Московский государственный технический университет им. Н.Э. Баумана.

Факультет «Информатика и управление»

Кафедра ИУ5. Курс «Базовые компоненты интернет-технологий»

Отчет по лабораторной работе №6

Выполнил:

студент группы ИУ5-31

Бондаренко Иван

Подпись и дата:

Проверил:

преподаватель каф. ИУ5

Гапанюк Ю. Е.

Подпись и дата:

Москва, 2020 г.

Описание задания:

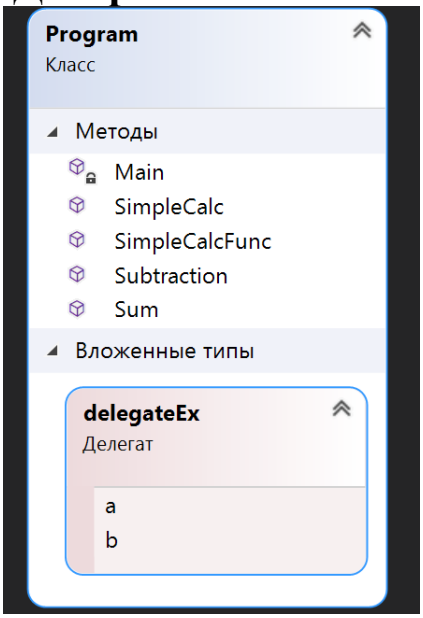
Часть 1.

1) Программа должна быть разработана в виде консольного приложения на языке C#.
2) Определите делегат, принимающий несколько параметров различных типов и возвращающий значение произвольного типа.
3) Напишите метод, соответствующий данному делегату.
4) Напишите метод, принимающий разработанный Вами делегат, в качестве одного из входным параметров. Осуществите вызов метода, передавая в качестве параметра-делегата:
   - метод, разработанный в пункте 3;
   - лямбда-выражение.
5) Повторите пункт 4, используя вместо разработанного Вами делегата, обобщенный делегат Func< > или Action< >, соответствующий сигнатуре разработанного Вами делегата.

Часть 2.

1) Программа должна быть разработана в виде консольного приложения на языке C#.
2) Создайте класс, содержащий конструкторы, свойства, методы.
3) С использованием рефлексии выведите информацию о конструкторах, свойствах, методах.
4) Создайте класс атрибута (унаследован от класса System.Attribute).
5) Назначьте атрибут некоторым свойствам классам. Выведите только те свойства, которым назначен атрибут.
6) Вызовите один из методов класса с использованием рефлексии.

Часть 1:
Диаграмма классов:



Текст программы:

