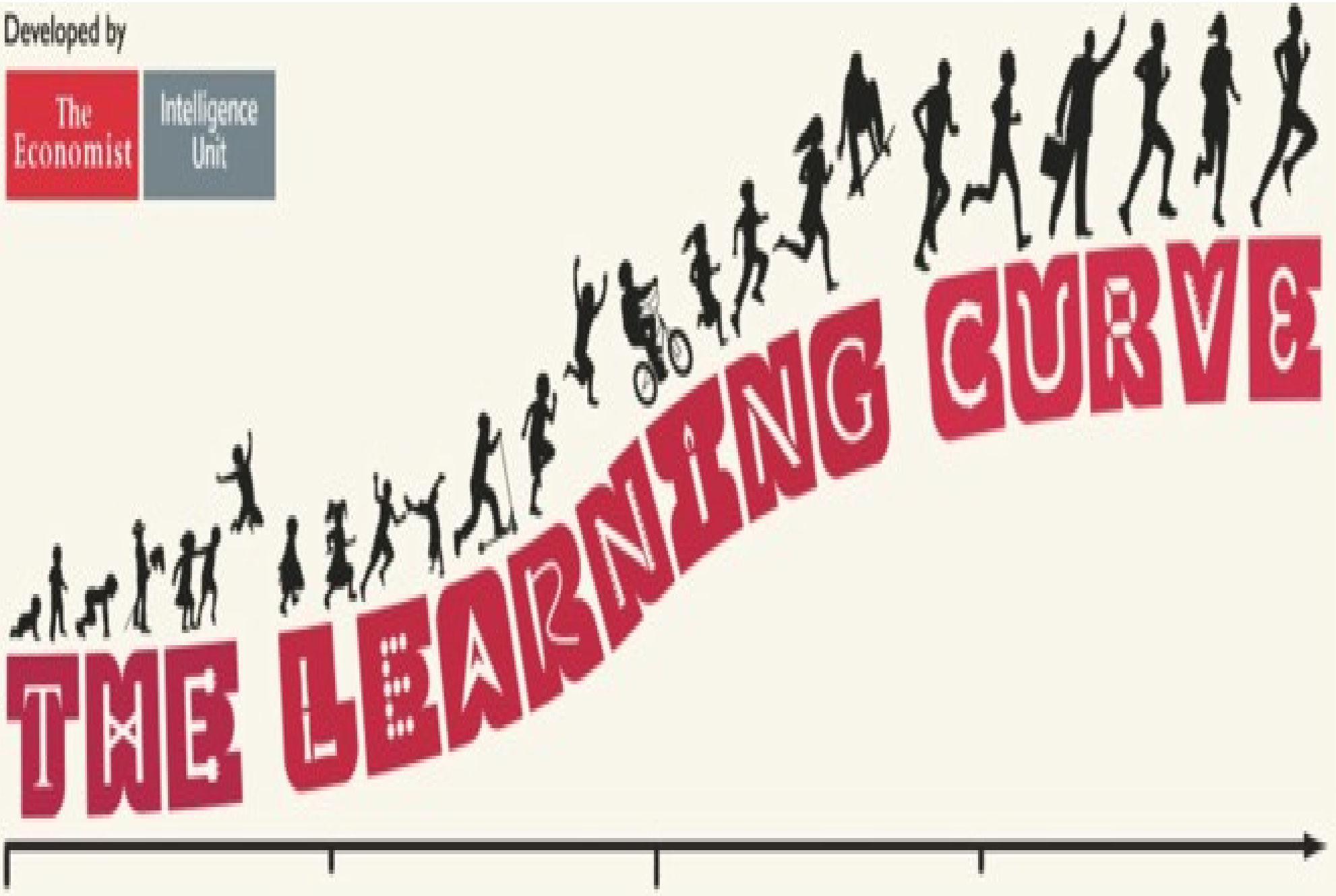
Programmazione ad oggetti

Linguaggio JAVA

La conoscenza che non *entra nella carne*
è solo rumore!

