Перегрузка метода

```
int Sum(int a, int b)
{
    return a + b; //Сумма двух целых чисел
}

double Sum(double a, double b)
{
    return a + b; //Сумма двух чисел с плавающей запятой (double)
}

float Sum(float a, float b, float c)
{
    return a + b + c; //Сумма трёх чисел с плавающей запятой (float)
}

int Sum(int[] array)
{
    int sum = 0;

    foreach (int num in array)
    {
        sum += num;
    }

    return sum; //Сумма всех целых чисел в массиве
}
```

Перегрузка конструкторов

```
class Item
{
    public string name;
    public int price;
    public int lvl;

    public Item()
    {
        this.name = "Item";
        this.price = 15;
        this.lvl = 1;
    }

    public Item(string name)
    {
        this.name = name;
```

```
        this.price = 15;
        this.lvl = 1;
    }

    public Item(string name, int price, int lvl)
    {
        this.name = name;
        this.price = price;
        this.lvl = lvl;
    }
}
```

Перегрузка операторов

Например, мы хотим улучшать предметы в играх. Во многих MMO1 популярна механика, когда один предмет улучшается за счёт другого. Мы можем сделать это с помощью перегрузки оператора сложения:

```
public static Item operator +=(Item i1)
{
    return new Item(i1.name + "+", i1.price + price / 2, lvl+i1.lvl);
}
```

Теперь при сложении двух объектов класса Item мы будем получать третий объект с улучшенными параметрами. Вот пример использования такого оператора:

```
Item i1 = new Item("Sword", 15, 1);
Item i2 = new Item();
Item i3 = new Item();

Console.WriteLine($"{i1.name} | Price: {i1.price} | Level: {i1.lvl}"); //Выводим параметры первого предмета

i1 += i2; //Улучшаем предмет с помощью другого предмета

Console.WriteLine($"{i1.name} | Price: {i1.price} | Level: {i1.lvl}"); //Снова выводим данные

i1 += i3; //Повторяем улучшение

Console.WriteLine($"{i1.name} | Price: {i1.price} | Level: {i1.lvl}"); //Выводим новые характеристики
```