1. Program.cs

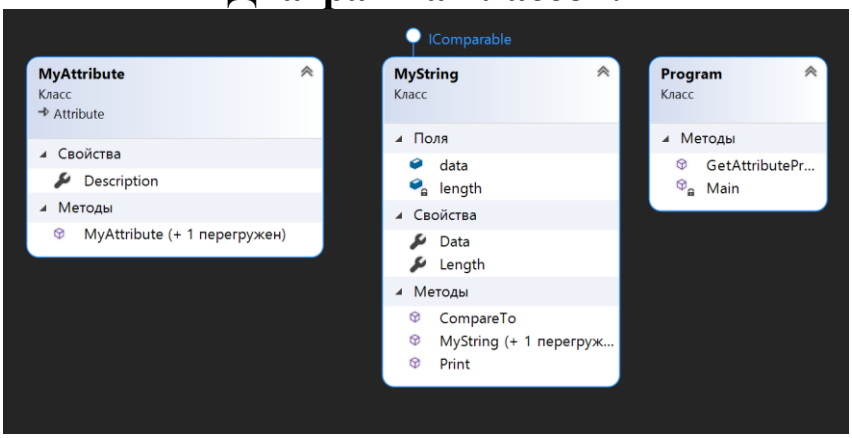
```
1.  using System;
2.
3.  namespace Lab6
4.  {
5.      class Program
6.      {
7.          public delegate int delegateEx(int a, int b);
8.
9.          public static int Sum(int a, int b)
10.         {
11.             return a + b;
12.         }
13.         public static int Subtraction(int a, int b)
14.         {
15.             return a - b;
16.         }
17.
18.         public static int SimpleCalc(int a, int b, delegateEx deg)
19.         {
20.             return deg.Invoke(a, b);
21.         }
22.
23.         public static int SimpleCalcFunc(int a, int b, Func<int,int,int>
    deg)
24.         {
25.             return deg.Invoke(a, b);
26.         }
27.
28.         static void Main(string[] args)
29.         {
30.             int a, b;
31.             Console.ForegroundColor = ConsoleColor.Green;
32.             Console.WriteLine("\tДелегаты");
33.             Console.ResetColor();
34.             while (true)
35.             {
36.                 Console.Write("Введите число №1: ");
37.                 string line = Console.ReadLine();
38.                 if (!Int32.TryParse(line, out a))
```

```
39.                 {
40.                     Console.WriteLine("Ошибка ввода!");
41.                 }
42.                 else
43.                     break;
44.             }
45.             while (true)
46.             {
47.                 Console.Write("Введите число №2: ");
48.                 string line = Console.ReadLine();
49.                 if (!Int32.TryParse(line, out b))
50.                 {
51.                     Console.WriteLine("Ошибка ввода!");
52.                 }
53.                 else
54.                     break;
55.             }
56.
57.             Console.Write($"Функция в качестве аргумента: {a} + {b} = ");

58.             Console.WriteLine(SimpleCalc(a, b, Sum));
59.             Console.Write($"Лямбда-
    выражение в качестве аргумента: {a} + {b} = ");
60.             Console.WriteLine(SimpleCalc(a, b, (m, n) => (m + n)));
61.             Console.Write($"Func<> в качестве аргумента: {a} + {b} = ");

62.             Console.WriteLine(SimpleCalcFunc(a, b, Sum));
63.         }
64.     }
65. }
```

Пример выполнения программы:

```
        Делегаты
Введите число №1: 34
Введите число №2: 23
Функция в качестве аргумента: 34 + 23 = 57
Лямбда-выражение в качестве аргумента: 34 + 23 = 57
Func<> в качестве аргумента: 34 + 23 = 57
Для продолжения нажмите любую клавишу . . .
```

Часть 2:
Диаграмма классов:



