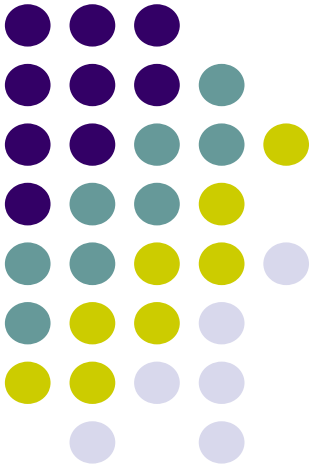
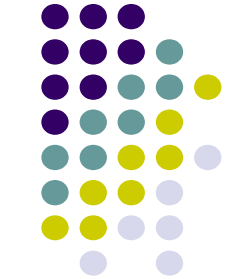
Corso di Programmazione

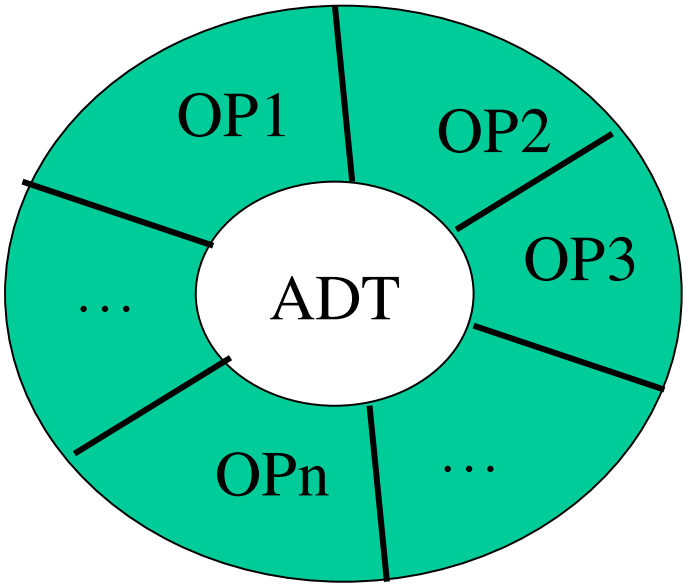
Programmazione Object Oriented



Definizione (ADT)



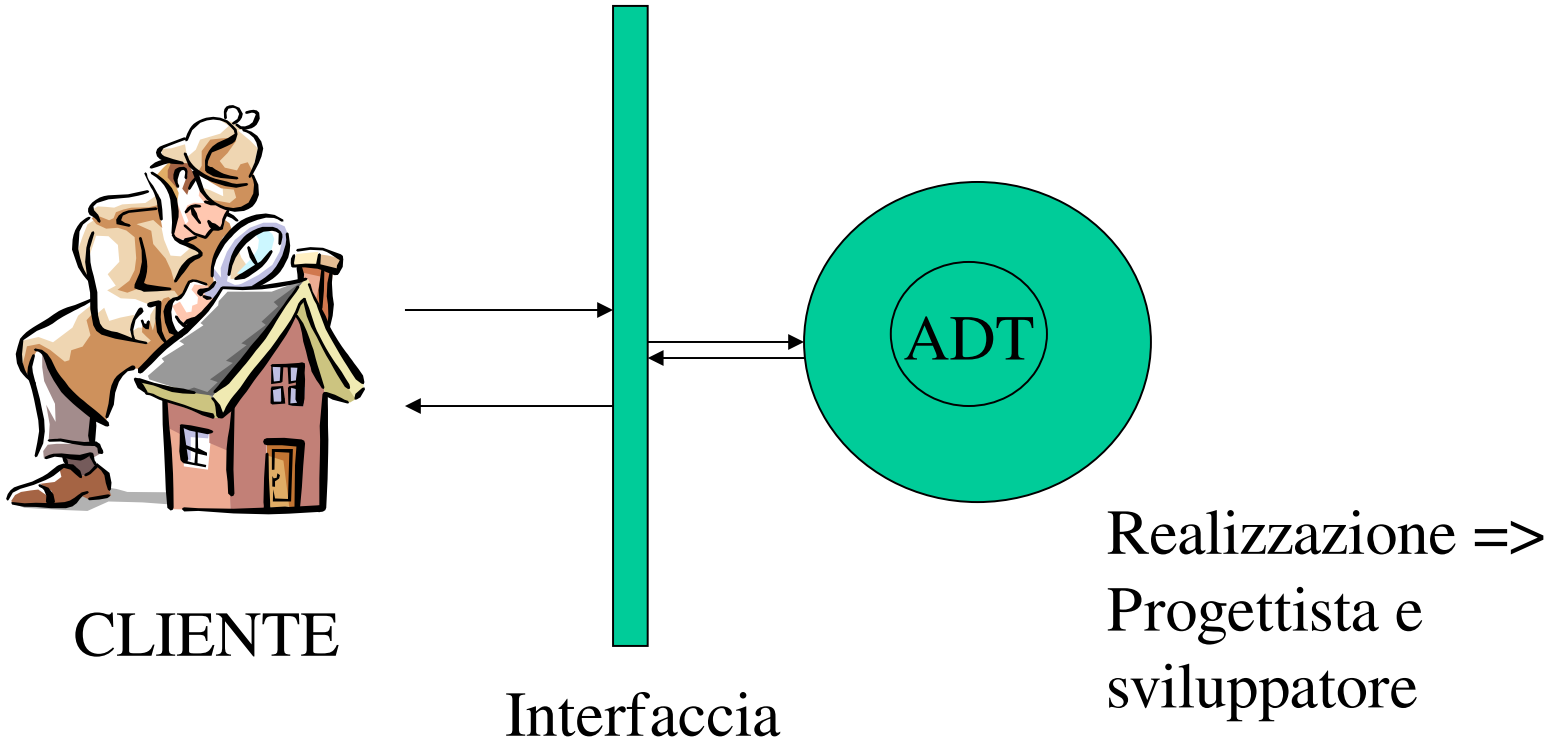
- Un tipo di dato astratto (Abstract Data Type: ADT) è una struttura dati incapsulata dalle operazioni consentite su di essa
- Non è possibile accedere alla struttura dati (né in lettura né in scrittura) se non attraverso le operazioni definite su di essa