Developed by

The Economist Intelligence Unit



I principi della OOP

- **Astrazione**: definire nuovi tipi di dati.
- **Incapsulamento**: nascondere i valori e le operazioni.
- **Ereditarietà**: riusare il codice esistente.
- **Polimorfismo**: estendere le capacità degli oggetti.

ADT – Tipi di dato astratti

- Definire nuovi tipi
  - dati, (proprietà, attributi, ...)
  - operazioni (metodi)
    - *che agiscono sui dati, leggendoli o scrivendoli*

Incapsulamento

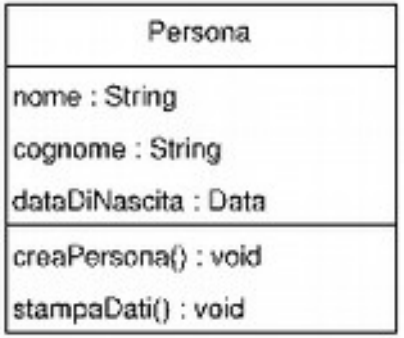
- i **dati** devono rimanere privati insieme all'implementazione
- solo l'**interfaccia** delle operazioni è resa pubblica all'esterno della classe: i **metodi** consentono di gestire le proprietà degli oggetti.

Descrizione di una classe

- i dati (o **attributi, fields**)
  - contengono le informazioni di un oggetto;
- le operazioni (o **metodi**)
  - consentono di leggere/scrivere gli attributi di un oggetto, cioè ne gestiscono il comportamento (**behavior**)

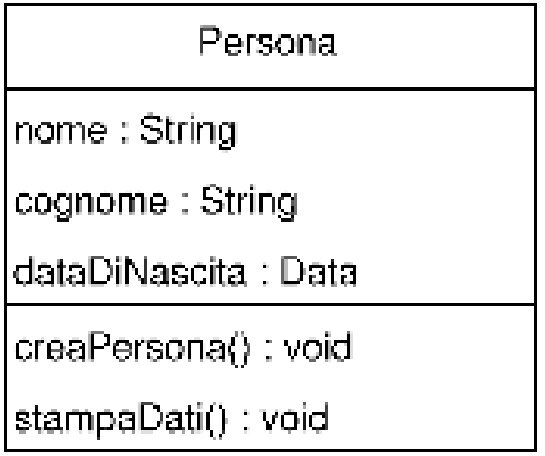
UML - Unified Modeling Language

- In UML una classe si rappresenta così:



L’oggetto

- un oggetto è una **istanza** di una classe.
- un oggetto **deve** essere conforme alla descrizione di una classe.
- un oggetto è contraddistinto da:
- 1. attributi;
- 2. metodi;
- 3. identità;



L’oggetto

un oggetto non dovrebbe mai manipolare direttamente i dati di un altro oggetto

la **classe** è una entità **statica** cioè viene caricta in memoria a *tempo di compilazione (compile-time)*

l’oggetto è una entità **dinamica** cioè a *tempo di esecuzione* (*run-time*)

Relazioni fra le classi

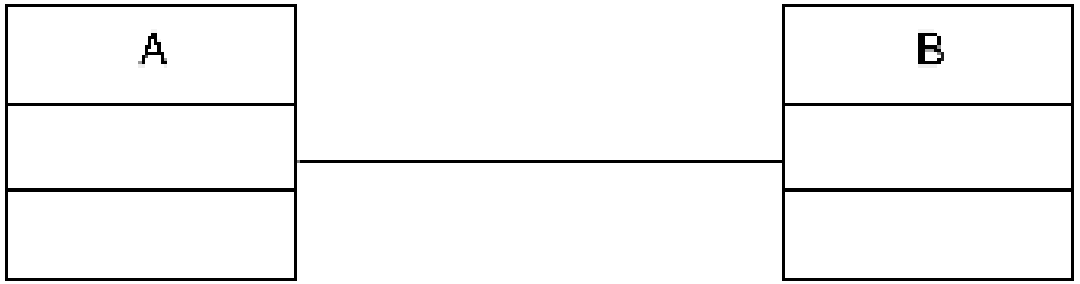
- **uso**: una classe può **usare** oggetti di un’altra classe;
- **aggregazione**: una classe può **avere** oggetti di un’altra classe;
- **composizione**: una classe può **essere composta** da oggetti di un’altra classe (aggregazione **forte**);
- **ereditarietà**: una classe può estendere un’altra classe.
- ...

