Clase 2 - Parámetros, clases

martes, 21 de marzo de 2023 16:54

```
package whatsapp;
import java.awt.Image;

public class Contacto {           Estado
  private String nombre;
  private Image imagen;
  private String estado;
  private int id;
                                  Comportamiento
  public String getNombre() {
    return nombre;
  }
  public void setNombre(String nombre){
    this.nombre = nombre;
  }
  . . .
}
```

Cada vez q' hacemos new se crea una nueva instancia con su propio

Estado → Lo q' cada obj. sabe de sí mismo

Comportamiento → Lo q' cada ob puede hacer

métodos → Nombre
- → lista de arg. (opcional)
- → tipo de retorno
- → modificador de acceso (opcional) → public / private } encapsulamiento

Hacer métodos públicos

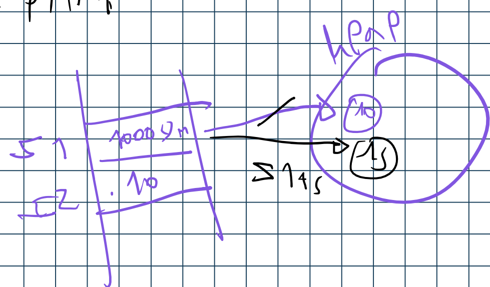
Hacer var privadas { con privado si rompe menos q' solo se modifican desde el método

Datos primitivos

Incorporados por eficiencia

Cada prim tiene un wrapper q' almacena solo 1 prim

```
Integer I1 = new Integer(10);   referencias
int     i2 = 10;
```



wrappers son inmutables → I1+5 crea una nueva cosa en heap y cambia pointer → ineficiente.

↳ tienen + cosas.

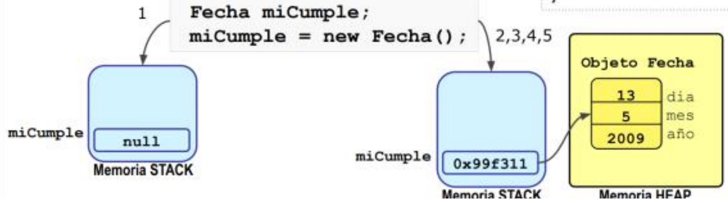
Entero: **byte, short, int, long**
Punto flotante: **float y double**
Un carácter de texto: **char**
Lógico: **boolean**
} primitivos

```
Integer nro = 3;          → boxing
Character letra = 'a';
int num = nro;            → unboxing.
```

1. Se aloca espacio para la variable (Fecha micumple;)
2. Se aloca espacio para el objeto en la HEAP y se inicializan los atributos con valores por defecto.
3. Se inicializan explícitamente los atributos del objeto.
4. Se ejecuta el **constructor** (*parecido* a un método que tienen el mismo nombre de la clase)
5. Se asigna la referencia del nuevo objeto a la variable. } new

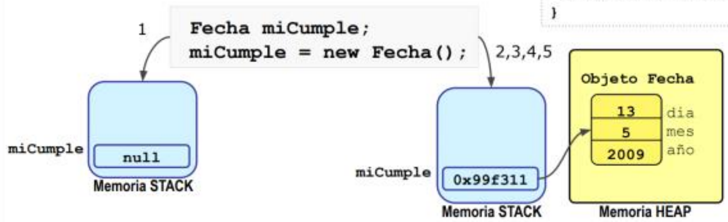
```
public class Fecha {
  private int dia = 13;
  private int mes = 5;
  private int año = 2009;
  // métodos de instancia
}
```

```
Fecha miCumple;
miCumple = new Fecha();
```



1. Se aloca espacio para la variable Fecha miCumple;
2. Se aloca espacio para el objeto en la HEAP y se inicializan los atributos con valores por defecto.
3. Se inicializan explícitamente los atributos del objeto.
4. Se ejecuta el **constructor** (*parecido* a un método que tienen el mismo nombre de la clase)
5. Se asigna la referencia del nuevo objeto a la variable. new

```
public class Fecha {
  private int dia = 13;
  private int mes = 5;
  private int año = 2009;
  // métodos de instancia
}
```



constructores

Definen estado inicial de un objeto.

Si no declaro ninguno, crea uno x defecto.

Puede haber sobrecarga → tener varios constructores con ≠ tipos y cantidad de objetos

```
public class Vehiculo {
  private String marca;
  private double precio;

  public Vehiculo(String mar,double pre){
    marca = mar;
    precio = pre;
  }
}
```

```
public class Vehiculo {
  private String marca;
  private double precio;

  public Vehiculo(String marca,double precio){
    this.marca = marca;
    this.precio = precio;
  }
}
```

Codificaciones equivalentes

Si le sacas el this, le asignas a la var marca el valor marca.
NO modifico el objeto.

this refiere al objeto receptor del msj.

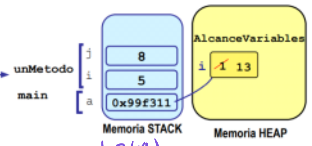
Var de instancia vs de instancia

Locales → Definidas en cada método
→ Pasadas como parámetro
→ Viven hasta q' se ejecute el método

De instancia → Creadas cuando se construye el objeto
→ Vive mientras viva el objeto

```
public class AlcanceVariables {
  private int i=1;
  public void unMetodo(int i){
    int j=8;
    this.i=i+j;
  }
}
```

lo pone en el objeto de instancia.



local

param. tapa var de instancia

static → var y method de clase

- se pueden invocar sin haber creado el objeto.

