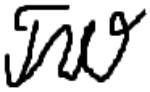
Московский государственный технический университет
им. Н.Э. Баумана

Факультет “Информатика и системы управления”
Кафедра “Системы обработки информации и управления”

Дисциплина «Парадигмы и конструкции языков программирования»

Отчет по лабораторной работе №6



Выполнил:
студент группы ИУ5-36Б
Германенко И. М.
Преподаватель:
Гапанюк Ю.Е.

Москва, 2025

Описание задания:

- Использовать библиотеку **FParsec** для парсинга текста;
- Результат разбора — значение **алгебраического типа**;
- Применить парсер-комбинаторы (>>., .>>, <|>, many, и др.).

Листинг кода:

DelegatesPart/Program.cs:

```
using System;

namespace DelegatesPart
{
    // Объявляем делегат
    public delegate int MathOperation(int x, int y);

    class Program
    {
        // Метод, соответствующий делегату
        static int Add(int x, int y) => x + y;
        static int Multiply(int x, int y) => x * y;

        // Метод, принимающий делегат как параметр
        static void Calculate(int a, int b, MathOperation op)
        {
            Console.WriteLine($"Результат операции: {op(a, b)}");
        }

        // То же самое, но с обобщенным делегатом Func
        static void CalculateFunc(int a, int b, Func<int, int, int> op)
        {
            Console.WriteLine($"Результат Func: {op(a, b)}");
        }

        static void Main(string[] args)
        {
            Console.WriteLine("--- Делегаты ---");

            // 1. Использование именованного метода
            Calculate(10, 20, Add);

            // 2. Использование лямбда-выражения
            Calculate(5, 5, (x, y) => x - y);

            Console.WriteLine("\n--- Обобщенные делегаты (Func) ---");

            // 3. Func с лямбдой
            CalculateFunc(6, 7, (x, y) => x * y);

            // 4. Action (ничего не возвращает)
            Action<string> print = msg => Console.WriteLine($"[LOG]: {msg}");
```

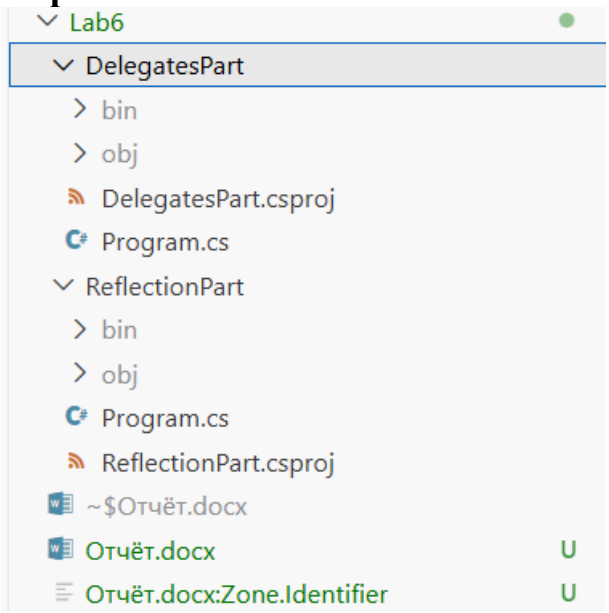
```
            print("Тест Action завершен.");
        }
    }
}
```

Program.cs:

```
using System;
using System.Reflection;

namespace ReflectionPart
{
    // Класс для исследования
    class User
    {
        public string Name { get; set; }
        public int Age { get; set; }
        private string _secret = "12345";

        public User(string name, int age) { Name = name; Age = age; }
        public void SayHello() => Console.WriteLine($"Привет, я {Name}!");
        private void RevealSecret() => Console.WriteLine($"Секрет: {_secret}");
    }

    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("--- Рефлексия ---");
            Type t = typeof(User);

            Console.WriteLine("Свойства класса:");
            foreach (var prop in t.GetProperties())
                Console.WriteLine($"- {prop.Name} ({prop.PropertyType.Name})");

            Console.WriteLine("\nМетоды класса:");
            foreach (var method in t.GetMethods(BindingFlags.Public |
BindingFlags.NonPublic | BindingFlags.Instance | BindingFlags.DeclaredOnly))
                Console.WriteLine($"- {method.Name} (IsPrivate:
{method.IsPrivate})");

            Console.WriteLine("\nДинамический вызов:");
            // Создаем объект на лету
            object? user = Activator.CreateInstance(t, new object[] { "Alex", 25
});

            // Вызываем публичный метод
            MethodInfo? sayHello = t.GetMethod("SayHello");
            sayHello?.Invoke(user, null);
```

```
            // Взламываем приватный метод
            MethodInfo? secret = t.GetMethod("RevealSecret",
BindingFlags.NonPublic | BindingFlags.Instance);
            secret?.Invoke(user, null);
        }
    }
}
```

Скриншоты:



```
root@ILGERPC:/home/PCPL/Lab5/Lab5Console# cd /home/PCPL/Lab6
root@ILGERPC:/home/PCPL/Lab6# cd ./DelegatesPart/
root@ILGERPC:/home/PCPL/Lab6/DelegatesPart# dotnet run
--- Делегаты ---
Результат операции: 30
Результат операции: 0

--- Обобщенные делегаты (Func) ---
Результат Func: 42
[LOG]: Тест Action завершен.
root@ILGERPC:/home/PCPL/Lab6/DelegatesPart# cd ../ReflectionPart/
root@ILGERPC:/home/PCPL/Lab6/ReflectionPart# dotnet run
--- Рефлексия ---
Свойства класса:
- Name (String)
- Age (Int32)

Методы класса:
- get_Name (IsPrivate: False)
- set_Name (IsPrivate: False)
- get_Age (IsPrivate: False)
- set_Age (IsPrivate: False)
- SayHello (IsPrivate: False)
- RevealSecret (IsPrivate: True)

Динамический вызов:
Привет, я Alex!
Секрет: 12345
root@ILGERPC:/home/PCPL/Lab6/ReflectionPart#
```