Introduzione diagramma delle classi UML

Fattori di qualità del software su cui agisce la programmazione ad oggetti

- Esterni
  - Estendibilità
  - Riusabilità
- Interni
  - Strutturazione
  - Modularità
  - Comprensibilità

Caratteristiche della programmazione ad oggetti

- Connessione esplicita fra funzioni e dati
- Incapsulamento
- Ereditarietà
- Concetto di classe
- Enfasi sulla rappresentazione

Metodi per la Progettazione SW

- Orientati alle funzioni (metodologie del passato)
  - diagrammi funzionali
  - diagrammi di flusso di controllo
  - diagrammi di flusso di dati
- Orientati agli oggetti (metodologie attuali)
  - Booch
  - OOSE (Jacobson)
  - OMT (Rambaugh)
  - Unified Modeling Language (UML)
    - *G. Booch, J. Rumbaugh, I. Jacobson, "The unified modeling language user guide", Addison Wesley, 1999.*
    - *http://www.rational.com/uml* 1999:

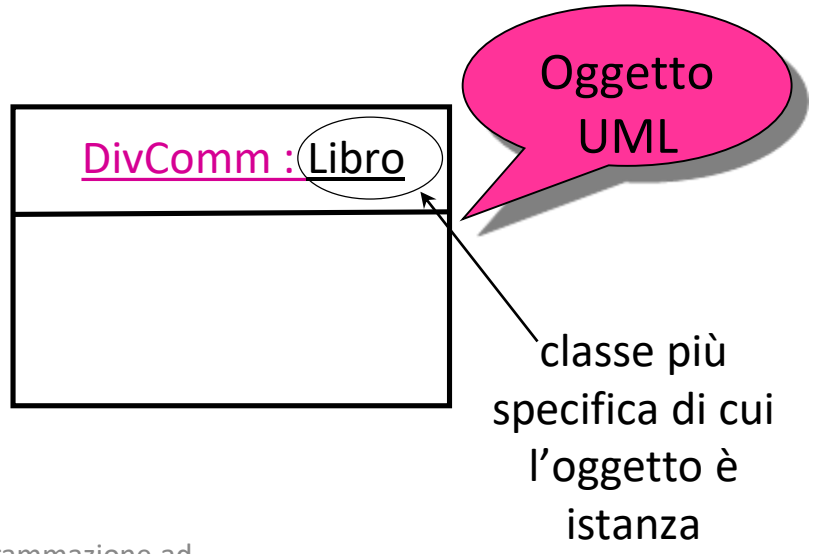
Diagrammi UML

- Diagrammi strutturali:
  - Diagramma delle classi e degli oggetti (class and object diagram)
- Diagrammi comportamentali:
  - Diagramma dei casi d'uso (use case diagram),
  - Diagramma degli stati e delle transizioni (state/transition diagram),
  - Interaction (Sequence e Collaboration diagram),
  - Activity diagram
- Diagrammi architetturali:
  - Component diagram
  - Deployment diagram

Oggetti in UML

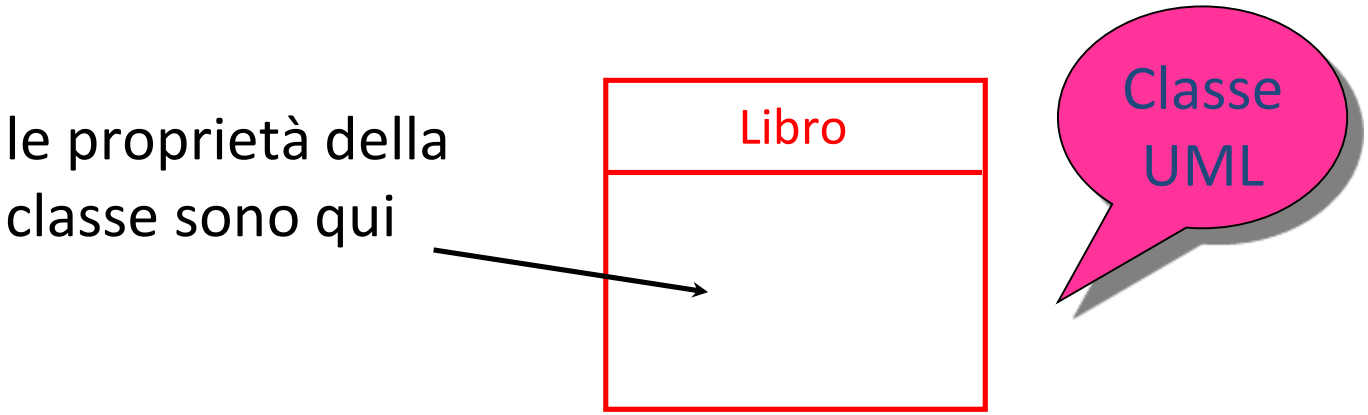
- Un oggetto in UML modella un elemento del dominio di analisi che
  - ha vita propria
  - è identificato univocamente mediante l'identificatore di oggetto
  - è istanza di una classe (la classe più specifica – un oggetto può essere istanza di più classi, ma esiste sempre quella più specifica)