- Asociado a la clase, no a la instancia

math.set(90) ←→ m1.set(90)
(método) (var)

- Lo estático se carga al cargar la clase
- var de clase las comparten todas las instancias de la clase
- Métodos de clase se usan cuando comportamientos no dependen de la instancia particular

```java
public class Contacto {
  private static int ultCont = 0;
  private String nombre;
  private Image imagen;
  private String estado;
  private int id;

  public Contacto() {
    ultCont++;
    this.id = ultCont;
  }
  public static int getUltCont {
    return ultCont;
```

→ var de clase (ultCont)

→ método de clase (getUltCont)

No puedo acceder a var de instancia desde método estático → a qué objeto refiere? ¿Y si no hay objetos creados?

Arreglo

Cliente[] cliente = new cliente[3]

se crea vacío con espacio para clientes.

```java
int[]  intArray = {6, 3, 2, 4, 9};
Cliente[] cliArray = {new Cliente(), new Cliente(), new Cliente()};
String[] items = { "item1", "item2", "item3", "item4"};
```

} crea y llena de una.

```java
public int suma2(int[] a) {
  int result = 0;
  for (int elto: a)
     result = result + elto;
  return result;
}
```

(2) *For-each*: La línea `for(int elto:a)` se lee así: para cada **elemento elto de tipo int**, en el arreglo **a**

matrices - arreglo bidimensional

```java
public class RecorridoMatriz{
    public static void main (String args []){
     int [][] notas =
              {{66,78,78,89,88,90},
               {76,80,80,82,90,90},
               {90,92,87,83,99,94}};
       for (int x = 0; x < notas.length; x ++){
              for(int y = 0; y < notas[x].length; y++) {
                 System.out.println(notas[x][y]);
```

1º dimensión — 2º dimensión

Cant. de filas de la matriz (sería 3)

Cant. de columnas de esa fila (sería 6)

(1) tradicional

```java
for(int[] fi
  for(int elt
    System.out
}
```

notas[x].length

[] []	0	1	2	3	4	5
0	66	78	78	89	88	90
1	76	80	80	82	90	90
2	90	92	87	83	99	94

} array.length

(2) Usando for-each

```java
for(int[] fila: notas){
  for(int elto: fila)
    System.out.print(elto);
}
```

pasaje de parámetros

pasados x valor, se hace una copia.

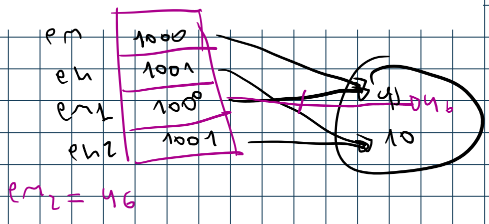
```java
package  ayed.tp02;

public class PasajePorValor {
  public static int mult(int x, int y) {
    return x * y;
  }
  public static void main(String[] args) {
    int alto = 10;
    int ancho = 5;
    int area = mult(alto, ancho);
  }
}
```

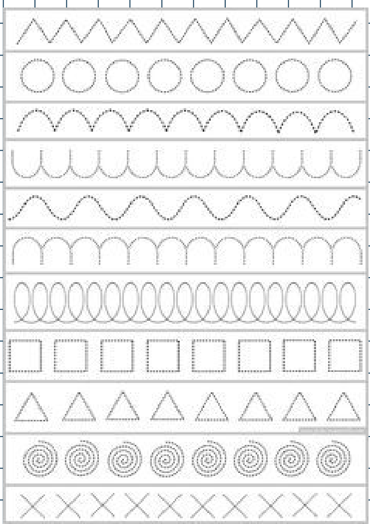
Parámetros formales
Son los parámetros en la definición del método

Parámetros actuales/reales
Son los parámetros en la invocación al método

```java
public static void main(String[] args) {
    Integer edadMadre = 0;
    Integer edadHijo = 0;
    TestPasajeParametros.pedirEdades(edadMadre, edadHijo);
    System.out.println("La madre tuvo a su hijo a los: "+(edadMadre-edadHijo)+" años");
  }
}
```

```
La madre tuvo a su hijo a la edad de : 0 años
```





Aaaaaaaaaaaaaaaaaaaaaaaaaaaa
Bbbbbbbbbbbbbbbbbbbbbbbbbb
Cccccccccccccccccccccccccccccccc
Ddddddddddddddddddddddddddd
Eeeeeeeeeeeeeeeeeeeeeeeeeeeeee
Ffffffffffffffffffffffffffffffffffffffffff
Ggggggggggggggggggggggggggggggggg
Hhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhh
Iiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiii
Jjjjjjjjjjjjjjjjjjjjjjjjjjjjjjjjjjjjjjjjjjjjjjj
Kkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkk
Llllllllllllllllllllllllllllllllllllllllllllllllllllllllllllllllll
Mmmmmmmmmmmmmmmmmmmmmmmmmm
Nnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnn
Oooooooooooooooooooooooooooooooo
Ppppppppppppppppppppppppppppppppp
Qqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqq
Rrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrr
Ssssssssssssssssssssssssssssssssssssssssss
Ttttttttttttttttttttttttttttttttttt
Uuuuuuuuuuuuuuuuuuuuuuuuuuuuuuuuu
Vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
Wwwwwwwwwwwwwwwwwwwwwwwwwwww
Xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Yyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyy
zzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzz
zz