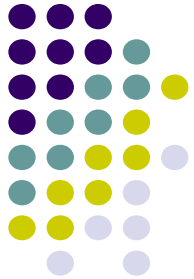
Metodologicamente:

Interfaccia

Realizzazione



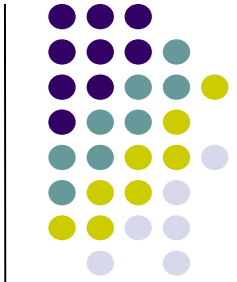
Clienti e sviluppatori



- Il cliente fa uso delle funzionalità offerte per realizzare procedure di un'applicazione o per costruire operazioni su dati più complessi
- Il progettista e lo sviluppatore devono realizzare le astrazioni e le funzionalità previste per il dato facendo uso di un linguaggio di programmazione o appoggiandosi sulle primitive di un altro tipo di dato già disponibile

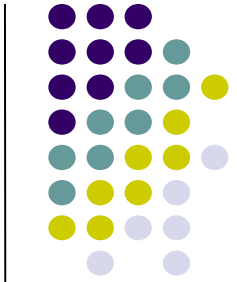
Un progettista o sviluppatore può essere il cliente di un altro tipo di dato astratto

Vantaggi principali



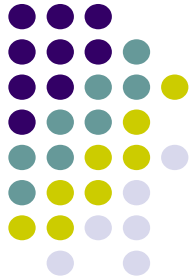
- La struttura dei dati non può venire alterata da operazioni scorrette
- La realizzazione della struttura dei dati può essere modificata senza influenzare i moduli che ne fanno uso

Tecnicamente...



- Utilizzo di moduli
- Information Hiding
- Descrizione (o specifica) dell'interfaccia

La programmazione ad oggetti (OOP – object oriented programming)



- Si individuano le *classi* di *oggetti* (entità del mondo reale o concettuale) che caratterizzano il dominio applicativo
  - le diverse classi vengono poi modellate, progettate e implementate (come ADT)
  - ogni classe è descritta da un'*interfaccia* che specifica il comportamento degli oggetti della classe
- L'applicazione si costruisce con l'*approccio ascendente* (*bottom-up*), assemblando oggetti e individuando le modalità con cui questi devono collaborare per realizzare le diverse funzionalità dell'applicazione

I tre principi OOP

Tutti i linguaggi di programmazione orientati agli oggetti offrono i seguenti meccanismi che implementano il modello orientato agli oggetti.