- DivComm è l'identificatore di oggetto
- Libro è la classe (più specifica) di cui l'oggetto è istanza
- Si noti la sottolineatura



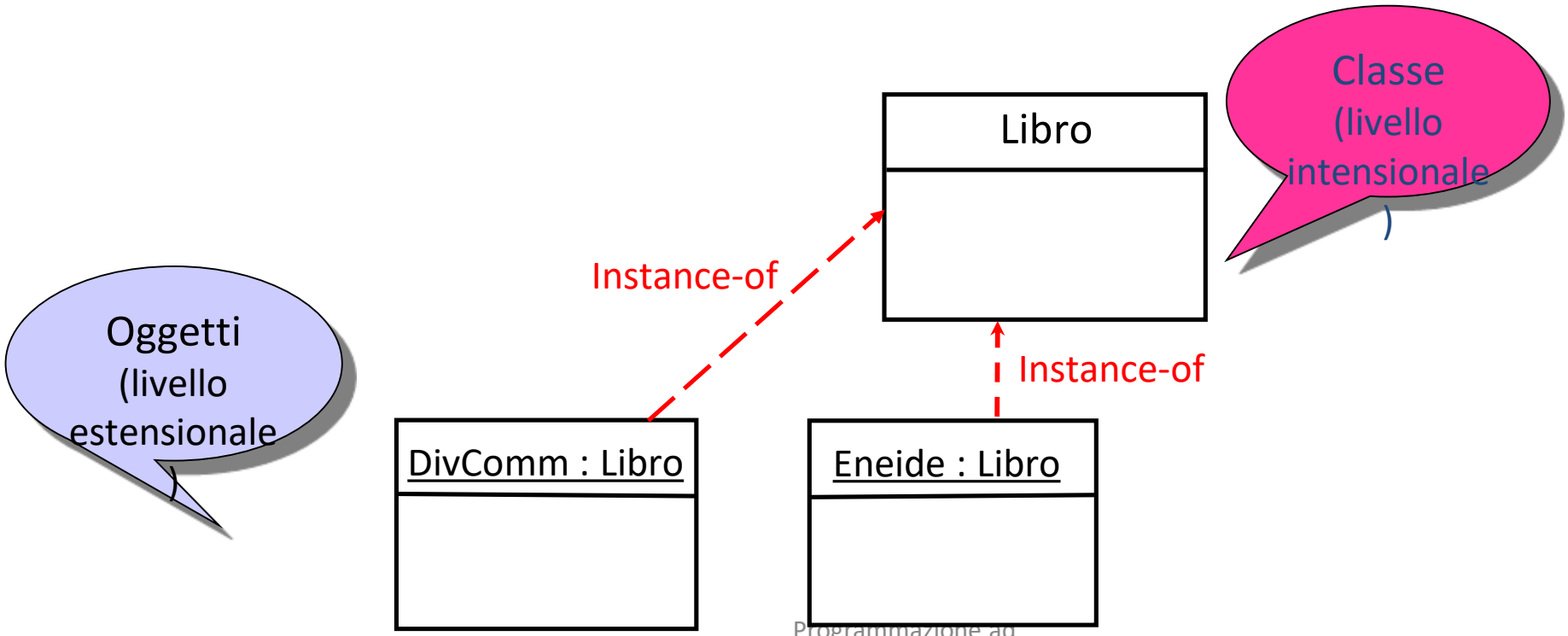
Classi in UML

- Una classe modella un insieme di oggetti omogenei (le istanze della classe) con associate proprietà statiche e dinamiche (operazioni). Una classe descritta da:
  - un nome
  - un insieme di proprietà (astrazioni delle proprietà comuni degli oggetti che sono istanze delle classi)
  - un insieme di operazioni (astrazioni delle operazioni comuni che si possono effettuare sugli oggetti che sono istanze della classe)



