Programmazione orientata agli oggetti
Il modello “classico”

Il concetto di oggetto

Un oggetto è un'entità dotata di stato e di comportamento

- In pratica un oggetto aggrega in un entità unica e indivisibile una struttura dati (**stato**) e l'insieme di operazioni che possono essere svolte su di essa (**comportamento**)
- E' quindi un'insieme di variabili e di procedure: le variabili vengono comunemente chiamate **attributi** dell'oggetto
- Le procedure vengono comunemente chiamate **metodi**
- **Sparisce il dualismo di fondo del modello procedurale**

Incapsulamento e astrazione

- Lo stato di un oggetto:
  - Non è accessibile all'esterno
  - Può essere visto e manipolato solo attraverso i metodi
- Quindi: lo stato di un oggetto è **protetto**

  → **Il modello ad oggetti supporta in modo naturale l'incapsulamento**
- Dal momento che dall'esterno possiamo vedere solo i metodi c'è una separazione netta tra cosa l'oggetto è in grado di fare e come lo fa
- Abbiamo quindi una separazione netta fra **interfaccia** e **implementazione**

  → **Il modello ad oggetti supporta in modo naturale l'astrazione**

Classi e oggetti - 1

- Per poter operare con gli oggetti abbiamo bisogno di un meccanismo che ci consenta di:
  - Definire una struttura e un comportamento
  - Creare un numero qualunque di oggetti con identica struttura e comportamento ma con identità diversa
- Nella programmazione procedurale abbiamo il concetto di **tipo**: una volta definito un tipo possiamo dichiarare più variabili identiche fra di loro
- Se vogliamo avere un comportamento dinamico ci serve anche un meccanismo di **creazione**, che si comporti come la malloc in C
- Nell'OOP questi due ruoli vengono svolti da un'entità chiamata **classe**

Classi e oggetti - 2

- Quindi una **classe** è un entità che permette di
  - Definire la struttura e il comportamento di un oggetto
  - Creare un numero qualunque di oggetti con la struttura specificata
- Gli oggetti creati da una classe si chiamano **istanze** della classe

  → **Ogni oggetto è istanza di una qualche classe**
- Tutte le istanze di una classe hanno la **stessa struttura** (lo stato è fatto nello stesso modo) e lo **stesso comportamento** (l'insieme dei metodi definiti dalla classe)
- Ogni istanza possiede un **proprio stato** ("contenuto delle variabili") e una **propria identità.**

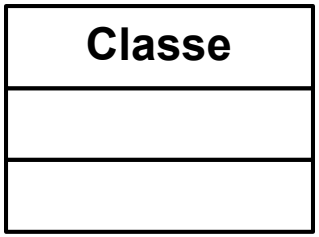
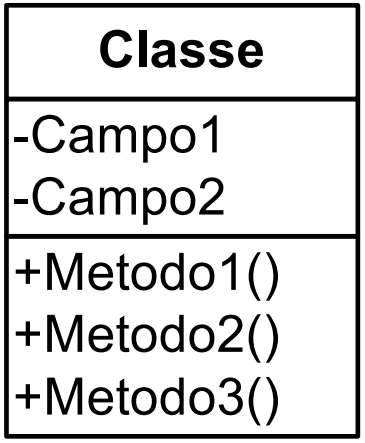
Classi e oggetti - 3

Nel modello "classico" dell'OOP esistono solo classi e oggetti

- Gli oggetti sono entità **dinamiche**:
  - Possono essere creati in qualunque momento a partire da una classe
  - Vengono distrutti quando non servono più
- Le classi sono invece entità **statiche**:
  - Sono sempre disponibili durante l'esecuzione di un'applicazione
- Quando si crea un oggetto si ottiene un **riferimento** all'istanza appena creata
- I riferimenti rappresentano l'unico modo per comunicare con gli oggetti

Notazione grafica UML

- UML (*Unified Modeling Language*, "linguaggio di modellazione unificato") è un **linguaggio di modellazione e specifica** basato sul paradigna object-oriented
- Nucleo del linguaggio definito nel **1996** da **Grady Booch**, **Jim Rumbaugh** e **Ivar Jacobson** sotto l'egida dello **OMG**, che tuttora gestisce lo standard di UML
- Standard **industriale** unificato.
- Gran parte della letteratura OOP usa UML per descrivere soluzioni analitiche e progettuali in modo sintetico e comprensibile

Rappresentazione delle classi

- Diagramma delle Classi
- In UML le classi vengono rappresentate con un rettangolo
- Nella forma sintetica compare solo il nome della classe mentre nella forma estesa vengono indicati anche i campi e i metodi