методів згідно умов завдання.

Скласти опис класу для послідовності. Зберігає послідовність цілих чисел. Методи: тип (спадна, зростаюча, неспадна, незростаюча, геометрична, арифметична прогресія), належність елемента, чи рівні дві послідовності, максимум, мінімум, встановити роздільники підпослідовностей (лок. максимуми, лок. мінімуми, лок. екстремуми), найбільша (найменша) підпослідовність.

Завдання 2

Створити у попередньому завданні два методи з використанням серіалізації та десеріалізації JSON.

- **Метод 1**. Зберігає створений об’єкт класу з Завдання 1 у JSON файл
- **Метод 2.** Відкриває JSON файл з даними та створює об’єкт класу з цими даними для виконання Завдання 1.

Лістинг програми

Program.cs

```
using System;

namespace Lab_1
{
    class Program
    {
        static void Main(string[] args)
        {

            Console.OutputEncoding = System.Text.Encoding.Unicode;
            Console.WriteLine("Введіть послідовність цілих чисел ↓");
            string[] input = Console.ReadLine().Split(' ');
            int[] array = new int[input.Length];
            for (int i = 0; i < input.Length; i++)
            {
                array[i] = int.Parse(input[i]);
            }
```

```
            Count s = new Count(array);

            Console.WriteLine("\n" + "Належність елемента:");
            Console.WriteLine(s.Is_in(9) + "\n");//належність елемента


            Console.WriteLine("Дана послідовність рівна {5, 6, 7, 8, 9}:");
            int[] list = new int[] { 5, 6, 7, 8, 9 };
            Console.WriteLine(s.Is_equal(list));//чи рівні послідовності

            Console.WriteLine("\n" + "Максимум:");
            Console.WriteLine(s.Max());

            Console.WriteLine("\n" + "Мінімум:");
            Console.WriteLine(s.Min());


            s.My_Type();
            Console.WriteLine("\n" + "Локальні максимуми: ");
            for (int i = 0; i < s.loc_max().Length; i++) {
Console.WriteLine(s.loc_max()[i] + " "); }

            Console.WriteLine("\n" + "Локальні мінімуми: ");
            for (int i = 0; i < s.loc_min().Length; i++) {
Console.WriteLine(s.loc_min()[i] + " "); }

            Console.WriteLine("\n" + "Екстремуми: ");
            for (int i = 0; i < s.extremes().Length; i++) {
Console.WriteLine(s.extremes()[i] + " "); }

            Console.WriteLine("\n" + "Найбільша підпослідовність: ");
            for (int i = 0; i < s.max_subsequence().Length; i++) {
Console.WriteLine(s.max_subsequence()[i] + " "); }

            Console.WriteLine("\n" + "Найменша підпослідовність: ");
            for (int i = 0; i < s.min_subsequence().Length; i++) {
Console.WriteLine(s.min_subsequence()[i] + " "); }

            s.ToJson("new.json");
            var json_array = Count.FromJson("test.json");
            Console.WriteLine("\n" + "Масив з json-файлу: ");
            for (int i = 0; i < json_array.array.Length; i++) {
Console.Write(json_array.array[i] + " "); }
        }
    }
}
```

Count.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Newtonsoft.Json;
using System.IO;


public class Count
{
	public int[] array;
	public Count(int[] numbers)
	{
		array = numbers;
	}

	public bool Is_in(int n)
	{
		for (int i = 0; i < array.Length; i++) //належність елемента
		{
			if (array[i] == n)
			{
				return true;

			}
		}
		return false;
	}

	public bool Is_equal(int[] list)//чи рівні послідовності
	{
		if (array.Length != list.Length) { return false; }
		for (int i = 0; i < array.Length; i++)
		{
			if (array[i] != list[i])
			{
				return false;
			}
		}
		return true;
	}

	public int Max()
	{
		int max = array[0];
		for (int i = 1; i < array.Length; i++)
		{
			if (max < array[i])
			{
				max = array[i];
			}
		}
		return max;
	}

	public int Min()
	{
		int min = array[0];
		for (int i = 1; i < array.Length; i++)
		{
			if (min > array[i])
			{
```