Uso

L’uso o associazione è la relazione più semplice che intercorre fra due classi.

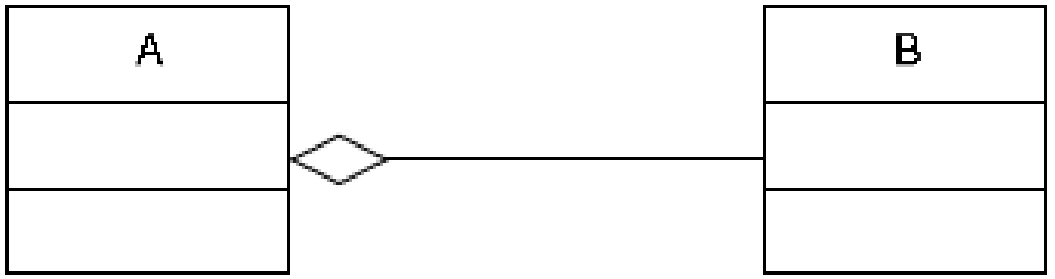
Per definizione diciamo che una classe A usa una classe B se:

- un metodo della classe A invia messaggi agli oggetti della classe B , oppure
- un metodo della classe A crea, restituisce, riceve oggetti della classe B .



Aggregazione

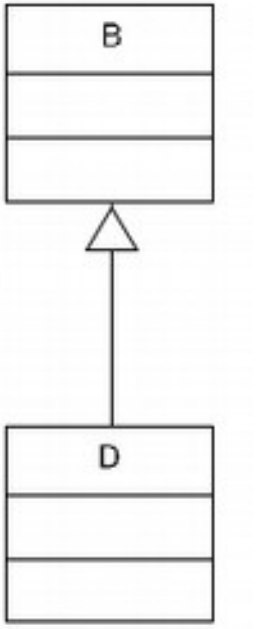
- una classe A **aggrega** oggetti di una classe B quando la classe A **contiene** oggetti della classe B
- è un caso speciale della relazione di uso
- relazione **has-a** (ha-un)



il rombo è attaccato alla classe che contiene l’altra

Ereditarietà

- **riuso** del codice
- classe derivata o **sottoclasse**
- classe base o **superclasse**
- Relazione **is-a** (è-un)