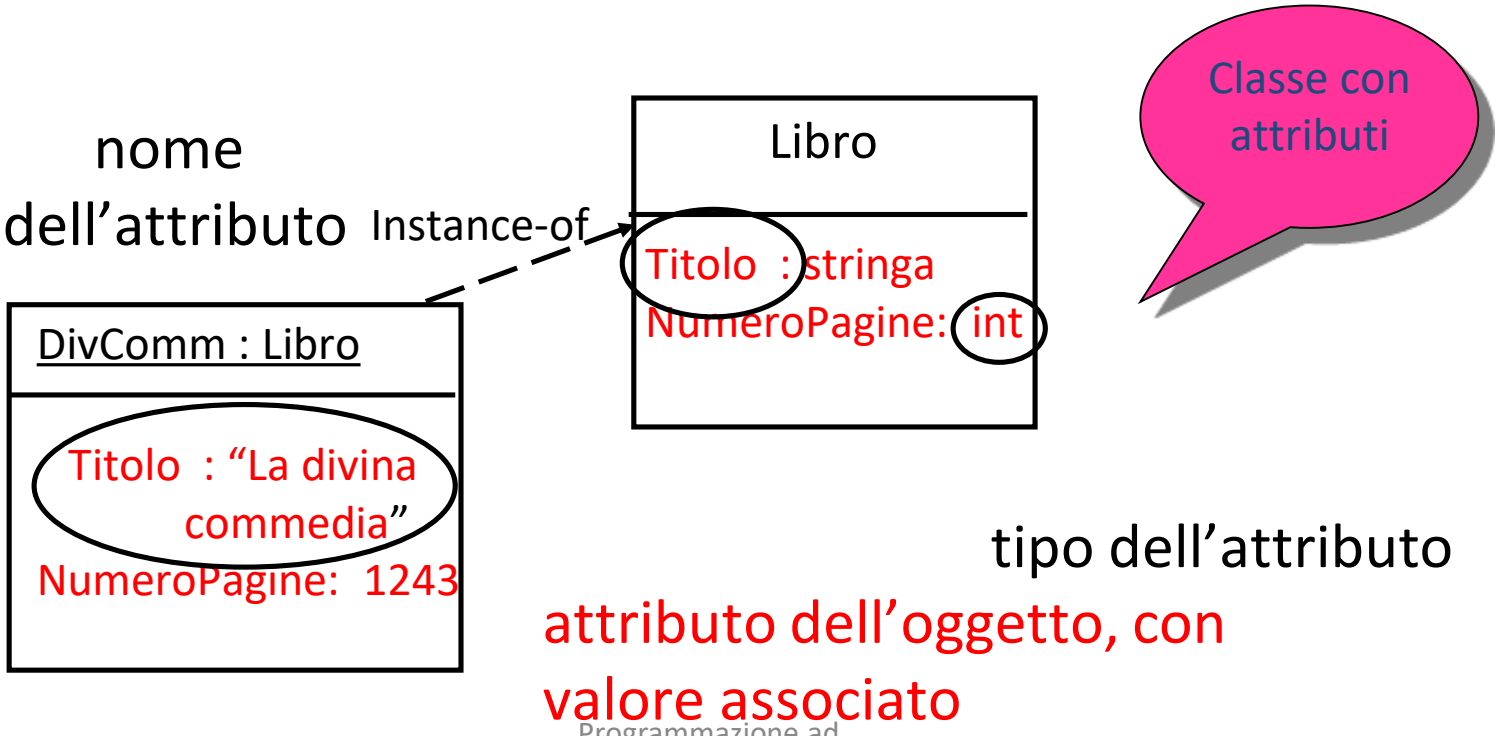
Rapporto tra classi e istanze

- Tra un oggetto che è istanza di una classe C e la classe C si traccia un arco Instance-of
- Gli oggetti formano il livello estensionale, mentre le classi a livello intensionale