```
                        min = array[i];
                    }
                }
                return min;
        }

        public void My_Type()
        {
                bool is_descending = true; // спадна послідовність
                bool is_ascending = true; // зростаюча послідовність
                bool is_notdes = true; // неспадна послідовність
                bool is_notasc = true; // незростаюча послідовність
                for (int i = 0; i < array.Length - 1; i++)
                {
                        if (array[i] <= array[i + 1])
                        {
                                is_descending = false;
                        }

                        if (array[i] >= array[i + 1])
                        {
                                is_ascending = false;
                        }

                        if (array[i] < array[i + 1])
                        {
                                is_notdes = false;
                        }

                        if (array[i] > array[i + 1])
                        {
                                is_notasc = false;
                        }
                }

                if (is_descending) { Console.WriteLine("Спадна послідовність"); }
                if (is_ascending) { Console.WriteLine("Зростаюча послідовність"); }
                if (is_notasc & is_ascending != true) { Console.WriteLine("Неспадна
послідовність"); }
                if (is_notdes & is_descending != true) { Console.WriteLine("Незростаюча
послідовність"); }

                bool is_arithmetic = true; // неспадна послідовність
                bool is_geometric = true; // незростаюча послідовність
                if (array.Length >= 2)
                {
                        int d = array[1] - array[0];
                        int q = array[1] / array[0];
                        for (int i = 1; i < array.Length - 1; i++)
                        {
                                if (array[i + 1] - array[i] != d)
                                {
                                        is_arithmetic = false;
                                }

                                if (array[i] != 0)
                                {
                                        if (array[i + 1] / array[i] != q)
                                        {
                                                is_geometric = false;
                                        }
                                }
                                else { is_geometric = false; }
                        }
                }
```

```
        if (is_arithmetic) { Console.WriteLine("Арифметична прогресія"); }
        if (is_geometric) { Console.WriteLine("Геометрична прогресія"); }
}

public int[] loc_max()
{
        string numbers = "";

        if (array.Length == 1) { numbers += array[0] + " "; }
        if (array.Length >= 2)
        {
                if (array[0] > array[1]) { numbers += array[0] + " "; }
                if (array[^1] > array[^2]) { numbers += array[^1] + " "; }
        }
        if (array.Length >= 3)
        {
                for (int i = 1; i < array.Length - 1; i++)
                {
                        if (array[i] > array[i - 1] & array[i] > array[i + 1])
                        {
                                numbers += array[i] + " ";
                        }
                }
        }

        string[] s = numbers.Split(' ');
        int[] a = new int[s.Length - 1];
        for (int i = 0; i < s.Length - 1; i++) { a[i] = int.Parse(s[i]); }

        return a;
}

public int[] loc_min()
{
        string numbers = "";

        if (array.Length == 1) { numbers += array[0] + " "; }
        if (array.Length >= 2)
        {
                if (array[0] < array[1]) { numbers += array[0] + " "; }
                if (array[^1] < array[^2]) { numbers += array[^1] + " "; }
        }
        if (array.Length >= 3)
        {
                for (int i = 1; i < array.Length - 1; i++)
                {
                        if (array[i] < array[i - 1] & array[i] < array[i + 1])
                        {
                                numbers += array[i] + " ";
                        }
                }
        }

        string[] s = numbers.Split(' ');
        int[] a = new int[s.Length - 1];
        for (int i = 0; i < s.Length - 1; i++) { a[i] = int.Parse(s[i]); }

        return a;
}

public int[] extremes()
{
        string numbers = "";

