

Лабораторна робота №1

Дисципліна «КРОС-ПЛАТФОРМНЕ ПРОГРАМУВАННЯ»

Тема роботи: ВИВЧЕННЯ ОСНОВНИХ ОПЕРАТОРІВ МОВИ C#

Цілі роботи

1. Ознайомитись з основними типами даних мови C# та платформи .NET
2. Набути практичних навичок роботи з операторами вибору та циклу

Завдання до лабораторної роботи

1. Написати консольний застосунок, який реалізує підрахунок суми елементів масиву до першого від'ємного (без використання оператора break)
2. Написати консольний застосунок, що знаходить перший унікальний (такий, що не повторюється) символ у рядку символів.
3. Написати консольний застосунок, що моделює кидання гральної кості та обчислює суму значень, що випали. Побудувати гістограму розподілу суми значень:
 - a. одна гральна кість.
 - b. дві гральні кості (наприклад, суми 2 та 1» будуть випадати доволі рідко, а 7 – доволі часто).
 - c. N гральних костей та K граней.

Виконання роботи

Завдання 1

Текст програми

```
// See https://aka.ms/new-console-template for more information

//all the variables
int n;
int i = 0;
int sum = 0;

//initializing array
Console.WriteLine("Enter the number of array elements:");
if (!Int32.TryParse(Console.ReadLine(), out n))
    n = 10;
Console.WriteLine($"Array will have {n} elements!");
int[] myArray = new int[n];
Console.WriteLine();

//fill the array
```

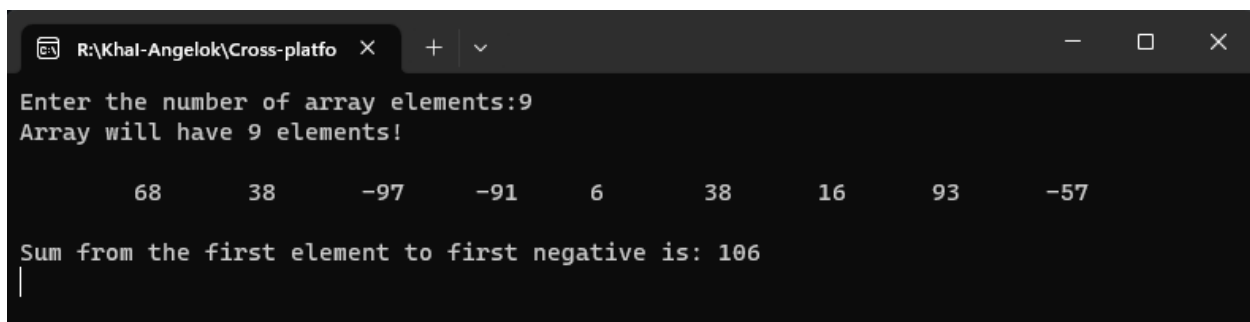
```

Random random = new Random();
for (i = 0; i < n; i++)
{
    myArray[i] = random.Next(-100, 100);
    Console.Write($"{t}{myArray[i]}");
}
Console.WriteLine();

//calculate sum
i = 0;
while (myArray[i] >= 0 && i < n)
{
    sum = sum + myArray[i];
    i++;
}
Console.Write($"{n}Sum from the first element to first negative is:
{sum}\n");
Console.ReadKey();

```

Результат виконання програми



```

R:\Khal-Angelok\Cross-platfo
Enter the number of array elements:9
Array will have 9 elements!
68 38 -97 -91 6 38 16 93 -57
Sum from the first element to first negative is: 106

```

Примітка

Робота виконувалась мовою C# у середовищі Visual Studio 2022 Community Edition з використанням нового шаблону консольних застосунків (доступний починаючи з .NET 6). В цьому випадку вам достатньо написати код, який в класичній програмі містився б всередині методу main, а компілятор зробить решту роботи за вас (т.зв. режим top level statements).

