|  |  |  |
| --- | --- | --- |
| Защищено:  Гапанюк Ю.Е.    "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_2020 г. |  | Демонстрация:  Гапанюк Ю.Е.  "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_2020 г. |

**Отчет по лабораторной работе № 6 по курсу**

**Базовые компоненты интернет-технологий**

**ГУИМЦ**

#### Тема работы: " Делегаты и рефлексия "

10

(количество листов)

Вариант № **4**

|  |  |
| --- | --- |
| ИСПОЛНИТЕЛЬ: |  |
| студент группы ИУ5Ц-52Б | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  | (подпись) |
| Чиварзин А.Е. | "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_2020 г. |

Москва, МГТУ - 2020

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

**СОДЕРЖАНИЕ**

[1. Описание задания 3](#_Toc50574201)

[2. Диаграмма классов 3](#_Toc50574202)

[3. Текст программы 3](#_Toc50574203)

[4. Результаты выполнения программы 8](#_Toc50574204)

# 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.**

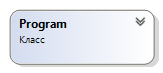
(В качестве примера можно использовать проект «Reflection»).

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

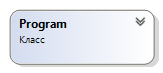
# Диаграмма классов

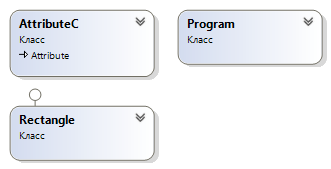
Поскольку проект разделён на 3 независимых модуля, то приводится 3 независимые диаграммы:

**Модуль LAB\_6**



**Модуль testFuncProject**



**Модуль Part2**  


# Текст программы

**LAB\_6🡪Program.cs**

using System;

namespace LAB\_6

{

class Program

{

delegate bool CorrectObject(int ID, double dID);

static bool oEquals(int i1, double d2)

{

return Convert.ToDouble(i1) == d2;

}

static void test(int i1, double d1, string str, CorrectObject co)

{

Console.WriteLine(str + "//// Сравниваем числя");

bool coResult = co(i1, d1);

if (coResult)

Console.WriteLine("Равны");

else

Console.WriteLine("НЕ равны");

}

////////////////////////////////////////////

static void Main(string[] args)

{

Console.WriteLine("Чиварзин А. Е. ИУ5Ц-52Б\n");

Console.WriteLine("Через метод");

test(1, 1.0, "12345", oEquals);

Console.WriteLine("\nЛямбда-выражение");

CorrectObject coL = (int i, double d) =>

{

bool res = i + 100 == d - 200;

return res;

};

Console.WriteLine(coL(-100, 200.0));

}

}

}

**testFuncProject🡪Program.cs**

using System;

namespace testFuncProject

{

class Program

{

static Func<int, double, bool> funcTest;

static bool test(int i1, double d1)

{

Console.WriteLine("//// Сравниваем числя");

return i1 == d1;

}

static void Main(string[] args)

{

Console.WriteLine("Через метод");

funcTest = test;

if (funcTest(1, 1.0))

Console.WriteLine("Равны");

else

Console.WriteLine("НЕ равны");

Console.WriteLine("\nЛямбда-выражение");

funcTest = (int i, double d) =>

{

bool res = i + 100 == d - 200;

return res;

};

if (funcTest(1, 1.0))

Console.WriteLine("Равны");

else

Console.WriteLine("НЕ равны");

}

}

}

**Part2🡪Program.cs**

using System;

using System.Linq;

using System.Reflection;

namespace Part2

{

class Program

{

/// <summary>

/// Проверка, что у свойства есть атрибут заданного типа

/// </summary> /// <returns>Значение атрибута</returns>

public static bool GetPropertyAttribute(PropertyInfo checkType, Type attributeType, out object attribute)

{

bool Result = false; attribute = null;

//Поиск атрибутов с заданным типом

var isAttribute = checkType.GetCustomAttributes(attributeType, false);

if (isAttribute.Length > 0)

