Lenguajes de programación

Tipos:

- Naturales: idiomas
- Formales: programación (símbolos y reglas formalmente especificados)
  - Máquina: más bajo nivel (0,1)
  - Ensamblador: versión simbólica lenguajes máquina
  - Altonivel: estructuras de control, variables de tipo, recursividad,..
  - Orientados a problemas: cuarta generación. (SQL)

Función:

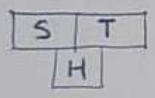
- Representan conjunto de construcciones abstractas centradas en resolver problemas en base a paradigmas mentales y de desarrollo.
- Si los problemas evolucionan, los lenguajes evolucionan y pueden aparecer nuevos lenguajes.

- Traductor:

- Lee y traduce un programa de lenguaje fuente de alto nivel a lenguaje objeto
- Funciones:
  - Intérprete: ejecuta directamente lo traducido, sin almacenarlo en disco duro
  - Compilador: genera un fichero donde almacenar lo traducido sin necesidad de volver a traducirlo.

- Intérprete: ejecuta el código según lee (siempre presente en traducción y ejecución)

- Cuando lee una línea comprueba si es correcta y si lo es, la ejecuta
- Ejecución interactiva
- Los intérpretes más puros no guardan copia del programa.



S: lenguaje fuente
H: lenguaje anfitrión
T: lenguaje objeto

- Tipos

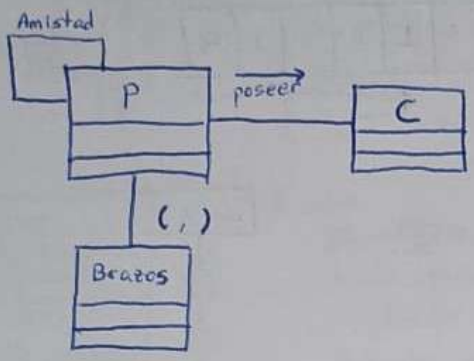
Implementación
- En metal: máquina física
- Máquina virtual

Ejecución
- Máquina de pila
- Máquina de registros (von Neumann)

Multiplicidad / Cardinalidad

- Son sist. dinámicos
- Hay máx. y mín. (representan relaciones control).

Asociación → Agregación → Composición



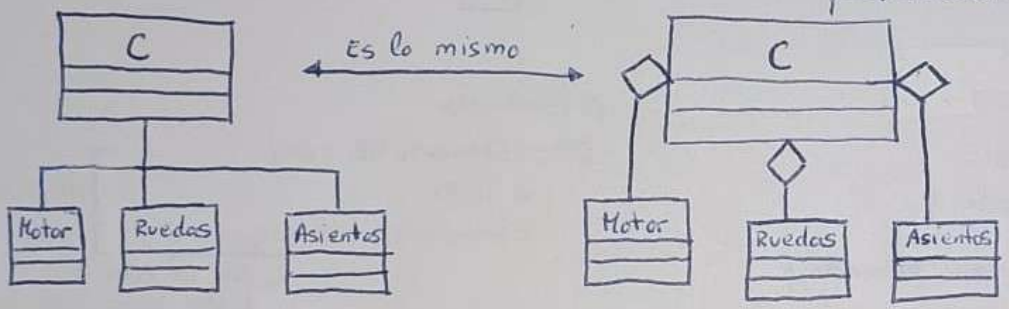
- Máx: puede haber relación entre dos estancias o más (multiplicidad)
- Mín: no hay relaciones

Ej: Mín {0, 1} Máx {1, n} ← multiplicidad

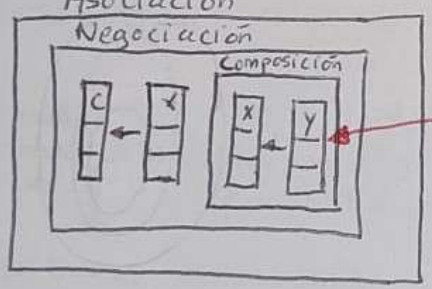
Opciones
(0,1)
(0,n)
(1,1) → obligado a tener 1
(1,n)

Class P{
posee coche (variable) (,1)
posee coche [ ] (,n)
→ Class C{
Arrays (múltiples valores}

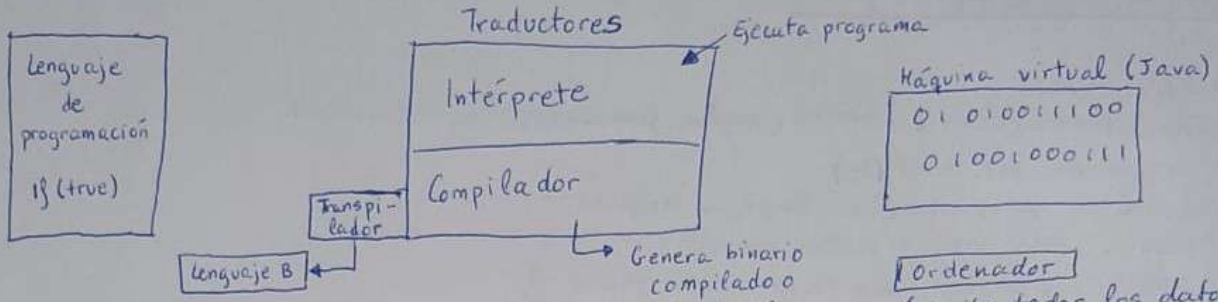
- 1:1 posee coche
- 1:N posee coche [ ]
- N:M 1:N + es poseido por [ ]



→ Exclusividad.