При цьому середовище також неявно підключає наступні простори імен:

- System;
- System.IO;
- System.Collections.Generic;
- System.Linq;
- System.Net.Http;
- System.Threading;
- System.Threading.Tasks;

Новий шаблон є надзвичайно зручним для простих завдань, коли вам треба швидко створити просту консольну утиліту і ви не хочете витратити час на написання конструкцій верхнього рівня.

Для складніших завдань (методи, підключення інших просторів імен тощо) доведеться використовувати старий шаблон (я зробила так у завданні 3).

Завдання 2

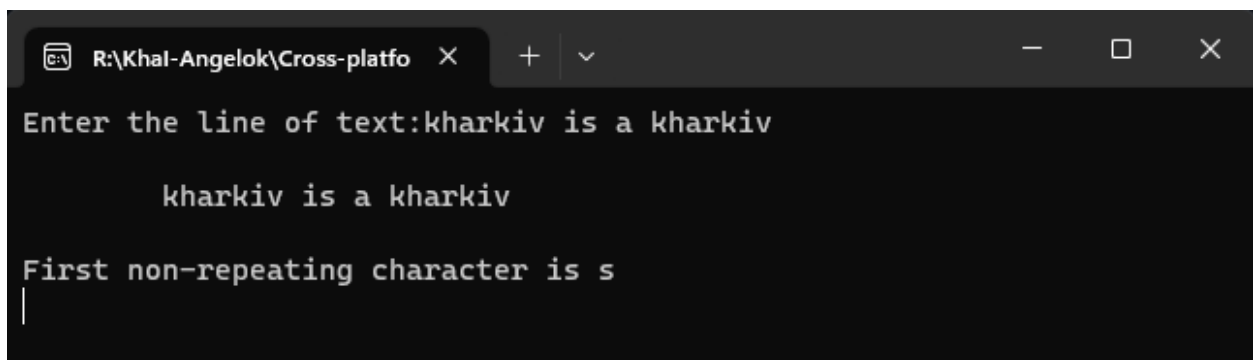
Текст програми

```
// See https://aka.ms/new-console-template for more information

//Entering a line of text
Console.Write("Enter the line of text:");
String? s = Console.ReadLine();
if (string.IsNullOrEmpty(s))
    s = "The Cat in the Hat is Not a Cat";
Console.WriteLine();
s = s.ToLower();
Console.WriteLine($"\\t{s}");

//finding a first non-repeating character
for (int i = 0; i < s.Length; i++)
{
    if (s.IndexOf(s[i], s.IndexOf(s[i]) + 1) == -1)
    {
        Console.WriteLine($"\\nFirst non-repeating character is {s[i]}\\n");
        break;
    }
}
Console.ReadKey();
```

Результат виконання програми



```
R:\Khal-Angelok\Cross-platfo
Enter the line of text:kharkiv is a kharkiv

    kharkiv is a kharkiv

First non-repeating character is s
|
```

Примітка

Робота виконувалась мовою C# у середовищі Visual Studio 2022 Community Edition з використанням нового шаблону консольних застосунків (доступний починаючи з .NET 6). В цьому випадку вам достатньо написати код, який в класичній програмі містився б всередині методу main, а компілятор зробить решту роботи за вас (т.зв. режим top level statements).

При цьому середовище також неявно підключає наступні простори імен:

- System;
- System.IO;
- System.Collections.Generic;
- System.Linq;
- System.Net.Http;
- System.Threading;
- System.Threading.Tasks;

Новий шаблон є надзвичайно зручним для простих завдань, коли вам треба швидко створити просту консольну утиліту і ви не хочете витрачати час на написання конструкцій верхнього рівня.

Для складніших завдань (методи, підключення інших просторів імен тощо) доведеться використовувати старий шаблон (я зробила так у завданні 3).