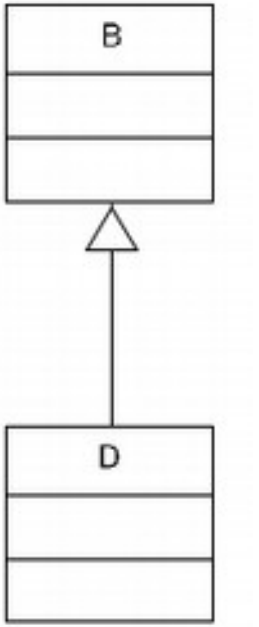


Ereditarietà: si ottiene il riuso del codice

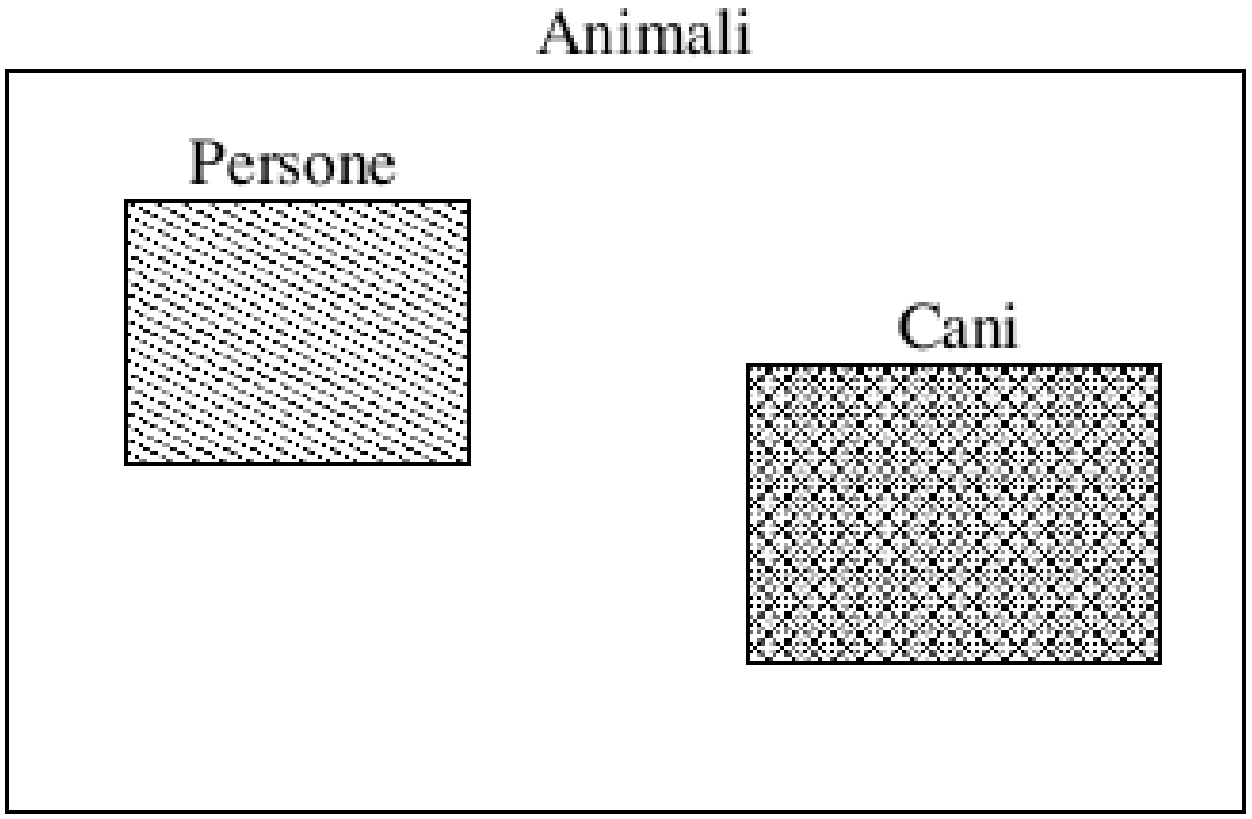
Consideriamo una **classe base B** che ha un metodo f(...) ed una **classe derivata D** che eredita da B .

La classe D può usare il metodo f(...) in tre modi:

- lo **eredita**: quindi f(...) può essere usato come se fosse un metodo di D ;
- lo **riscrive** (*override*): cioè si da un nuovo significato al metodo riscrivendo la sua implementazione nella classe derivata, in modo che tale metodo esegua una azione diversa;
- lo **estende**: cioè richiama il metodo f(...) della classe base ed aggiunge altre operazioni.



