

Цель: - формирование навыков создания класса, объекта, конструкторов, методов в C#

Пример 1. Создание класса и объекта. Создание и вызов метода. Конструкторы. con101

- В едином пространстве имен **con101** с шаблоном класса **Program** создадим класс **Build**.
Наберем текст программы без выделенных серым фрагментов,
т.е. будем создавать объекты и инициализировать с конструктором по умолчанию.

```
class Build // по умолчанию модификатор доступа internal
{
    // объявление полей, модификатор доступа public
    public string name;    public int floors;           // название строения; количество этажей
    public double area;    public int kvo;             // общая площадь; количество жильцов

    public Build() {} // конструктор без параметров
    public Build(string nm, int flo, double ar, int k) // с параметрами
    {
        this.name = nm; this.floors = flo; this.ar = area; this.kvo = k;
    }

    public void ShowInfo() // метод вычисляет площадь на одного человека и выводит информацию
    {
        Console.WriteLine("\n Строение {0} имеет {1} этажа, жильцов {2},\n общая
            площадь {3}, на человека {4:f2}", name, floors, kvo, area, area/kvo);
    }
}

class Program
{
    static void Main()
    {
        Build dom1 = new Build(); dom1.ShowInfo(); // создание объекта dom1, вызов метода
        Build dom2 = new Build(); // создание объекта dom2
        dom2.name="Дом2"; dom2.area=90; dom2.kvo=7; dom2.ShowInfo(); // инициализация
        Build dom3 = new Build("Дом3", 3, 240, 27); dom3.ShowInfo(); // создание dom3

        Console.ReadKey();
    }
}
```

- Протестируем результат (F5), изменяя параметры инициализации.
- Модифицируем программу, добавив фрагменты, выделенные серым: в класс **Build** конструкторы, в класс **Program** создание объекта **dom3** с инициализацией. Протестируем результат

Пример 2. Перегрузка метода (overloading) con102

- В классе **Program** создадим три статических метода с одинаковым именем, но разными параметрами

```
class Program
{
    static void Perim(int a, int b) // два параметра - прямоугольник
    { Console.WriteLine("Периметр прямоугольника = {0}", 2*a+2*b); }
    static void Perim(int a, int b, int d) // три параметра - треугольник
    { Console.WriteLine("Периметр треугольника = {0}", a+b+d); }
    static void Perim(params int[] ar) // любое к-во параметров - n-угольник
    {
        int p = 0;
        foreach (int x in ar) p += x;
        Console.WriteLine("Периметр {0}-угольника = {1}", ar.Length, p);
    }

    static void Main()
    {
        Perim(10, 20); Perim(3, 4, 5); Perim(2, 3, 4, 5, 6, 7, 9); // 60; 12; 7-угольник 36
        Console.ReadKey();
    }
}
```

- Протестируем результат, изменяя параметры.

✳ Задания для самостоятельной работы sam10*

- Модифицируйте программу примера 1 (**con101**), добавив создание двух объектов типа **Build** с разными параметрами и способами инициализации.
- Модифицируйте программу примера 2 (**con102**), добавив перегрузку метода **Perim** для определения периметра квадрата по его стороне: **4*a**
Создайте приложения **sam10***, в которых определяются классы, поля, конструкторы, методы, создаются и инициализируются 2-3 объекта.
- Класс **Transport**. Метод **ShowInfo** выводит параметры транспортного средства: тип (автомобиль, мотоцикл, велосипед...), скорость, цвет, масса...
- Класс **Figura**. Метод **Area** выводит название (квадрат, прямоугольник, трапеция) и параметры фигуры (стороны, высоту), вычисляет и выводит площадь.
- Класс **Tovar**. Метод **ShowTovar** выводит название, цену, цвет, вес, наличие на складе, количество
- Класс **Animal**. Метод **Golos** выводит вид (кошка, собака, попугай...), имя, голос (мяу, гав, ppp...)