Incapsulamento

Il meccanismo che controlla l'accesso ad una struttura dati solo attraverso opportune operazioni

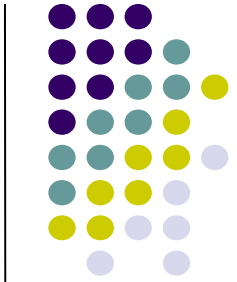
Ereditarietà

Il meccanismo tramite il quale un oggetto acquisisce le proprietà di un altro. Fondamentale perché sostiene il concetto della classificazione gerarchica. Modella cioè una della possibili ***relazioni tra oggetti.***

Polimorfismo

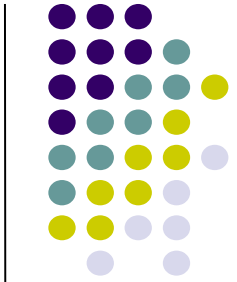
Il meccanismo tramite il quale è possibile progettare un'interfaccia generica per specificare una classe generale di azioni (un'interfaccia, più implementazioni).

Classi e oggetti



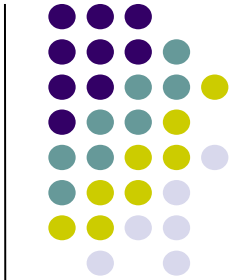
- Le classi e gli oggetti sono concetti base della programmazione orientata agli oggetti (OOP) utilizzati per rappresentare concetti ed entità del mondo reale.
- La classe rappresenta un gruppo di oggetti con proprietà e comportamento simili.
  - Ad esempio, il tipo animale Gatto è una classe mentre un particolare gatto chiamato Gerry è un oggetto della classe Gatto.

Classi e oggetti



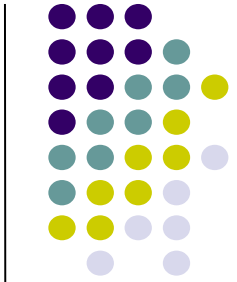
- Una classe in Java è un insieme di oggetti che condividono caratteristiche/comportamenti comuni e proprietà/attributi comuni.
  - È uno schema o un prototipo definito dall'utente da cui vengono creati gli oggetti.
- Un oggetto in Java è un'unità base della programmazione orientata agli oggetti e rappresenta entità della vita reale.
  - Gli oggetti sono le istanze di una classe create per utilizzare gli attributi e i metodi di una classe.
  - Un tipico programma Java crea molti oggetti che interagiscono tra di loro tramite invocazione di metodi.
- I concetti di metodi e attributi *saranno chiari tra poco*

Classi e oggetti (Esempio)



- *Esempio di classe Calciatore.*
  - *Ogni calciatore può essere caratterizzato dei seguenti "attributi"*
    1. *Cognome*
    2. *Società di appartenenza*
    3. *Ruolo*
    4. *Numero di maglia*
    5. *… altri? Ovviamente dipende dal mio, vostro P.O.V.*
- *Secondo l'astrazione precedente possibili oggetti della classe Calciatore sono:*
  - **Kvara (Kvaratskhelia, SSCN, Attaccante, 77)**
  - **IL_Professore (Rui, SSCN, Difensore, 6)**
  - **Cholito (Simeone, SSCN, Attaccante, 18)**

