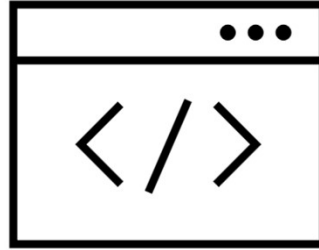


ООП

Инкапсуляция

Наследование

Полиморфизм

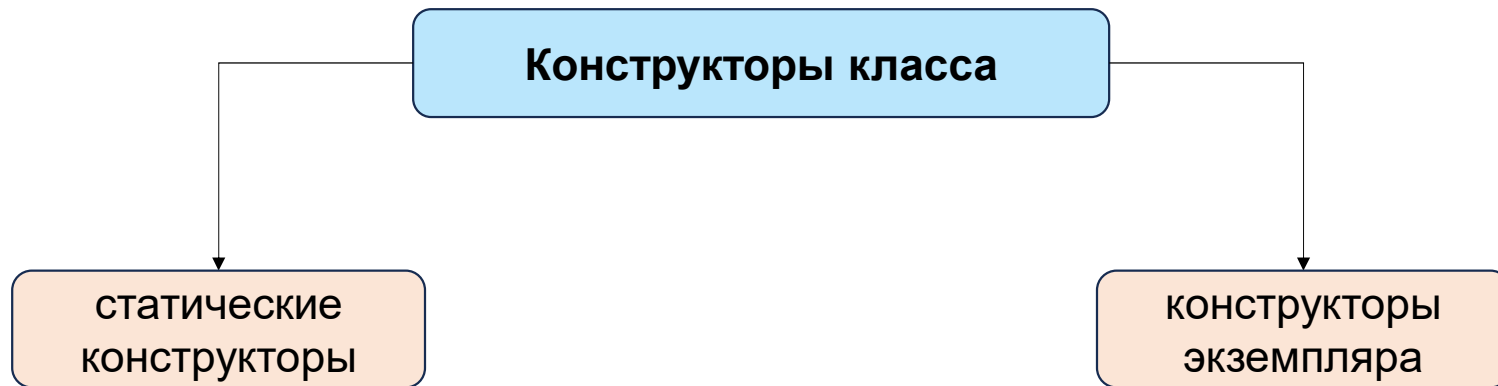


Классы. Конструкторы



КубГТУ, Кафедра «Информационных систем и программирования»

Конструктор инициализирует объект при его создании. У конструктора такое же имя, как и у его класса, а с точки зрения синтаксиса он подобен методу.



Конструктор экземпляра вызывается автоматически при создании объекта класса с помощью операции new.

До сих пор задавали начальные значения полей класса при описании класса. Это удобно в том случае, когда для всех экземпляров класса начальные значения полей одинаковы. Если же при создании объектов требуется присваивать полю разные значения, это следует делать с помощью явного задания конструктора.



Рассмотрим основные свойства конструкторов:

1. Конструктор не возвращает значение, даже типа void.

2. Класс может иметь несколько конструкторов с разными параметрами для разных видов инициализации, выбор конструктора происходит автоматически, в зависимости от количества и типа передаваемых параметров.

3. Если программист не указал ни одного конструктора или какие-то поля не были инициализированы, то полям значимых типов присваивается ноль, полям ссылочных типов – значение null.



Добавим в класс Circle два конструктора: первый – инициализирует только поле radius, второй – поля x, y, radius.

```
namespace MyProgram
{
    class Circle
    {
        public int x;
        public int y;
        public int radius;
        public static readonly string name =
"Окружность";
        //конструктор 1 – инициализирует
только поле radius
        public Circle(int radius)
        {
            this.radius= radius;
        }
    }
}
```

```
// Конструктор 2 - инициализирует
поля x, y, radius
    public Circle (int x, int y, int radius)
    {
        this.x = x;
        this.y = y; this.
        radius= radius;
    }

    public void Set(int x, int y, int radius)
    {
        this.x = x;
        this.y = y;
        this.radius= radius;
    }
}
```

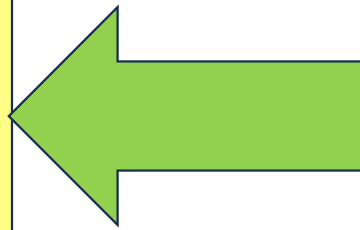


```
public void Show()
{
    Console.WriteLine("{0} с центром в точке ({1},{2}), радиусом {3}", name, x, y, radius);
}
```

Если один из конструкторов выполняет какие-либо действия, а другой должен делать то же самое плюс еще что-то, то можно воспользоваться параметром `this` для того, чтобы вызвать один конструктор из другого. **Например:**

```
public Circle( int radius) //конструктор 1
{
    this.radius= radius;
}
public Circle (int x, int y, int radius) :this
(radius) //конструктор 2
{
    this.x = x;
    this.y = y;
}
```

В данном случае конструктор 2 вызывает конструктор 1. Запись вида **:this(radius)** называется инициализатором, то есть кодом, который будет выполнен до начала выполнения тела конструктора 2.



Рассмотрим, как вызвать конструкторы класса Circle из класса Program:

```
using System;
namespace MyProgram
{
    class Program
    {
        static void Main()
        {
            //вызов конструктора 1
            Circle oneCircle = new Circle(1);
            oneCircle.Show();
            //вызов конструктора 2
            Circle twoCircle=new Circle(1, 1, 100);
            twoCircle.Show();
        }
    }
}
```

Результат работы программы:

Окружность с центром в точке (0, 0)
радиусом 1

Окружность с центром в точке (1, 1)
радиусом 100



Существуют следующие типы конструкторов:

- конструктор без параметров;
- параметризованный конструктор;
- конструктор по умолчанию.

Конструктор по умолчанию создается в языке C#, если в классе не реализованы конструкторы. По определению такой конструктор никогда не принимает аргументов.



Конструктор также может принимать один или несколько параметров. В конструктор параметры вводятся таким же образом, как и в метод. Для этого достаточно объявить их в скобках после имени конструктора.

Конструкторы, как и другие методы можно перегружать, используя разные параметры.



КубГТУ, Кафедра «Информационных систем и программирования»



Статические конструкторы обладают следующими свойствами:

- Статический конструктор не принимает модификаторы доступа и не имеет параметров.
- Класс или структура могут иметь только один статический конструктор.
- Статические конструкторы не могут быть унаследованы или перегружены.
- Статический конструктор нельзя вызывать напрямую. Он запускается автоматически.

Статический конструктор используется для инициализации любых статических данных или для выполнения определенного действия, которое требуется выполнить только один раз. Статический конструктор будет вызываться по крайней мере один раз.



КубГТУ, Кафедра «Информационных систем и программирования»