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 № 1 з дисципліни «Основи програмування-2. Методології програмування» «Перевантаження операторів»

Варіант 31

Виконав студент ІП-13 Сокур Антон Юрійович
(шифр, прізвище, ім'я, по батькові)

Перевірив Вєчерковська Анастасія Сергіївна
(прізвище, ім'я, по батькові)

Київ 2022

Мета – вивчити механізми створення класів з використанням перевантажених операторів (операцій).

Завдання:

31. Розробити клас ««Мультимножина» для представлення множини символів. Реалізувати для нього декілька конструкторів, геттери, метод визначення приналежності заданого елемента множині. Перевантажити оператори "+", "*" та "-" для знаходження об'єднання, перетину і різниці множин відповідно. Створити три множини (A, B, C), використовуючи різні конструктори. На їх основі побудувати множину D=(A∪B)\ C∩B.

Постановка задачі: створюємо клас для представлення множини символів. Реалізуємо декілька конструкторів, гетер та метод визначення приналежності заданого елемента множині. Перевантажуємо оператори “+”, “*” та “-” для знаходження об’єднання, перетину і ризниці множини. Реалізую функцію для побудови множини D.

Код програми

C#

MultiSet.cs

```
namespace csahrp;

public class MultiSet
{
    private char[] symbols { get; set; }

    public void OutputSet()
    {
        var output = "";
        foreach (var chr in symbols)
        {
            output += chr;
        }
        Console.WriteLine(output);
        Console.WriteLine();
    }

    public MultiSet(char[] characters)
    {
        symbols = characters;
    }

    public MultiSet(int a, int b)
    {
        var characters = new char[b - a];
        var index = 0;
        for (var i = a; i < b; i++)
        {
            characters[index] = (char)i;
            index++;
```

```
        }

        symbols = characters;
    }

    public static MultiSet operator +(MultiSet a, MultiSet b)
    {
        var union = a.symbols.Union(b.symbols);
        return new MultiSet(union.ToArray());
    }

    public static MultiSet operator *(MultiSet a, MultiSet b)
    {
        var intersect = a.symbols.Intersect(b.symbols);
        return new MultiSet(intersect.ToArray());
    }

    public static MultiSet operator -(MultiSet a, MultiSet b)
    {
        var diff = a.symbols.Except(b.symbols);
        return new MultiSet(diff.ToArray());
    }

    public bool Includes(char chr)
    {
        foreach (var symbol in symbols)
        {
            if (symbol == chr) return true;
        }

        return false;
    }
}
```

Program.cs

```
namespace csahrp
{
    class Program
    {
        public static MultiSet CalculateD(MultiSet A, MultiSet B, MultiSet C)
        {
            return (A + B) - C * B;
        }

        public static MultiSet CreateMultiset()
        {
            Console.WriteLine("Enter 1 if you want to create multiset manually and 0
if automatically");
            var answer = Console.ReadLine();

            if (answer == "1")
            {
                return CreateManually();
            }

            if (answer == "0")
            {
                return CreateAutomatically();
```

```
        }

        return CreateMultiset();
    }

    public static MultiSet CreateManually()
    {
        int n;
        Console.WriteLine("Enter size of multiset: ");
        int.TryParse(Console.ReadLine(), out n);
        if (n > 0)
        {
            var array = new char[n];
            for (var i = 0; i < n; i++)
            {
                array[i] = Console.ReadKey().KeyChar;
            }

            return new MultiSet(array);
        }
        return CreateManually();
    }

    public static MultiSet CreateAutomatically()
    {
        int a, b;
        Console.WriteLine("Enter character range: ");
        Console.WriteLine("a = ");
        int.TryParse(Console.ReadLine(), out a);
        Console.WriteLine("b = ");
        int.TryParse(Console.ReadLine(), out b);

        if (a < b)
        {
            return new MultiSet(a, b);
        }
        return CreateAutomatically();
    }

    public static void CheckCharacters(MultiSet D)
    {
        char key = Console.ReadKey().KeyChar;
        if (key == (char)7)
        {
            return;
        }
        Console.WriteLine(D.Includes(key) ? " - includes" : " - excludes");
        CheckCharacters(D);
    }

    static void Main(string[] args)
    {
        var A = CreateMultiset();
        Console.WriteLine("\nA:");
        A.OutputSet();
        var B = CreateMultiset();
        Console.WriteLine("\nB:");
        B.OutputSet();
        var C = CreateMultiset();
        Console.WriteLine("\nC:");
        C.OutputSet();
        var D = CalculateD(A, B, C);
```

```
            Console.WriteLine("D:");
            D.OutputSet();
            Console.WriteLine("Enter character to see if it is in multiset D (enter
CTRL + G to complete):");
            CheckCharacters(D);
        }

    }
}
```

Робота програми на C#

Висновок: під час виконання даної лабораторної роботи я вивчив особливості створення класів з використання перевантажених операторів у мові C#