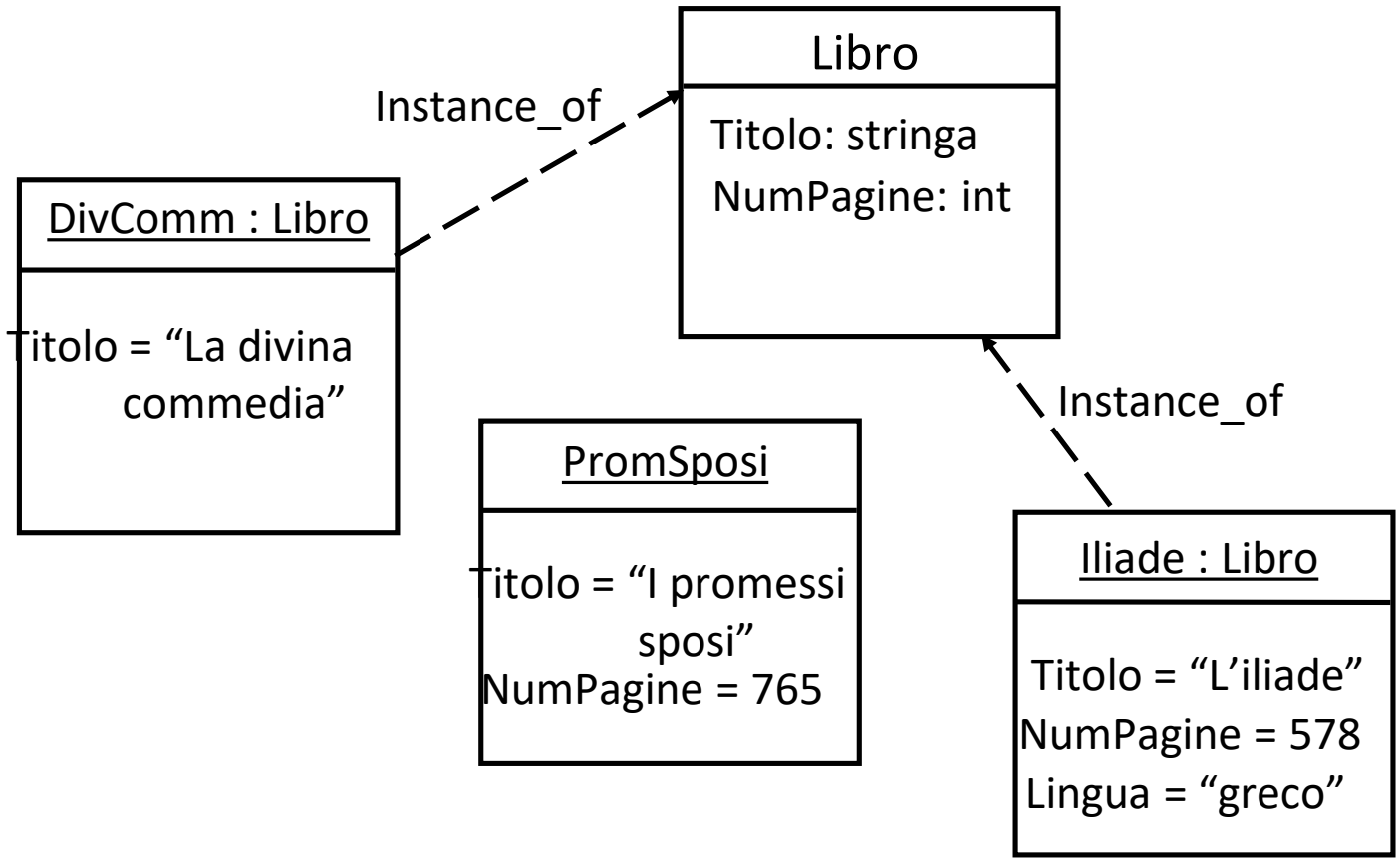


Attributi di classi ed oggetti in UML

- Un attributo è proprietà locale (indipendente da altri oggetti) della classe con un nome ed un tipo di valori associati
- Un attributo di una classe è valido per tutte le istanze della classe, ovvero
- Gli attributi di una classe determinano gli attributi delle istanze (anche se non i valori degli attributi delle istanze)