P.es. la classe Persona deriva da una classe molto più grande, cioè la classe degli Animali



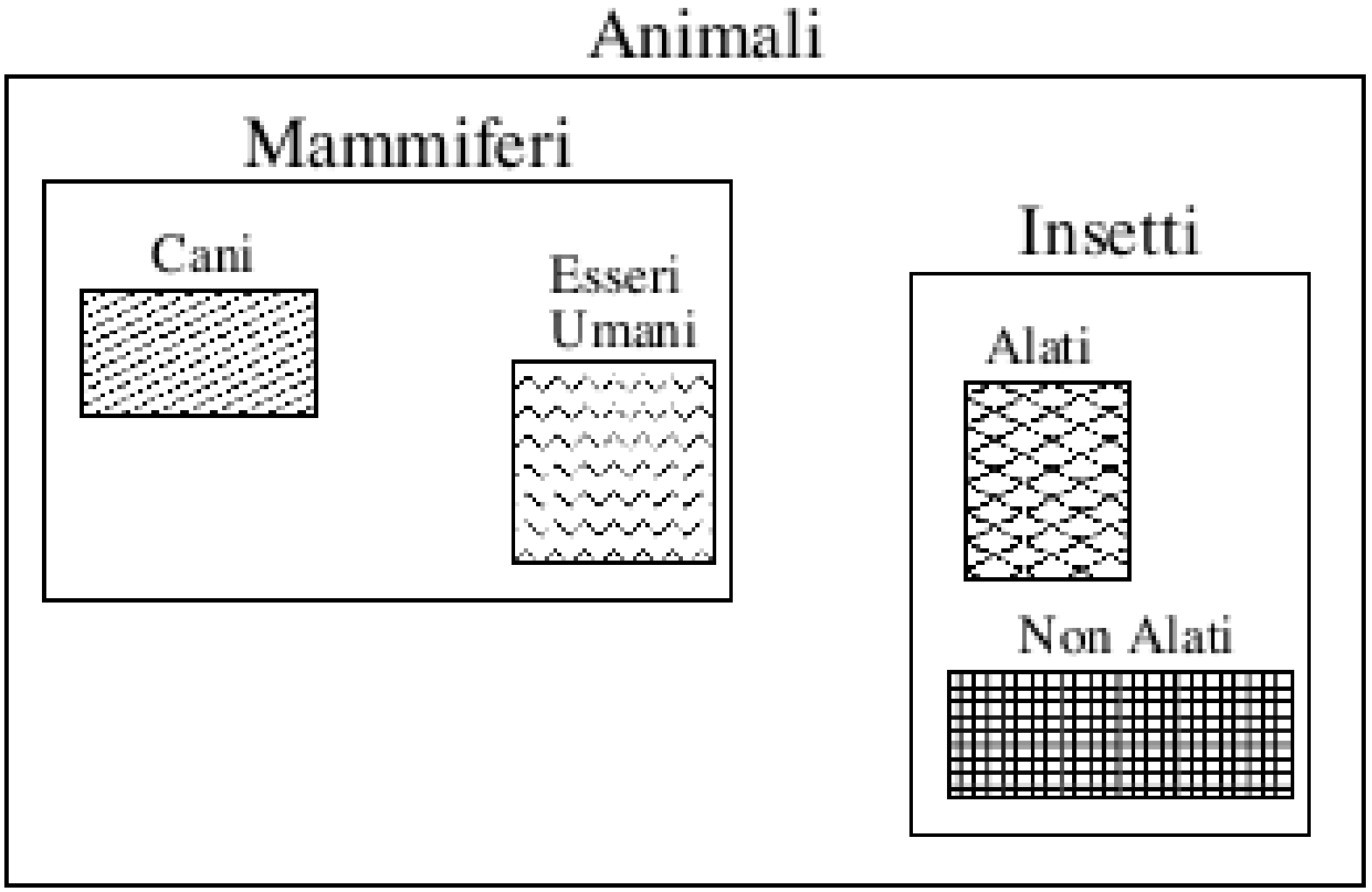
Quando usare l’ereditarietà

- Usare l’ereditarietà solo quando il legame fra la classe base e la classe derivata è **per sempre**, cioè dura per tutta la vita degli oggetti della classe derivata.
- Se tale legame non è duraturo è meglio usare **l’aggregazione** al posto della **specializzazione**.