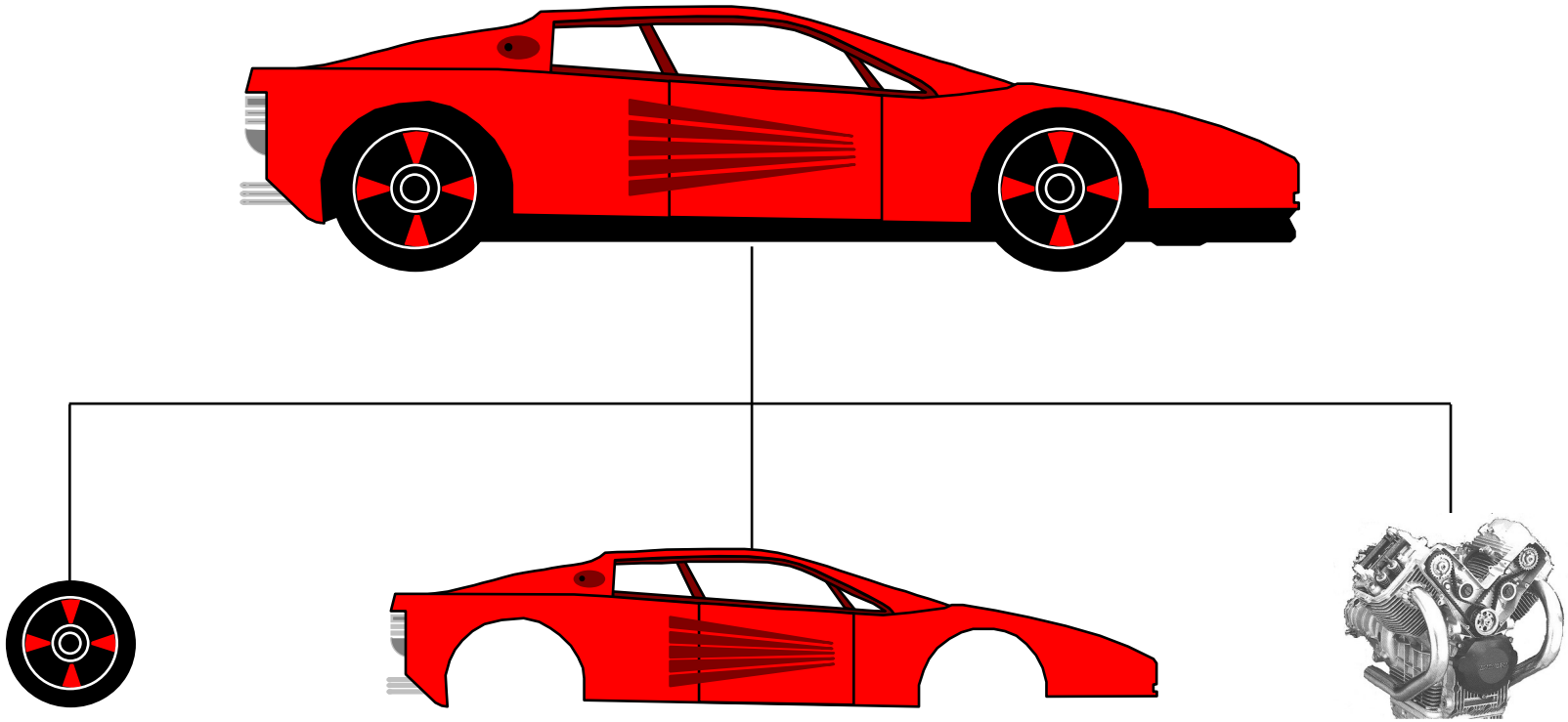
Relazioni tra classi (e oggetti)



- Composizione
- Generalizzazione-Specializzazione
- Associazione

Relazioni: Composizione

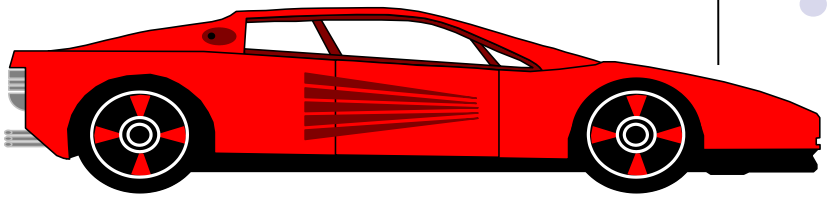
Un oggetto complesso può essere composto di oggetti più semplici detti *componenti*



Ogni oggetto di tipo Auto è composto da 4 oggetti di tipo Ruota, da un oggetto di tipo carrozzeria, da un oggetto di tipo Motore, etc…