{

Result = true;

attribute = isAttribute[0];

}

return Result;

}

// ////////////////////////////////////////////////////////

static void Main(string[] args)

{

Console.WriteLine("Чиварзин А. Е. ИУ5Ц-52Б\n");

Rectangle obj = new Rectangle(1, 2);

Type t = obj.GetType();

Console.WriteLine("\nИнформация о типе:");

Console.WriteLine("Тип " + 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);

}

Console.WriteLine("\nRectangle реализует IComparable -> " + t.GetInterfaces().Contains(typeof(IComparable)));

Console.WriteLine("\n========================================\n");

Console.WriteLine("\nСвойства, помеченные атрибутом:");

foreach (var x in t.GetProperties())

{

object lab6;

if (GetPropertyAttribute(x, typeof(AttributeC), out lab6))

{

AttributeC attr = lab6 as AttributeC;

Console.WriteLine(x.Name + " - " + attr.Description);

}

}

Console.WriteLine("\n========================================\n");

Console.WriteLine("\nВызов метода:");

//Создание объекта //ForInspection fi = new ForInspection(); //Можно создать объект через рефлексию

Rectangle fi = (Rectangle)t.InvokeMember(null, BindingFlags.CreateInstance, null, null, new object[] { });

//Параметры вызова метода

object[] parameters = new object[] { 1, 2 };

//Вызов метода

object Result = t.InvokeMember("setAB", BindingFlags.InvokeMethod,null, fi, parameters);

Console.WriteLine("setAB(1,2)={0}", Result);

}

}

}

//Деасемблирование CTRL + ALT + D в режиме отладки

**Part2🡪Rectangle.cs**

using System;

using System.Collections.Generic;

using System.Text;

namespace Part2

{

/// <summary>

/// Прямоугольник

/// </summary>

public class Rectangle : IComparable

{

int a;

int b;

public double test = 12.345;

public Rectangle()

{

a = 0;

b = 0;

}

public Rectangle(int a, int b)

{

this.a = a;

this.b = b;

}

[AttributeC("12345")]

public int CompareTo(object obj)

{

int s = a \* b;

return CompareTo(s);

}

public string setAB(int a, int b)

{

this.a = a;

this.b = b;

return "Сумма длинны и ширины = " + (a + b);

}

[AttributeC(Description = "Периметр")]

public int perimetr

{

get => 2 \* a + 2 \* b;

//set => throw new NotImplementedException(); // set не требуется

}

public override string ToString()

{

return "Длинна = " + a + " Шмрина = " + b;

}

}

}

**Part2🡪AttributeC.cs**

using System;

using System.Collections.Generic;

using System.Text;

namespace Part2

{

[AttributeUsage(AttributeTargets.All, AllowMultiple = false, Inherited = false)]

class AttributeC : Attribute

{

public string Description

{

get;

set;

}

public AttributeC() {}

public AttributeC(string descriptionParam)

{

Description = descriptionParam;

