Constructores

Constructores

```
class Animal{

    public Animal(){

    }
}
```

Mismo nombre de la clase

SIN tipo de retorno

```
new Animal();
```

Se invoca cuando se instancia el objeto con el operador ***new***

Constructores

```
class Animal{
    public Animal(){}
}

new Animal();
```

Cuando no se agrega un constructor,
el lenguaje AGREGA un ***CONSTRUCTOR POR DEFECTO***

Constructores

```
class Persona{
   String nombre;
   int edad;
   public Persona(String nombre){ }
   public Persona(String nombre, int edad){ }
}

new Persona(“Juan”);
new Persona(“Juan”,20);
```

Se pueden SOBRECARGAR LOS CONSTRUCTORES

Constructores

```
class Persona{
   String nombre;
   int edad;
   public Persona(String nombre){ }
   public Persona(String nombre, int edad){ }
}

new Persona("Juan");
new Persona("Juan",20);
```

*Cuando se agrega un constructor, el lenguaje **NO AGREGA** el CONSTRUCTOR POR DEFECTO*

~~`new Persona();`~~ ERROR DE COMPILACIÓN!!

Constructores

```
class Persona{
    String nombre;
    int edad;
    public Persona(String nombre){
        this.nombre=nombre
    }
    public Persona(String nombre, int edad){
        this(nombre);
        this.edad = edad;
    }
}
```

*Con **this( )** se puede invocar a un constructor sobrecargado EN LA MISMA CLASE*

Constructores

```
class Figura{
    public Figura(int tipo){}
}

class Triangulo extends Figura{
    public Triangulo(){
        super(5);
    }
}
```

*Con **super()** se puede invocar a un constructor*
DE LA CLASE PADRE

Constructores

```
class Animal{
    public Animal(){
        super();
    }
    public Animal(String tipo){
        super();
    }
}
```

*Si un constructor no tiene una llamada explícita a this() o super(), el lenguaje coloca una llamada a **super() sin parámetros***

Constructores

```
class Animal{
    String tipo;
    public Animal(String tipo){
        this.tipo=tipo;
        super();
    }
}
```

Error de compilación

*Si existe una llamada a **this()** o **super()*** debe ser la primera línea en el constructor.

Constructores

```
class Animal{
    public Animal(){
        super();
    }
    public Animal(String tipo){
        super(); this();
    }
}
```

Error de compilación

*Si existe una llamada a **this()** o **super()*** debe ser la primera línea en el constructor.

No puede existir en un mismo constructor ambas llamadas

Constructores

```
class Animal{
    public Animal(){

    }
    public Animal(String tipo){
        Animal();
    }
}
```

Error de compilación

Un constructor se invoca:
Con this()
Con super()
Instanciado el objeto con new, por ejemplo new Animal().

NUNCA como un método normal

EJERCICIO

```
class X{
    public X(){
        System.out.print("X");
    }
}
class Y entends X{
    public Y(){
        System.out.print("Y");
    }
}

new Y();
```

Imprime en consola XY

Constructores

EJERCICIO

```
class X{
    public X(int value){
        System.out.print("X");
    }
}
class Y entends X{
    public Y(){
        System.out.print("Y");
    }
}

new Y();
```

Error de compilación

EJERCICIO

```
class X{
    public X(int value){
    System.out.print("X");
    }
}
class Y entends X{
    public Y(){
            super();
        System.out.print("Y");
    }
}

new Y();
```

Error de compilación