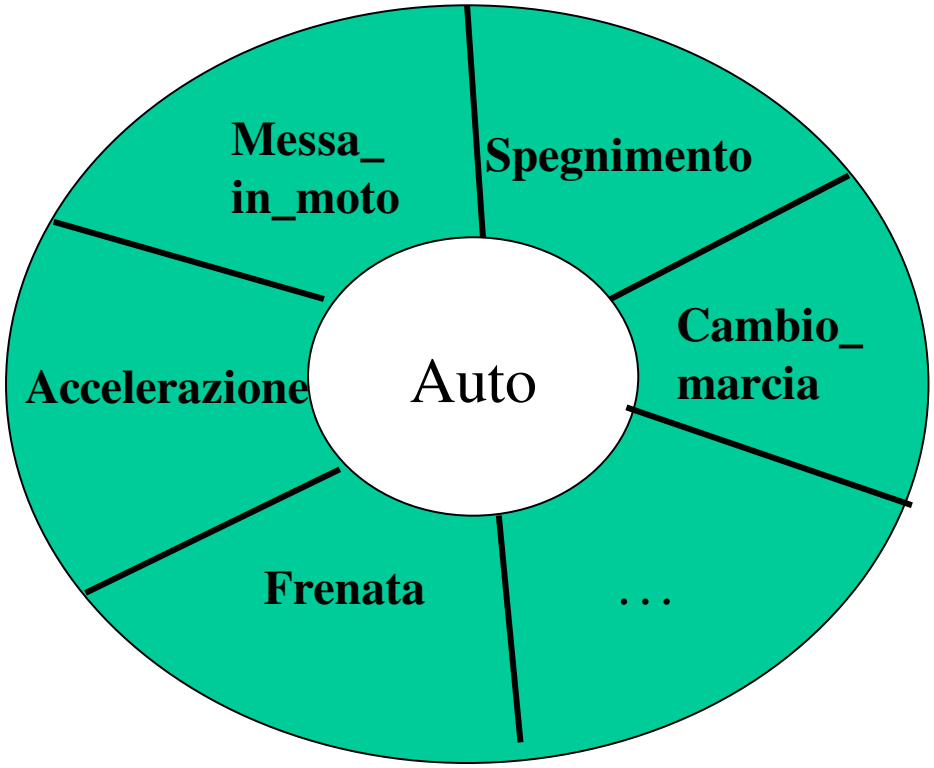
Un esempio di oggetto “composto”

Funzioni	Dati
- Avviati	- Targa
- Fermati	- Colore
- Accelera	- Cilindrata motore
- …	- …



Il conducente interagisce con una interfaccia per effettuare le operazioni consentite sull’automobile:

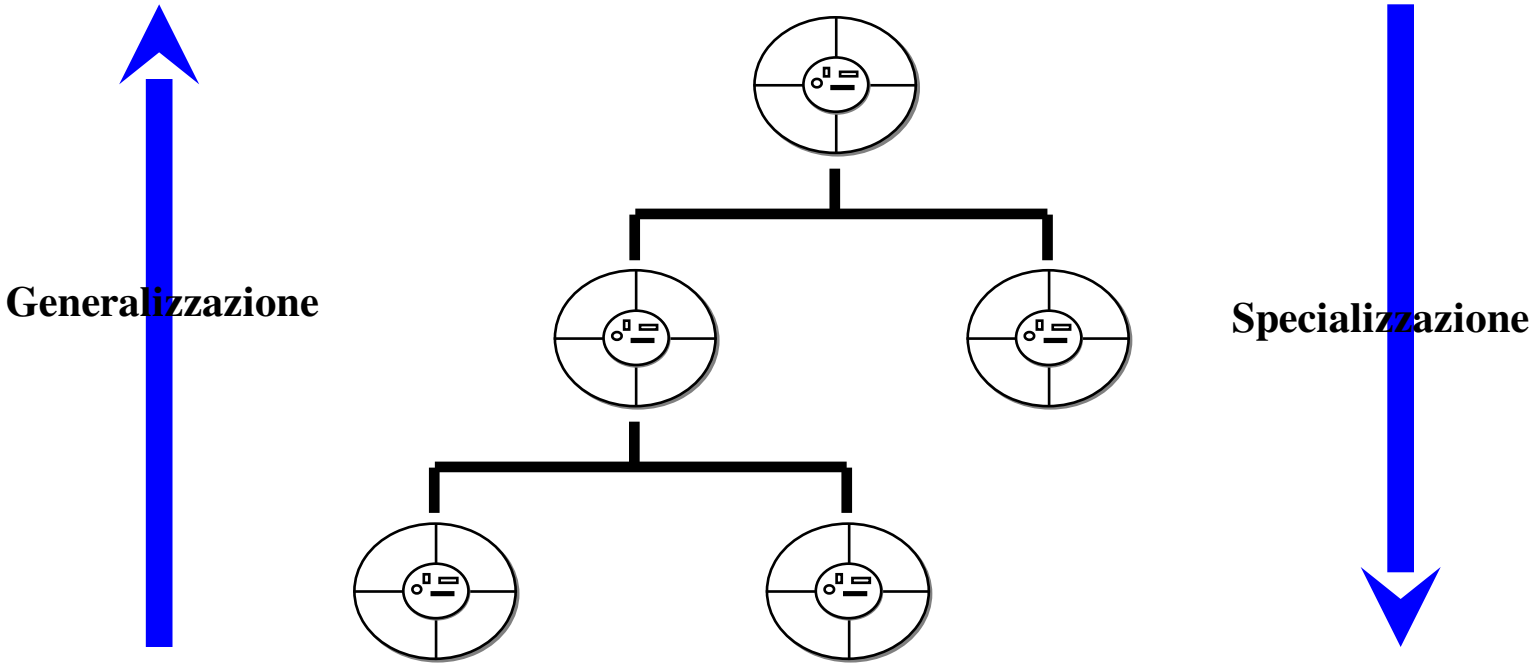
- Pedale del freno
- Pedale dell’acceleratore
- Leva del cambio
- Sistema di accensione
- …