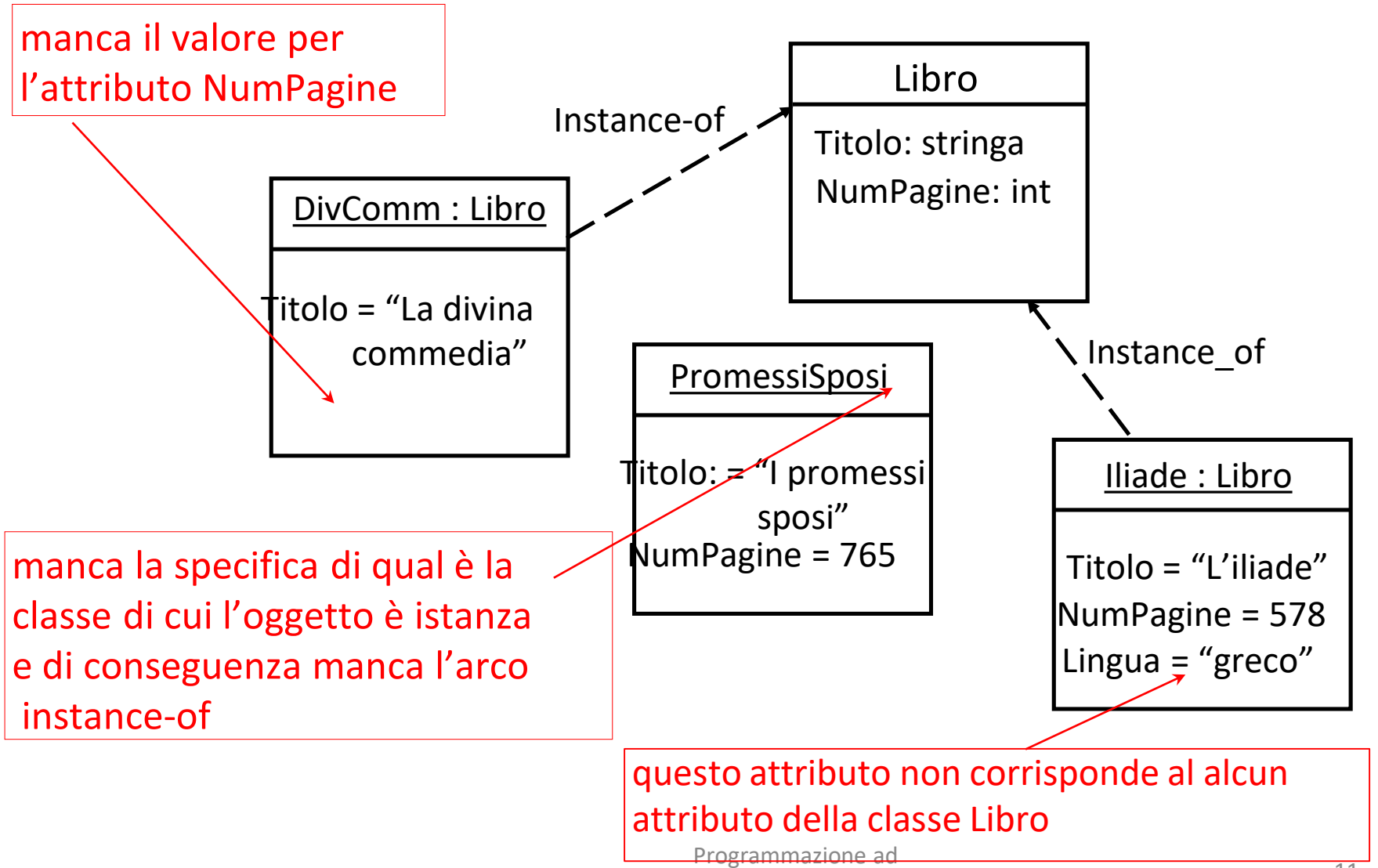


Esercizio 1



Il diagramma è corretto? Se no, quali sono gli errori?