Завдання 3

Текст програми

```
namespace LlTask3
{
    internal class Program
    {
        //internal variables
        private static Random _random = new Random();
        private static int[] _trials;

        //get one dice value in general case
        static int DiceRoll(int sides) {
            return _random.Next(sides) + 1;
        }

        //get one 6-sided dice value
        static int OneDice()
        {
            return DiceRoll(6);
        }
    }
}
```

```

}

//get two 6-sided dice value
static int TwoDice()
{
    return DiceRoll(6) + DiceRoll(6);
}

//get N K-sided dice value
static int NDice(int N, int K)
{
    int sum = 0;
    for (int i = 0; i < N; i++)
        sum = sum + DiceRoll(K);
    return sum;
}

//draw a histogram
static void hist()
{
    SortedDictionary<uint, int> histogram = new
        SortedDictionary<uint, int>();
    foreach (uint item in _trials)
    {
        if (histogram.ContainsKey(item))
        {
            histogram[item]++;
        }
        else
        {
            histogram[item] = 1;
        }
    }
    foreach (KeyValuePair<uint, int> pair in histogram)
    {
        Console.WriteLine($"{String.Format(
            "{0,4}", pair.Key)}\t{new string('■', pair.Value)} ({pair.Value})");
    }
}

static void Main(string[] args)
{
    Console.Clear();

    //all the variables
    int n;
    int N;
    int K;

    //get number of trials and initialize array
    Console.Write("Enter the number of trials:");
    if (!Int32.TryParse(Console.ReadLine(), out n))
        n = 10;
}

```

```

Console.WriteLine($"Dice will rool {n} times!");
_trials = new int[n];

//create a menu
Console.WriteLine("\nChoose a number of dice:");
Console.WriteLine("\t1 - one dice with 6 sides");
Console.WriteLine("\t2 - two dice with 6 sides");
Console.WriteLine("\t3 - N dice with K sides");
Console.WriteLine("\t4 - exit");
Console.Write("\r\nSelect an option 1,2,3 or 4: ");

//calculate sum and display a histogram
switch (Console.ReadLine())
{
    case "1": //one 6-sided dice
        for (int i = 0; i < n; i++)
            _trials[i] = OneDice();
        hist();
        break;
    case "2": //two 6-sided dice
        for (int i = 0; i < n; i++)
            _trials[i] = TwoDice();
        hist();
        break;
    case "3": //N K-sided dice
        Console.Write("Enter the number of dice:");
        if (!Int32.TryParse(Console.ReadLine(), out N))
            N = 3;
        Console.WriteLine($"You choose a {N} dice!");
        Console.Write("Enter the number of sides:");
        if (!Int32.TryParse(Console.ReadLine(), out K))
            K = 8;
        Console.WriteLine($"Your dice will have {K} sides!");
        for (int i = 0; i < n; i++)
            _trials[i] = NDice(N,K);
        hist();
        break;
    case "4": // exit
        Environment.Exit(0);
        break;
}
Console.ReadKey();
}
}
}

```

Результат виконання програми

```
R:\Khal-Angelok\Cross-platfo  X  +  v  -  □  X

Enter the number of trials:100
Dice will rool 100 times!

Choose a number of dice:
    1 - one dice with 6 sides
    2 - two dice with 6 sides
    3 - N dice with K sides
    4 - exit

Select an option 1,2,3 or 4: 3
Enter the number of dice:4
You choose a 4 dice!
Enter the number of sides:8
Your dice will have 8 sides!

 7  ■ (1)
 8  ■■■■ (4)
 9  ■■■ (3)
10  ■■ (2)
11  ■ (1)
12  ■■■■ (4)
13  ■■■ (3)
14  ■■■■ (5)
15  ■■■■■ (6)
16  ■■■■■■ (8)
17  ■■■■■■■■ (11)
18  ■■■■■■■■ (9)
19  ■■■■■■■■ (9)
20  ■■■■ (5)
21  ■■■■■■ (7)
22  ■■■■ (5)
23  ■■■■ (4)
24  ■■■■ (4)
25  ■■■ (3)
26  ■■■ (3)
27  ■ (1)
29  ■ (1)
32  ■ (1)
```

Примітки

1. Робота виконувалась мовою C# у середовищі Visual Studio 2022 Community Edition з використанням старого шаблону консольних застосунків, оскільки це завдання було трохи складніше за попередні, тож доцільно було розбити код на кілька статичних методів

2. Якщо вже прагнути досконалості 😊, то код завдання потребує рефакторингу аби позбавитись дублювання в альтернативах оператора варіанту.