Relazioni: Gen-Spec

Generalizzazione: dal particolare al generale

Specializzazione dal comportamento generale ad un comportamento specializzato



*Nel paradigma a oggetti, col meccanismo d**ell'ereditarietà** ci si concentra sulla creazione di tassonomie del sistema in esame*

Relazioni: Associazione

Una Ferrari e il suo pilota: la relazione è definita da un "contratto" .

Un oggetto può anche essere associato a più oggetti di un'altra classe. Ad esempio una persona può possedere più di una automobile, un libro avere più autori e un autore aver scritto più libri…

Relazioni: Associazione

Una squadra di calcio può essere associata a più calciatori (ogni squadra ha una rosa di calciatori tesserati)

Relazioni: Associazione

Un calciatore può essere associato più squadre (ad esempio tutte le squadre in cui quel giocatore ha militato)

Erediatarietà

PERSONA

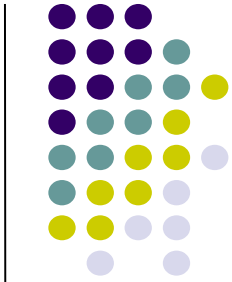
STUDENTE

LAVORATORE

Uno studente e un lavoratore sono persone: hanno quindi in comune una parte del loro comportamento che è quello di un persona.

In particolare poi uno studente può anche essere un lavoratore!!!
Uno studente-lavoratore ha in comune una parte del suo comportamento con tre diversi tipi di oggetto.

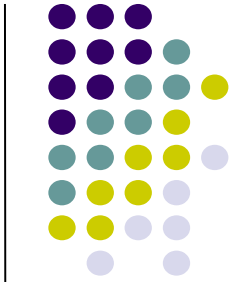
Polimorfismo



Introduce la possibilità di chiamare una funzione (cioè di inviare un “messaggio”) senza sapere a tempo di compilazione a quale tipo di oggetto verrà applicato (inviato):

Ad esempio: “Spostati!!!”.... Ma non sai -se non a tempo di esecuzione- chi si sposterà... Se lo studente, o il lavoratore, o in generale una persona...

Linguaggi orientati agli oggetti



Forniscono i meccanismi per:

- Definire classi e oggetti
- Realizzare l'ereditarietà
- Realizzare il polimorfismo

Vantaggi della programmazione OO

Rispetto alla programmazione tradizionale, la programmazione orientata agli oggetti (OOP) offre vantaggi in termini di:

- ✓ **modularità**: le classi sono i moduli del sistema software;
- ✓ **coesione dei moduli**: una classe è un componente software ben coeso in quanto rappresentazione di una unica entità;
- ✓ **disaccoppiamento dei moduli**: gli oggetti hanno un alto grado di disaccoppiamento in quanto i metodi operano sulla struttura dati interna ad un oggetto; il sistema complessivo viene costruito componendo operazioni sugli oggetti;
- ✓ ***information hiding***: sia le strutture dati che gli algoritmi possono essere nascosti alla visibilità dall'esterno di un oggetto;
- ✓ **riuso**: l'ereditarietà consente di riusare la definizione di una classe nel definire nuove (sotto)classi; inoltre è possibile costruire librerie di classi raggruppate per tipologia di applicazioni;
- ✓ **estensibilità**: il polimorfismo agevola l'aggiunta di nuove funzionalità, minimizzando le modifiche necessarie al sistema esistente quando si vuole estenderlo.

Riferimenti