Текст программы:

```
using System;
using System.Reflection;
using System.Collections.Generic;

namespace Lab6._2
{
    [AttributeUsage(AttributeTargets.Property, AllowMultiple = false,
        Inherited = false)]
    public class MyAttribute : System.Attribute
            {
                public string Description { get; set; }
                public MyAttribute()
                {

                }
                public MyAttribute(string Description)
                {
                    this.Description = Description;
                }
            }

            public class MyString : IComparable
            {
                public char[] data;
                [MyAttribute(Description = "Атрибут MyAttribute: важная информация")]
                public char[] Data
                {
                    get { return data; }
                    set { data = null; }
                }

                private int length;

                public int Length { get; private set; }

                public MyString()
                {
                    data = null;
                    length = 0;
                }

                public MyString(char[] str)
                {
                    length = str.Length;
                    data = new char[length];
                    int i = 0;
                    foreach (char a in str)
                    {
                        data[i] = a;
                        i++;
                    }
                }

                public int CompareTo(object obj)
                {
                    if (obj.GetType().Name == "MyString")
                    {
                        foreach(var field in obj.GetType().GetFields())
                        {
                            if (field.GetValue(this) != field.GetValue(obj))
                                return 1;
                        }
                        return 0;
                    }
                    return 1;
                }
```

```
67.
68.                 public void Print()
69.                 {
70.                     for (int i = 0; i < length; i++)
71.                         Console.Write(data[i]);
72.                 }
73.             }
74.
75.             class Program
76.             {
77.                 public static bool GetAttributeProperty(PropertyInfo propertyInfo,
78.         Type type, out object att)
79.                 {
80.                     bool res = false;
81.                     att = null;
82.                     var isAtt = propertyInfo.GetCustomAttributes(type, false);
83.                     if (isAtt.Length > 0)
84.                     {
85.                         res = true;
86.                         att = isAtt[0];
87.                     }
88.                     return res;
89.                 }
90.                 static void Main(string[] args)
91.                 {
92.
93.                     Console.ForegroundColor = ConsoleColor.Green;
94.                     Console.WriteLine("\tРефлексия");
95.                     Console.ResetColor();
96.
97.                     Assembly i = Assembly.GetExecutingAssembly();
98.                     Console.WriteLine("Информация о сборке:");
99.                     Console.WriteLine(i.FullName + '\n');
100.                    Console.WriteLine("Место расположения сборки:");
101.                    Console.WriteLine(i.Location + '\n');
102.
103.                    Console.ForegroundColor = ConsoleColor.Green;
104.                    Console.WriteLine("\tИнформация о типе");
105.                    Console.ResetColor();
106.
107.                    Type type = typeof(MyString);
108.
109.                    Console.WriteLine("Пространство имен: " + type.Namespace);
110.                    Console.WriteLine("Наследование: " + type.BaseType.FullName);
111.                    Console.WriteLine("Сборка: " + type.AssemblyQualifiedName);
112.
113.                    Console.ForegroundColor = ConsoleColor.Green;
114.                    Console.WriteLine("\nВывод конструкторов:");
115.                    Console.ResetColor();
116.
117.                    foreach (var construct in type.GetConstructors())
118.                    {
119.                        Console.WriteLine("\t" + construct);
120.                    }
121.
122.                    Console.ForegroundColor = ConsoleColor.Green;
123.                    Console.WriteLine("\nВывод методов:");
124.                    Console.ResetColor();
125.                    foreach (var method in type.GetMethods())
126.                    {
127.                        Console.WriteLine("\t" + method.Name);
128.                    }
129.
130.                    Console.ForegroundColor = ConsoleColor.Green;
131.                    Console.WriteLine("\nВывод свойств:");
132.                    Console.ResetColor();
133.                    foreach (var prop in type.GetProperties())
134.                    {
135.                        Console.WriteLine("\t" + prop.PropertyType.ToString() + " " +
```

```
136.      + prop.Name);
137.                  }
138.
139.                  Console.ForegroundColor = ConsoleColor.Green;
140.                  Console.WriteLine("\nВывод public полей:");
141.                  Console.ResetColor();
142.                  foreach (var prop in type.GetFields())
143.                  {
144.                      Console.WriteLine("\t" + prop.Name);
145.                  }
146.
147.                  Console.WriteLine("\nMyString реализует IComparable > " +
148.      + new HashSet<Type>(type.GetInterfaces()).Contains(typeof(IComparable)));
149.
150.                  Console.ForegroundColor = ConsoleColor.Green;
151.                  Console.WriteLine("\nВывод свойств с атрибутом:");
152.                  Console.ResetColor();
153.                  foreach (var x in type.GetProperties())
154.                  {
155.                      object attrObj;
156.                      if (GetAttributeProperty(x, typeof(MyAttribute), out attrObj))
157.                      {
158.                          MyAttribute attr = attrObj as MyAttribute;
159.                          Console.WriteLine(x.Name + " - " + attr.Description);
160.                      }
161.                  }
162.
163.                  Console.ForegroundColor = ConsoleColor.Green;
164.                  Console.WriteLine("\nВызов метода:");
165.                  Console.ResetColor();
166.                  char[] arr = { '1', '2', '3', '4' };
167.                  MyString str = new MyString(arr);
168.                  type.InvokeMember("Print", BindingFlags.InvokeMethod, null, str,
169.      new object[] { });
170.                  Console.ReadLine();
171.              }
172.          }
173.      }
```

Пример выполнения программы: