**Programación Orientada a Objetos**

[Programación estructurada / Programación orientada a objetos]

**Ventajas:**

* **Fácil comprensión**: modelo basado en el mundo real
* **Modularidad** (mejor organización)
* **Fácil mantenimiento**
* **Seguridad** (menor probabilidad de cometer errores)
* **Reusabilidad** (de los módulos, clases)

**Características:**

* **Abstracción** (mediante clases)
* **Modularidad**
* **Encapsulamiento** (ocultamiento de la información interna)
* **Jerarquía**
  + **Herencia** (x es un y)
  + **Inclusión / Agregación**
* **Polimorfismo** (redefinir un método de una clase para adaptarlo)

**Objetos y clases**

**Características:**

* **Identidad** (diferencia objetos)
* **Estado** (parámetros sus valores que lo definen)
* **Comportamiento** (métodos del objeto)

**Partes:**

* **Campos (o variables miembro)**
* **Métodos** (única forma de manipular la información del objeto)

[Clase = abstracción de un objeto / Objeto = instancia de una clase]

**Creación y uso:**

* **Declaración** (“puntero a dirección en memoria”, sin instanciar apuntará a null)



* **Instanciación** (reserva memoria suficiente)



* **Manipulación**





**Métodos**

[Utilización]

**Parámetros:**

* **Por valor:** No altera las variables al acabar
* **Por referencia:** Pasa la dirección del dato, así que sí altera su valor al acabar

[Constructores / Constructor por defecto]

**Métodos estáticos**

* No es necesario crear un objeto de su clase para usarlo
*  (o nombre\_objeto / nada si misma clase)
* No afectan al estado de los objetos instanciados
* Se usan para operaciones comunes a todos los objetos de la clase

**Paquetes y librerías**

[(package) / sentencia import (clase de paquete distinto)]

Si un paquete contiene subpaquetes, el comodín no importará las clases de los subpaquetes, tan solo las que haya en el paquete.

**Entrada y salida de datos**



