# Clases en Java

Constructores

# Creación de objetos

[tipo] nombreVariable = new tipo([parámetro1[, parámetro2[, ...]]]);

- Crea un objeto con el nombre nombreVariable
- Le asigna memoria dinámicamente
- Inicializa sus variables de instancia a los valores por defecto: null para los objetos.
- Llama al constructor con los parámetros especificados
- Por último devuelve una referencia al objeto creado, es decir la dirección de memoria donde se encuentra dicho objeto.

Ejemplos:

```
Repuesto unaPieza = new Repuesto();

Automovil miCarro = new Automovil(5, "Golf");

Scanner teclado = new Scanner(System.in);
```

# Definición de Métodos

```
[modificadoresDeMétodos] tipoDeResultado nombreMétodo
 ([tipoParámetro1 parámetro1
 [,tipoParámetro2 parámetro2[, ...]]])
 {
 //cuerpo del método
}
```

# Ejemplo

```
public class Ejemplo1 {
 private int at1;
 private void calcularImpuestos(){
 // cuerpo del método
 }
public int calcularTotal (double x){
 // cuerpo del método
 }
/* Un método que devuelve un objeto de tipo MiObjeto y recibe un
entero y  una cadena de entrada */
 protected MiObjeto convertir(int z, String s) {
 // cuerpo del método
 }
} // clase Ejemplo1
```

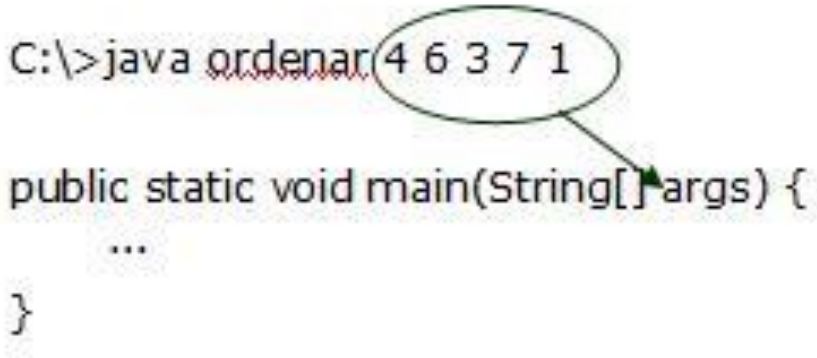
# Llamado a métodos

```
//método set
public void setAt1(int x)
    {at1=x; return;}
//método get
public int getAt1()
   {return at1;}
// interior de un método main
{
Ejemplo1 unaPieza = new Ejemplo1();
 unaPieza.calcularTotal(16.5);
  …
}
```

# Método main

public static void main(String args[] ) Este método

- Se llama antes de la creación de un objeto.
- Ha de declararse como static o de clase para que se pueda llamar sin tener que referirse a una instancia particular de la clase.
- La palabra reservada void indica que main no devuelve nada




| | |
|---|---|
| args[0] | "4" |
| args[1] | "6" |
| args[2] | "3" |
| args[3] | "7" |
| args[4] | "1" |

# Constructores

- Un **constructor** es un método especial de una clase que se llama automáticamente siempre que se declara un objeto de esa clase.
- Su función es inicializar el objeto y sirve para asegurarnos que los objetos siempre contengan valores válidos.

Características:

- Tiene el mismo nombre que la clase a la que pertenece.
- En una clase puede haber varios constructores con el mismo nombre y distinto número de argumentos (se puede sobrecargar)
- No se hereda.
- No puede devolver ningún valor (incluyendo void). No lleva void
- Debe declararse público (salvo casos excepcionales) para que pueda ser invocado desde cualquier parte donde se desee crear un objeto de su clase.

# Definición de constructor

```
[modificadoresDeConstructor]
nombreConstructor
 ([tipoParámetro1 parámetro1
 [,tipoParámetro2 parámetro2[, ...]]])
{
 //cuerpo del constructor
}
```

El encabezado, no lleva void

Un constructor debe ser invocado cuando se crea el objeto con el operador new.

# Constructor por Defecto

- Si para una clase no se define ningún método constructor se crea uno automáticamente por defecto.
- El **constructor por defecto** es un constructor sin parámetros que no hace nada.
- Los atributos del objeto son iniciados con los valores predeterminados por el sistema.

# Leer caracteres

```
import java.util.*;

public class Leecarac {

 public static void main(String args[]){
        String  nombre;
        char car;
        Scanner entrada = new Scanner(System.in);
        System.out.println("da un caracter");
        car=entrada.next().charAt(0);     // es un cero el argumento
        System.out.print("el caracter es " + car);
    }
}
```

La clase Scanner NO CONTIENE un método nextChar() para leer un dato de tipo char desde teclado
charAt(0) extrae el primer caracter

# Ejemplo: Con parámetros

```
class Constructor
{
 protected String nombre ;
 protected int edad ;
 public Constructor(String nom, int can)
            {
              nombre = nom;
              edad = can;
            }
public static void main(String args[])
      {
Constructor p = new Constructor("Juán Pérez", 23);
        System.out.println("Nombre: "+ p.nombre + " " + "Edad: " +
            p.edad);
}
}
```