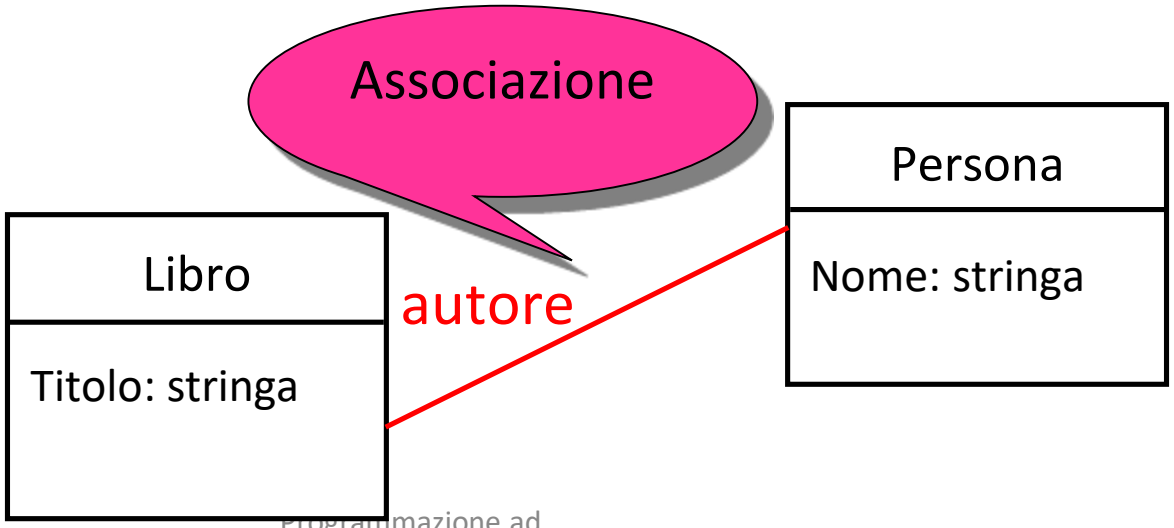
Errori Tipici in un Diagramma



Proprietà di classi: associazioni in UML

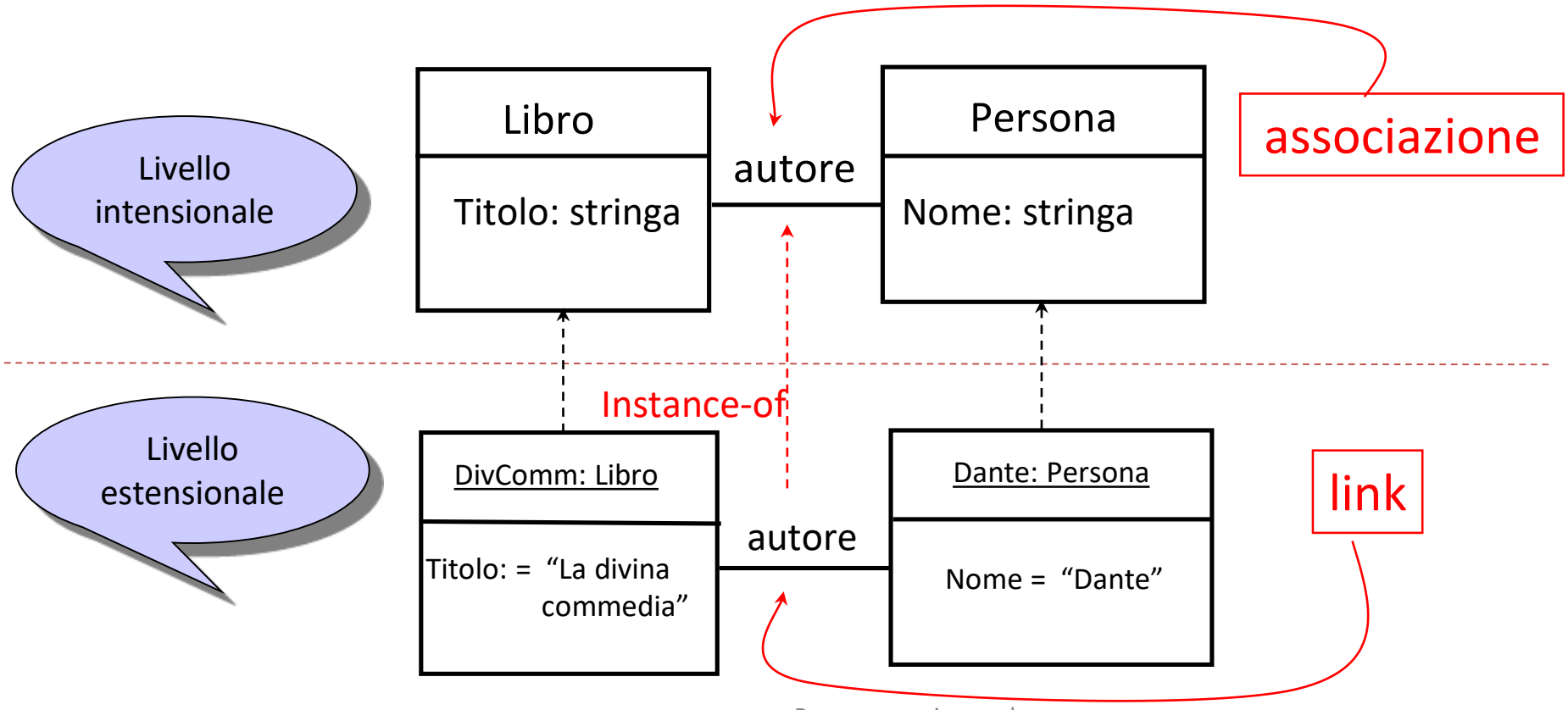
- Vediamo solo associazioni tra due classi (ma le associazioni possono coinvolgere N classi)
- Una associazione (o relazione) tra una classe C1 ed una classe C2 modella una relazione matematica tra l'insieme delle istanze di C2 e l'insieme delle istanze di C2
- Gli attributi modellano proprietà di una classe, le associazioni modellano proprietà che coinvolgono più classi.

