

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

Факультет «Информатика и системы управления»
Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Парадигмы и конструкции языков программирования»
Отчет по лабораторной работе №6

Выполнил:
Студент группы ИУ5-31Б
Куртинец Роман

Проверил:
Гапанюк Ю. Е.

2025 г.

Задание:

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.

Листинг программы:

MyClass.cs

```
public class MyAttribute : Attribute { }

public class MyClass
{
    public int Id { get; set; }
}
```

```

[MyAttribute]
public string Name { get; set; }

[MyAttribute]
public double Value { get; set; }

public MyClass() { }

public MyClass(int id, string name, double value)
{
    Id = id;
    Name = name;
    Value = value;
}

public void PrintInfo()
{
    Console.WriteLine($"Id: {Id}, Name: {Name}, Value: {Value}");
}

public int Add(int a, int b)
{
    return a + b;
}
}

```

Program.cs

```

using System;
using System.Reflection;

class Program
{
    public delegate string GameInfoDel(string gameName, int gameReleaseYear);

    static string GameInfo(string gameName, int gameReleaseYear)
    {
        string result;
        if (gameReleaseYear == 2014 || gameReleaseYear == 1998 ||
gameReleaseYear == 2025 || gameReleaseYear == 2007)
        {
            result = $"{gameName} released in good time";
        }
        else
        {
            result = $"{gameName} released in bad time";
        }

        return result;
    }
}

```

```

static void AskGame(string gameName, int gameReleaseYear, GameInfoDel game)
{
    string result = game(gameName, gameReleaseYear);
    System.Console.WriteLine(result);
}

static void AskGameFunc5(string gameName, int gameReleaseYear, Func<string,
int, string> game)
{
    string result = game(gameName, gameReleaseYear);
    System.Console.WriteLine(result);
}

static void AskGameFunc(string gameName, int gameReleaseYear, GameInfoDel
game)
{
    string result = game(gameName, gameReleaseYear);
    System.Console.WriteLine(result);
}

static void Main()
{
    System.Console.WriteLine("Введите название игры и год её релиза");
    string gameName = Console.ReadLine();
    int gameReleaseYear = Int32.Parse(Console.ReadLine());

    System.Console.WriteLine("\nРезультат с использованием метода
GameInfo:");
    GameInfoDel game = GameInfo;
    AskGame(gameName, gameReleaseYear, game);

    System.Console.WriteLine("\nРезультат с использованием лямбда-
выражения:");
    AskGame(gameName, gameReleaseYear, (name, year) =>
    {
        if (year % 2 == 0) return $"{name} released in even year!";
        else return $"{name} released in odd year!";
    });

    System.Console.WriteLine("\nРезультат с использованием метода GameInfo
(через Func<>:");
    AskGameFunc5(gameName, gameReleaseYear, GameInfo);

    System.Console.WriteLine("\nРезультат с использованием лямбда-выражения
(через Func<>:");
    AskGameFunc5(gameName, gameReleaseYear, (name, year) =>
    {
        if (year % 2 == 0) return $"{name} released in even year!";
        else return $"{name} released in odd year!";
    });
}

```

```

});

MyClass obj = new MyClass(1, "Example", 42.5);

Type type = typeof(MyClass);

for (int i = 0; i < 150; i++) System.Console.Write('=');
System.Console.WriteLine("\nКонструкторы:");

foreach (var ctor in type.GetConstructors())
{
    System.Console.WriteLine(ctor);
}

System.Console.WriteLine("\nСвойства:");
foreach (var prop in type.GetProperties())
{
    System.Console.WriteLine(prop);
}

System.Console.WriteLine("\nМетоды:");
foreach (var method in type.GetMethods(BindingFlags.Public |
BindingFlags.Instance | BindingFlags.DeclaredOnly))
{
    System.Console.WriteLine(method);
}

System.Console.WriteLine("\nСвойства с атрибутом MyAttribute:");
foreach (var prop in type.GetProperties())
{
    if (prop.GetCustomAttribute<MyAttribute>() != null)
    {
        System.Console.WriteLine(prop);
    }
}

System.Console.WriteLine("\nВызов метода PrintInfo с использованием
рефлексии:");
MethodInfo printMethod = type.GetMethod("PrintInfo");
printMethod.Invoke(obj, null);

System.Console.WriteLine("\nВызов метода Add с использованием
рефлексии:");
MethodInfo addMethod = type.GetMethod("Add");
object result = addMethod.Invoke(obj, new object[] { 5, 3 });
System.Console.WriteLine($"Результат: {result}");
}
}

```

Результат выполнения:

Введите название игры и год её релиза

ac1

2007

Результат с использованием метода GameInfo:

ac1 released in good time

Результат с использованием лямбда-выражения:

ac1 released in odd year!

Результат с использованием метода GameInfo (через Func<>):

ac1 released in good time

Результат с использованием лямбда-выражения (через Func<>):

ac1 released in odd year!

=====
Конструкторы:

Void .ctor()

Void .ctor(Int32, System.String, Double)

Свойства:

Int32 Id

System.String Name

Double Value

Методы:

Int32 get_Id()

Void set_Id(Int32)

System.String get_Name()

Void set_Name(System.String)

Double get_Value()

Void set_Value(Double)

Void PrintInfo()

Int32 Add(Int32, Int32)

Свойства с атрибутом MyAttribute:

System.String Name

Double Value

Вызов метода PrintInfo с использованием рефлексии:

Id: 1, Name: Example, Value: 42.5

Вызов метода Add с использованием рефлексии:

Результат: 8