# Ejemplo: por defecto y con parámetros

```
public class Fecha {

   private int da;
   private int ms;
   private int ao;

   public Fecha() { }

   public Fecha(int d, int m, int a) {
            da=d;
            ms=m;
            ao=a;
   }
      public static void main(String[] args) {
            Fecha f1 = new Fecha();  // por default
            Fecha f2 = new Fecha(1,5,16);  // con parámetros
            System.out.println("el día es "+f1.da+" el mes es "+f1.ms+" el año
es " +f1.ao);
            System.out.println("el día es "+f2.da+" el mes es "+f2.ms+" el año
es " +f2.ao);

   }
}
```

# Ejemplo

```
import java.util.*;


public class Taxi{
        private String ciudad;
        private String matricula;
        private String distrito;
        private int tipoMotor; /*0=desconocido, 1=gasolina,
2=diésel */
```

# Su Constructor

```
// constructor
public Taxi(String valorCiudad, String valorMatricula, String valorDistrito, int valorTipoMotor){
        ciudad=valorCiudad;
        matricula=valorMatricula;
        distrito=valorDistrito;
        tipoMotor=valorTipoMotor;
}
```

# Métodos set

```
// método para asignar ciudad
public void setCiudad(String c){
        ciudad=c;
        return;
}
// método para asignar matrícula
public void setMatricula(String m){
        matricula=m;
        return;
}
// método para asignar distrito
public void setDistrito(String d){
        distrito=d;
        return;
}
// método para asignar el tipo de motor
public void setTipoMotor(int tm){
        tipoMotor=tm;
        return;
}
```

# Métodos get

```
// método para obtener ciudad
public String getCiudad(){
        return ciudad;
}
// método para obtener matrícula
public String getMatricula(){
        return matricula;
}
// método para obtener distrito
public String getDistrito(){
        return distrito;
}
// método para obtener tipo de motor
public int gettipoMotor(){
        return tipoMotor;
}
```

# Método main

```
public static void main(String[] ar){
    Taxi t1 = new
Taxi("Puebla","A","Puebla", 1);
    Taxi t2 = new
Taxi("Tlaxcala","A","Puebla", 2);
     Scanner entrada=new
Scanner(System.in);
     System.out.println("Ciudad:");
     String cd=entrada.nextLine();
     System.out.println("Matricula:");
     String ma=entrada.nextLine();
System.out.println("Distrito");
     String dto=entrada.nextLine();
     System.out.println("Tipo
motor:");
     int tr=entrada.nextInt();
     t1.setCiudad(cd);

System.out.println(t1.getCiudad())
;
     t1.setCiudad("Atlixco");
     System.out.println("Ciudad:");

System.out.println(t1.getCiudad())
;
   }
}
```

# Práctica 6

1. Define una clase Bombero considerando los siguiente:

- **atributos de clase**: nombre (String), apellidos (String), edad (int), casado (boolean), especialista (boolean).
- Define **un constructor** que reciba los parámetros necesarios para la inicialización
- Los métodos para poder establecer y obtener los valores de los atributos.

2. Cree la clase Docente que tenga

- como atributos privados: nombre, apellido, tipo (ciencias y letras) y horas
- Un constructor que inicialice a todos los atributos.
- Métodos de acceso: *set/get* para los atributos.
- Un método que retorne el nombre completo (nombre y apellido)
- Un método que retorne el sueldo Bruto ("ciencia"=horas*3 , "letras"= horas*5 )
- Un método que retorne el descuento del 0.10
- Un método que retorne el sueldo neto

# Tarea 7

1. Haz una clase llamada **Persona** bajo las siguientes condiciones:

- Sus atributos son: **nombre, edad, DNI, sexo** (H hombre, M mujer)**, peso y altura.** No queremos que se accedan directamente a ellos. Piensa que modificador de acceso es el más adecuado, también su tipo. Si quieres añadir algún atributo puedes hacerlo.
- Por defecto, todos los atributos menos el DNI serán valores por defecto según su tipo (o números, cadena vacía para String, etc.). Sexo sera hombre por defecto, usa una constante para ello.
- Se implantaran varios constructores:
  - Un constructor por defecto.
  - Un constructor con el nombre, edad y sexo, el resto por defecto.
  - Un constructor con todos los atributos como parámetro.

# Tarea 7

- Los métodos que se implementaran son:
  - **calcularIMC()**: calculara si la persona esta en su peso ideal (peso en kg/(altura^2  en m)), devuelve un -1 si esta por debajo de su peso ideal, un 0 si esta en su peso ideal y un 1 si tiene sobrepeso .Te recomiendo que uses constantes para devolver estos valores.
  - **esMayorDeEdad()**: indica si es mayor de edad, devuelve un booleano.
  - **comprobarSexo(char sexo)**: comprueba que el sexo introducido es correcto. Si no es correcto, sera H. No sera visible al exterior.

# Tarea 7

- Ahora, crea una clase ejecutable que haga lo siguiente:
  - Pide por teclado el nombre, la edad, sexo, peso y altura.
  - Crea 3 objetos de la clase anterior, el primer objeto obtendrá las anteriores variables pedidas por teclado, el segundo objeto obtendrá todos los anteriores menos el peso y la altura y el último por defecto, para este último utiliza los métodos set para darle a los atributos un valor.
  - Para cada objeto, deberá comprobar si está en su peso ideal, tiene sobrepeso o por debajo de su peso ideal con un mensaje.
  - Indicar para cada objeto si es mayor de edad.
  - Por último, mostrar la información de cada objeto.