МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ Н.Э.БАУМАНА

Факультет: «Информатика и системы управления»

Кафедра: «Системы обработки информации и управления (ИУ5)»

Отчет по лабораторной работе №6

По дисциплине «Базовые компоненты интернет-технологий»

Выполнил: Тураев Глеб Вадимович

Группа: ИУ5Ц-51Б

Преподаватель: Гапанюк Юрий Евгеньевич

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата: 25.12.2018

Подпись:

**Описание задания:**

Часть 1. Разработать программу, использующую делегаты.

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. Разработать программу, реализующую работу с рефлексией.

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.

**Текст программы:**

*Program.cs*

using System;

using System.Reflection;

namespace LR6

{

class Program

{

public delegate string Stringing(int x, double y);

static void Main(string[] args)

{

// Delegate Testing

Apply(MakeStringSum, 20, 25.5);

Apply((x, y) => $"{x} - {y} = {(x - y)}", 20, 25.5);

// Func<> Testing

ApplyFunc(MakeStringSum, 20, 25.5);

ApplyFunc((x, y) => $"{x} - {y} = {(x - y)}", 20, 25.5);

Console.WriteLine();

// Reflection Testing

var aInfo = typeof(A);

var propertyInfos = aInfo.GetProperties();

var methodInfos = aInfo.GetMethods();

var constructorInfos = aInfo.GetConstructors();

Console.WriteLine("Properties:");

foreach (var info in propertyInfos)

{

Console.WriteLine(info);

}

Console.WriteLine();

Console.WriteLine("Methods:");

foreach (var info in methodInfos)

{

Console.WriteLine(info);

}

Console.WriteLine();

Console.WriteLine("Constructors:");

foreach (var info in constructorInfos)

{

Console.WriteLine(info);

}

Console.WriteLine();

Console.WriteLine("Attributed Properties:");

foreach (var info in propertyInfos)

{

if (Attribute.IsDefined(info, typeof(MyAttribute)))

{

Console.WriteLine(info);

}

}

Console.WriteLine();

Console.WriteLine("Call MethodString:");

Console.WriteLine(aInfo.GetMethod("MethodString").Invoke(new A(17, 24.34, "25"), null));

}

public static void ApplyFunc(Func<int, double, string> f, int param1, double param2)

{

Console.WriteLine(f(param1, param2));

}

public static void Apply(Stringing f, int param1, double param2)

{

Console.WriteLine(f(param1, param2));

}

public static string MakeStringSum(int x, double y)

{

return $"{x} + {y} = {(x + y)}";

}

}

}

*MyAttribute.cs*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LR6

{

public class MyAttribute : System.Attribute

{

public string Role { get; set; }

public MyAttribute(string role)

{

Role = role;

}

}

}

*A.cs*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LR6

{

public class A

{

[My("Attributed")]

public int PropertyInt { get; set; }

public double PropertyDouble { get; set; }

[My("Attributed")]

public string PropertyString { get; set; }

public A(int propertyInt, double propertyDouble, string propertyString)

{

PropertyInt = propertyInt;

PropertyDouble = propertyDouble;

PropertyString = propertyString;

}

public int MethodInt()

{

return 17;

}

public double MethodDouble()

{

return 24.34;

}

public string MethodString()

{

return "25";

}

}

}

**Результат работы:**