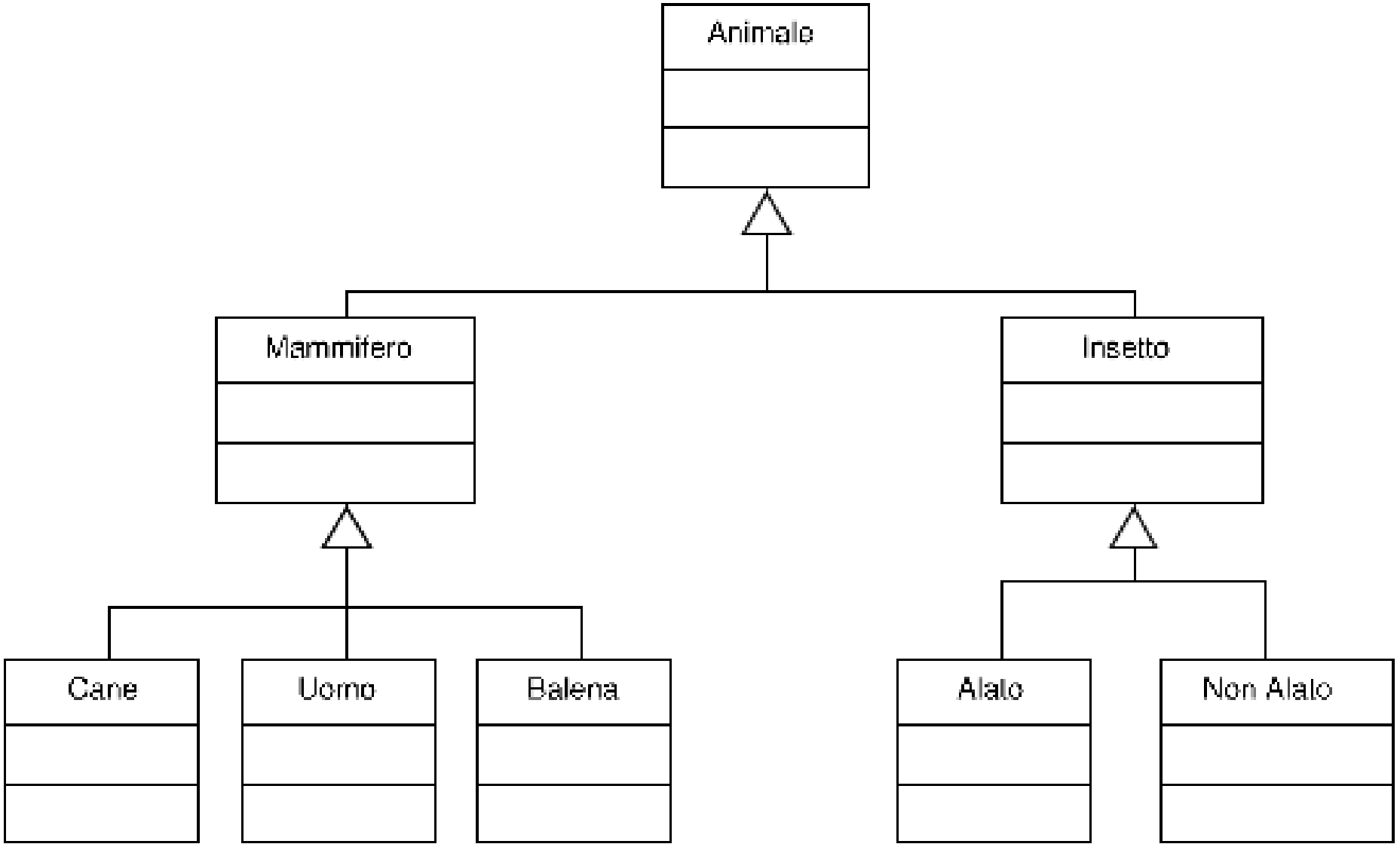
Polimorfismo

- La parola **polimorfismo** deriva dal greco e significa letteralmente **molte forme**.
- Nella **OOP** tale termine si riferisce ai **metodi**: per definizione, il polimorfismo è la **capacità di un oggetto**, la cui classe fa parte di una gerarchia, di chiamare la **versione corretta** di un metodo.
- Quindi il polimorfismo è necessario quando si ha una gerarchia di classi.