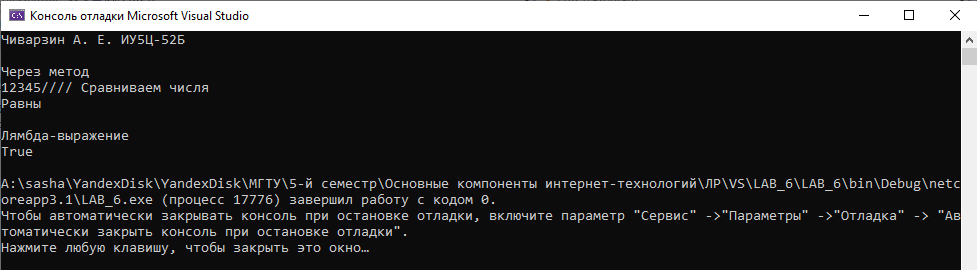
}

}

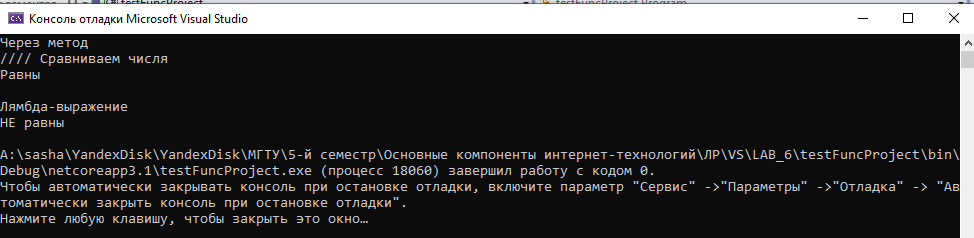
}

# Результаты выполнения программы

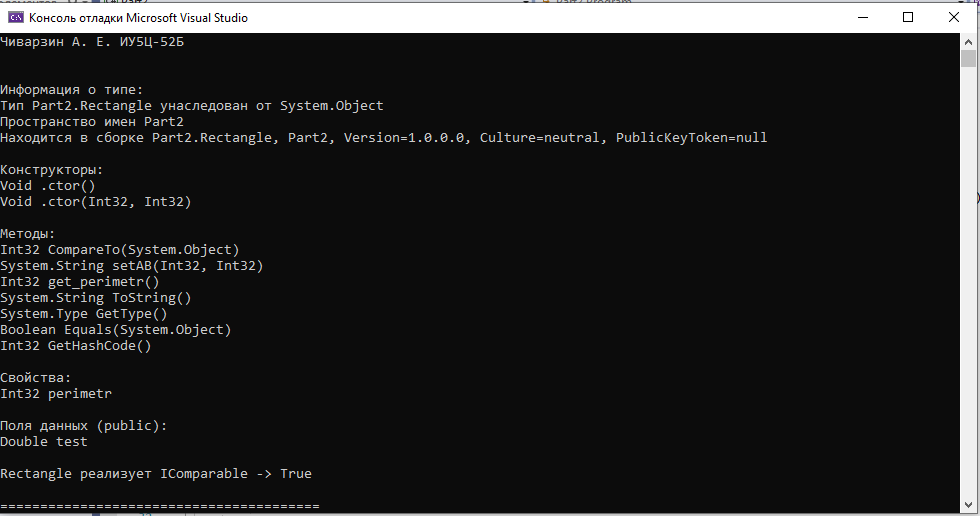
**Модуль LAB\_6**



**Модуль testFuncProject**



**Модуль Part2**



Чиварзин А. Е. ИУ5Ц-52Б

Информация о типе:

Тип Part2.Rectangle унаследован от System.Object

Пространство имен Part2

Находится в сборке Part2.Rectangle, Part2, Version=1.0.0.0, Culture=neutral, PublicKeyToken=null

Конструкторы:

Void .ctor()

Void .ctor(Int32, Int32)

Методы:

Int32 CompareTo(System.Object)

System.String setAB(Int32, Int32)

Int32 get\_perimetr()

System.String ToString()

System.Type GetType()

Boolean Equals(System.Object)

Int32 GetHashCode()

Свойства:

Int32 perimetr

Поля данных (public):

Double test

Rectangle реализует IComparable -> True

========================================

Свойства, помеченные атрибутом:

perimetr - Периметр

========================================

Вызов метода:

setAB(1,2)=Сумма длинны и ширины = 3

A:\sasha\YandexDisk\YandexDisk\МГТУ\5-й семестр\Основные компоненты интернет-технологий\ЛР\VS\LAB\_6\Part2\bin\Debug\netcoreapp3.1\Part2.exe (процесс 8260) завершил работу с кодом 0.

Чтобы автоматически закрывать консоль при остановке отладки, включите параметр "Сервис" ->"Параметры" ->"Отладка" -> "Автоматически закрыть консоль при остановке отладки".

Нажмите любую клавишу, чтобы закрыть это окно…