Sin "y" ~~no~~ "x"
Ese "y" para ese "x"

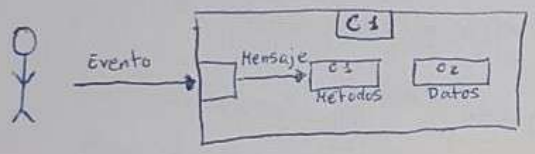


Proceso

- Abstracción
- Reutilización
- Encapsulamiento → principio ocultación
- Modularidad
- Herencia
- Polimorfismo
- Recolección basura

Clase

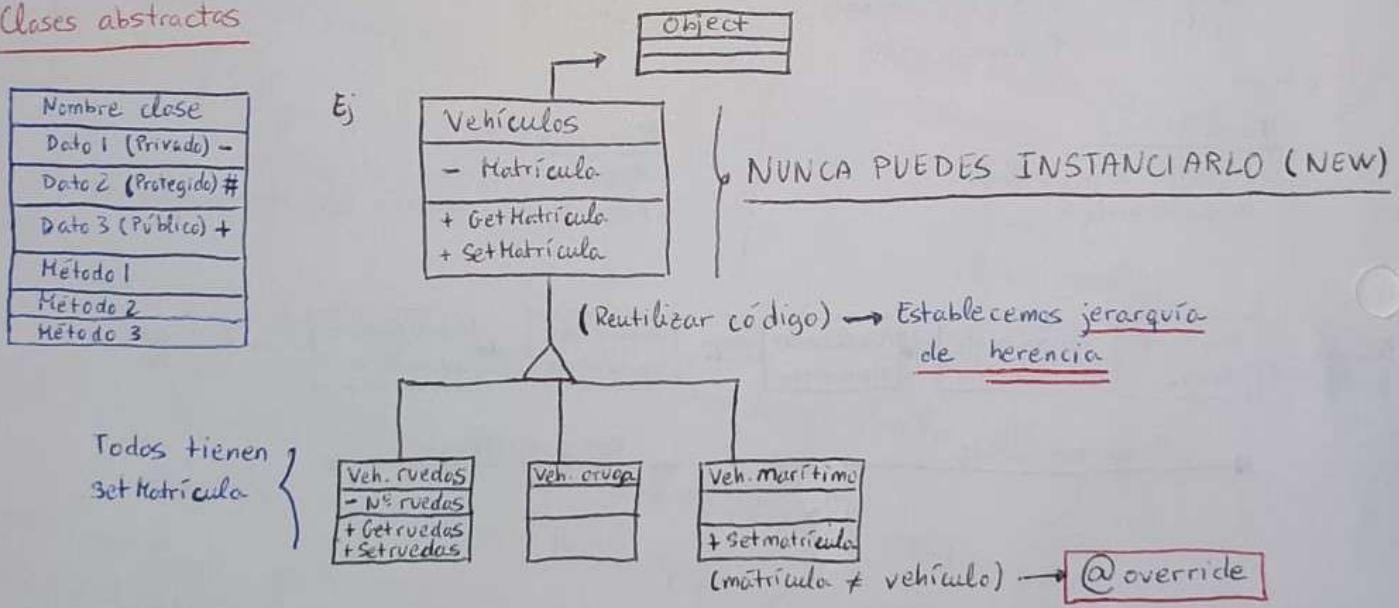
- Objeto → Implementación
- Métodos → Funcionalidad de clases
  - ↳ Invocación (mensaje que envía los métodos)
- Evento → mensaje externo (no controla receptor)
- Atributo / propiedad → dato, variable de clase, ...
- Identificador / referencia
- Estado interno

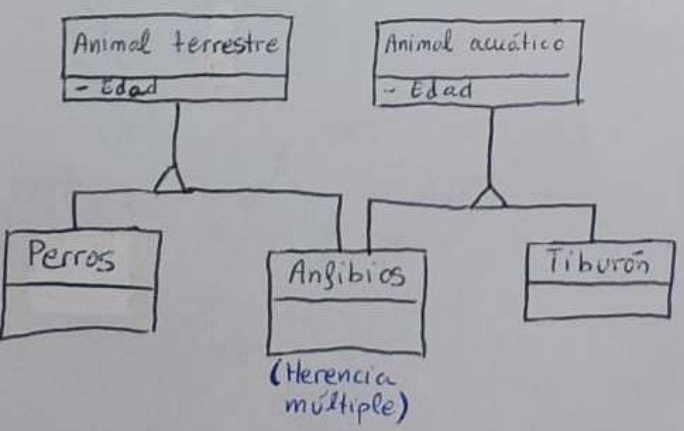


Diagramas

Juntos:
- De uso (interacciones usuario y sistema)
- Clases (representa clases, atributos, métodos y relaciones)
- Secuencia (orden e interacción de objetos)
- Estado (estados y transiciones de un objeto)
- Componentes (organización y dependencia hardware)

Clases abstractas





- La herencia múltiple en Java está prohibida
- Para ello usamos interfaces, con el fin de reutilizar código