        if (array.Length == 1) { numbers += array[0] + " "; }
```

```
            if (array.Length >= 2)
            {
                  if (array[0] != array[1]) { numbers += array[0] + " "; }
                  for (int i = 1; i < array.Length - 1; i++)
                  {
                        if ((array[i] < array[i - 1] & array[i] < array[i + 1]) ||
(array[i] > array[i - 1] & array[i] > array[i + 1]))
                        {
                              numbers += array[i] + " ";
                        }
                  }
                  if (array[^1] != array[^2]) { numbers += array[^1] + " "; }
            }

            string[] s = numbers.Split(' ');
            int[] a = new int[s.Length - 1];
            for (int i = 0; i < s.Length - 1; i++) { a[i] = int.Parse(s[i]); }

            return a;
      }


      public int[] max_subsequence()
      {
            string sub = array[0] + " ";
            string max_sub = "";
            int len = 1;
            int max_len = 0;
            for (int i = 1; i < array.Length; i++)
            {
                  sub += array[i] + " ";
                  len += 1;

                  if (i == array.Length - 1)
                  {
                        len += 1;
                        if (len >= max_len)
                        {
                              max_len = len;
                              max_sub = sub;
                        }
                  }
                  else
                  {
                        if ((array[i] > array[i - 1] && array[i] > array[i + 1])
|| (array[i] < array[i - 1] && array[i] < array[i + 1]))
                        {
                              if (len >= max_len)
                              {
                                    max_len = len;
                                    max_sub = sub;
                              }
                              sub = array[i] + " ";
                              len = 0;
                        }
                  }
            }

            string[] s = max_sub.Split(' ');
            int[] a = new int[s.Length - 1];
            for (int i = 0; i < s.Length - 1; i++) { a[i] = int.Parse(s[i]); }

            return a;
      }
```

```
        public int[] min_subsequence()
        {
                string sub = array[0] + " ";
                string min_sub = "";
                int len = 1;
                int min_len = array.Length;
                for (int i = 1; i < array.Length; i++)
                {
                        sub += array[i] + " ";
                        len += 1;

                        if (i == array.Length - 1)
                        {
                                len += 1;
                                if (len <= min_len)
                                {
                                        min_len = len;
                                        min_sub = sub;
                                }
                        }
                        else
                        {
                                if ((array[i] > array[i - 1] && array[i + 1] < array[i])
|| (array[i] < array[i - 1] && array[i + 1] > array[i]))
                                {
                                        if (len <= min_len)
                                        {
                                                min_len = len;
                                                min_sub = sub;
                                        }
                                        sub = array[i] + " ";
                                        len = 0;
                                }
                        }
                }

                string[] s = min_sub.Split(' ');
                int[] a = new int[s.Length - 1];
                for (int i = 0; i < s.Length - 1; i++) { a[i] = int.Parse(s[i]); }

                return a;
        }


        public void ToJson(string filePath)
        {
                string j = JsonConvert.SerializeObject(this);

                File.WriteAllText(filePath, j);
        }

        public static Count FromJson(string filePath)
        {
                return
JsonConvert.DeserializeObject<Count>(File.ReadAllText(filePath));
        }


}
```

Результат виконання програми

```
Введіть послідовність цілих чисел ↓
1 2 3 4 5 6 7 8 9

Належність елемента:
True

Дана послідовність рівна {5, 6, 7, 8, 9}:
False

Максимум:
9

Мінімум:
1
Зростаюча послідовність
Арифметична прогресія

Локальні максимуми:
9

Локальні мінімуми:
1

Екстремуми:
1
9
```

```
Найбільша підпослідовність:
1
2
3
4
5
6
7
8
9

Найменша підпослідовність:

Масив з json-файлу:
2 7 8 9 -5 6 25 4
D:\2 semestr\прога\labka1\labka1\bin\Debug\net6.0\labka1.exe (процесс 19836) завершил работу с кодом 0.
Нажмите любую клавишу, чтобы закрыть это окно:
```

Висновок

Під час виконання цієї лабораторної роботи, я навчилася працювати з класами, об’єктами, конструкторами, деструкторами та модифікаторами у мові C#, а також з json-файлами.