#### Московский государственный технический университет имени Н.Э. Баумана

## Лабораторная работа по дисциплине БКИТ №3

Выполнил: Байбарин Р.Г., ИУ5-34, 18.10.2017

### I. Описание задания

**Часть 1. Разработать программу, использующую делегаты.**

(В качестве примера можно использовать проект «Delegates»).

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.**

(В качестве примера можно использовать проект «Reflection»).

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.

### 2. Код программы

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Reflection;

using System.Collections;

namespace Lab\_6

{

class Program

{

static void Main(string[] args)

{

int a = 10;

int b = 5;

Console.WriteLine("Делегаты\n" +

"Имеем a = 10, b = 5");

Del.PlusOrMinusMethod("Параметр - лямбда-выражение(вычитание): ", a, b,

(int x, int y) =>

{

int z = x - y;

return z;

}

);

Del.PlusOrMinusMethod("Параметр - делегат минус: ", a, b, Del.Minus);

Del.PlusOrMinusMethodFunc("Параметр - лямбда-выражение(сложение): ", a, b,

(int x, int y) =>

{

int z = x + y;

return z;

}

);

Del.PlusOrMinusMethodFunc("Параметр - делегат плюс: ", a, b, Del.Plus);

Console.WriteLine("-------------------------------");

Ref reef = new Ref();

reef.Part1();

Console.ReadKey();

}

}

public delegate int PlusOrMinus(int p1, int p2);

class Del

{

//Методы, реализующие делегат (методы "типа" делегата)

public static int Plus(int p1, int p2) { return p1 + p2; }

public static int Minus(int p1, int p2) { return p1 - p2; }

static public void PlusOrMinusMethodFunc(string str, int i1, int i2, Func<int, int, int> PlusOrMinusParam) // Использование обощенного делегата Func<>

{

int Result = PlusOrMinusParam(i1, i2);

Console.WriteLine(str + Result.ToString());

}

static public void PlusOrMinusMethod(string str, int i1, int i2, PlusOrMinus PlusOrMinusParam) /// Использование делегата

{

int Result = PlusOrMinusParam(i1, i2);

Console.WriteLine(str + Result.ToString());

}

}

class Ref

{

public void Part1()

{

Console.Write("Рефлексия\n");

Type type = System.Type.GetType("System.Int32");

var bro = Activator.CreateInstance(type);

int i = 0;

Console.Write("Создана переменная bro неопределенного типа (изначально типа System.Int32). \n" +

"Присвоем ей значение другого типа. Выберите его из пунктов 1-4\n" +

"1) Char\n" +

"2) Int32\n" +

"3) String\n" +

"4) Boolean\n" +

"Ваш выбор: ");

bool f = int.TryParse(Console.ReadLine(), out i);

if (!f)

throw new ArgumentException();

switch (i)

{

case 1:

{

bro = 'c';

Console.WriteLine("bro успешно присвоено значение 'с' (bro = " + bro.ToString() + ")");

Console.WriteLine("Тип бро: " + bro.GetType().ToString());

}

break;

case 2:

{

bro = 100;

Console.WriteLine("bro успешно присвоено значение 100 (bro = " + bro.ToString() + ")");

Console.WriteLine("Тип бро: " + bro.GetType().ToString());

}

break;

case 3:

{

bro = "Big Bro";

Console.WriteLine("bro успешно присвоено значение \"Big Bro\" (bro = " + bro.ToString() + ")");

Console.WriteLine("Тип бро: " + bro.GetType().ToString());

}

break;

case 4:

{

bro = true;

Console.WriteLine("bro успешно присвоено значение true (bro = " + bro.ToString() + ")");

Console.WriteLine("Тип бро: " + bro.GetType().ToString());

}

break;

default:

{

Console.WriteLine("Ошибка ввода!");

}

break;

}

Console.WriteLine("Создадим новую переменную такого же типа, что и bro");

var newbro = Activator.CreateInstance(System.Type.GetType("System.Int32"));

ArrayList myList = new ArrayList();

myList.Add('c');

myList.Add(10);

myList.Add(" - best Bro :)");

myList.Add(false);

type = bro.GetType();

foreach (object cl in myList)

{

if (type == cl.GetType())

{

newbro = cl;

Console.WriteLine("new\_bro = {0}", newbro);

}

}

if (type.ToString() == "System.String")

{ bro = (string)bro + (string)newbro; }

else if (type.ToString() == "System.Int32")

{ bro = (int)bro + (int)newbro; }

else if (type.ToString() == "System.Char")

{ bro = (char)bro + (char)newbro; }

else

{

if (((bool)bro == false) && ((bool)newbro == false))

bro = false;

else bro = true;

}

Console.WriteLine("bro + new\_bro = {0}", bro);

Type t = bro.GetType();

Console.WriteLine("\nТип " + t.FullName + " унаследован от " + t.BaseType.FullName);

Console.WriteLine("Пространство имен " + t.Namespace);

Console.WriteLine("Находится в сборке " + t.AssemblyQualifiedName);

Console.WriteLine("\nКонструкторы:");

foreach (var x in t.GetConstructors())

{

Console.WriteLine(x);

}

Console.WriteLine("\nМетоды:");

foreach (var x in t.GetMethods())

{

Console.WriteLine(x);

}

Console.WriteLine("\nСвойства:");

foreach (var x in t.GetProperties())

{

Console.WriteLine(x);

}

Console.WriteLine("\nПоля данных (public):");

foreach (var x in t.GetFields())

{

Console.WriteLine(x);

}

}

}

}

### 3. Примеры работы программы