gerarchia di classi



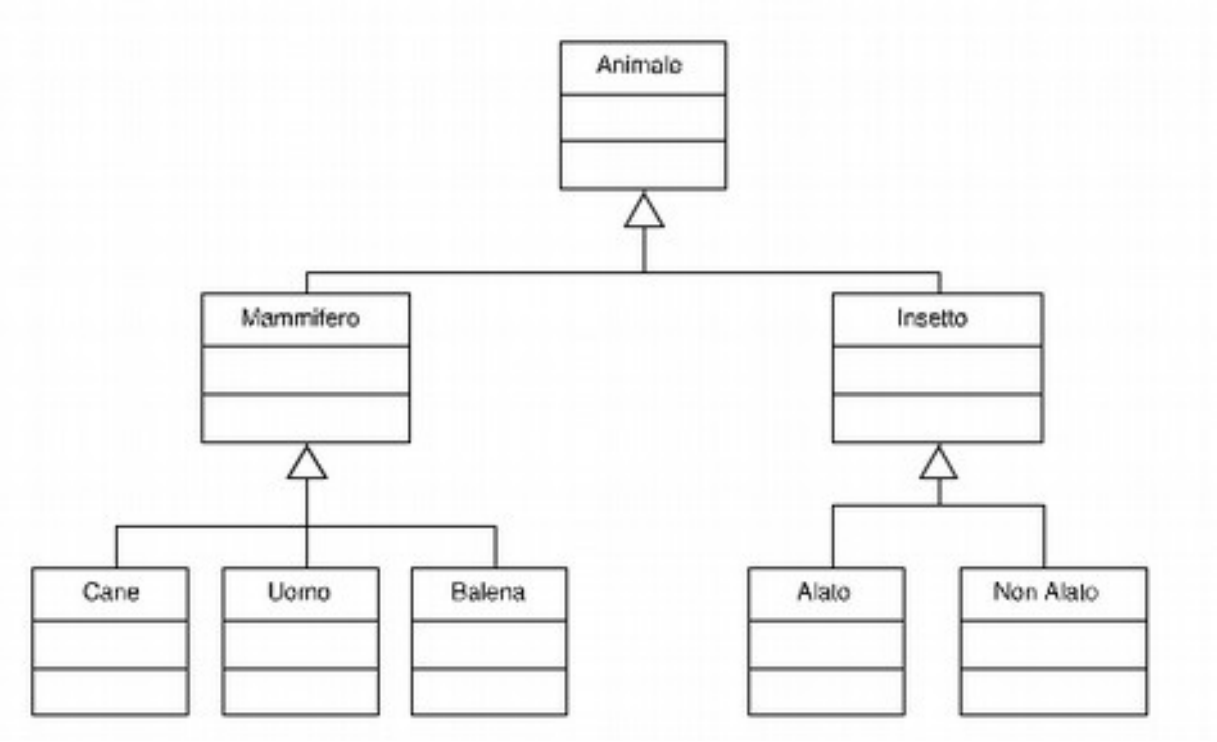
gerarchia di classi



Classi astratte

esaminiamo i metodi della classe base Animale

- comunica(…), mangia(), siMuove(), ...



classe astratta

- Una classe che ha almeno un metodo astratto si dice **classe astratta**
- può anche contenere dei metodi non astratti
- è **troppo generica** per essere istanziata
- usata come un **contenitore di comportamenti** (operazioni) comuni che ogni classe derivata sa come implementare