Фурасов Владислав, ИУ5-34Б

Поясните объявление делегата: Func<int, int, int, double>; Приведите примеры лямбда-выражений, которые соответствует делегату.

В языке C# делегаты могут быть обобщенными, и Func – имя обобщенного делегата. Делегат Func возвращает результат действия и может принимать параметры. Он также имеет различные формы: от Func<out T>(), где T - тип возвращаемого значения, до Func<in T1, in T2,...in T16, out TResult>(), то есть может принимать до 16 параметров.

В данном случае:

TResult Func<in T1, in T2, in T3, out TResult>(T1 arg1, T2 arg2, T4 arg4)

Последний параметр, в данном случае, double, задает тип возвращаемого значения, предыдущие – задают входные типы.

Преимуществом Func является то, что его, в отличие от обычного делегата, не нужно объявлять в тексте программы, он уже объявлен в библиотечных классах языка C#.

Лямбда-выражение (так же называемое “одноразовая функция”) обладает свойством сигнатурности, но у него нет свойства именованности. То есть у лямбда-выражения есть набор входных параметров и тип возвращаемого значения, но нет имени. Оно записывается в том месте, где должно вызываться.

Создание экземпляра делегата на основе лямбда-выражения:

```
static void PlusOrMinusMethodFunc(
        string str,
        int i1,
        int i2,
        int i3
        Func<int, int, int, double> PlusOrMinusParam)
                {
                int Result = PlusOrMinusParam(i1, i2, i3);
                Console.WriteLine(str + Result.ToString());
                }
PlusOrMinusMethodFunc(
        i1,
        i2,
        i3
        (x, y, z) => (x + y)/z
```

```
);
```

Пример объявления лямбда-выражения:

```
PlusOrMinus pm4 = (int x, int y) =>
{
int z = x + y;
return z;

};
```

Пример передачи лямбда-выражения в качестве параметра:

```
PlusOrMinusMethod(
        i1,
        i2,
        (int x, int y) =>
                {
                int z = x + y;
                return z;
                }
);
```

В примере лямбда-выражение передается в качестве параметра делегатного типа PlusOrMinus. Стоит заметить, что соответствие сигнатуры делегата и сигнатуры лямбда-выражения проверяется путем неявной типизации.
Еще одной интересная особенностью лямбда-выражений является возможность использования в них внешних переменных (замыкание). Суть замыкания состоит в том, что в каком-то блоке кода (чаще всего это обычная функция или лямбда-выражение) используются ссылки на переменные, которые объявлены снаружи этого блока кода, при этом такие внешние переменные не передаются явным образом в блок кода как параметры, другими словами, блок кода непосредственно “видит” переменные из внешнего контекста.