Nota: autore è una proprietà sia di libro sia di persona



Istanze di associazioni: link

- Le istanze di associazioni si chiamano link: se A è una associazione tra le classi C1 e C2, una istanza di A è un link tra due oggetti (una coppia), uno della classe C1 e l'altro della classe C2



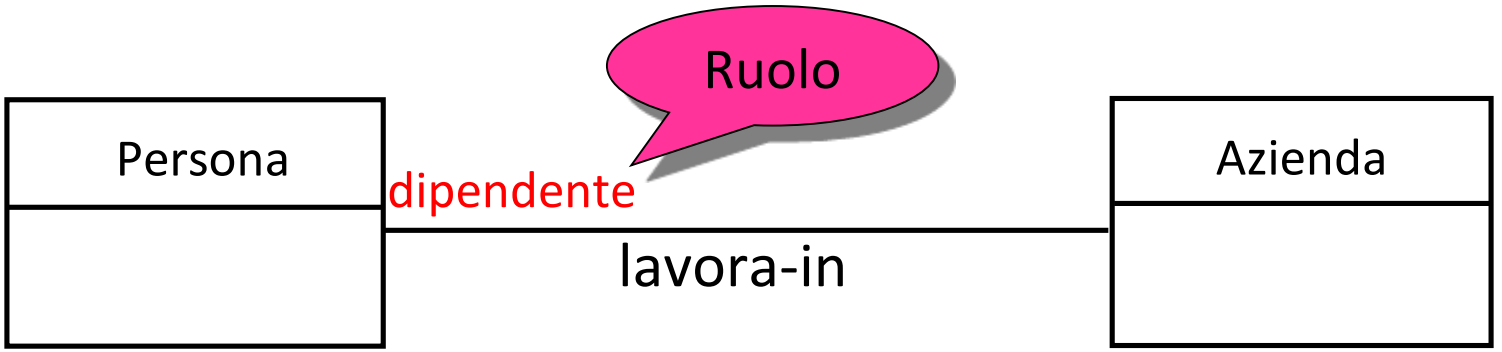
Esercizio 2

Tracciare il diagramma delle classi corrispondenti alle seguenti specifiche:

Si vogliono modellare gli studenti (con nome, cognome, numero di matricola, età), il corso di laurea in cui sono iscritti, ed i corsi di cui hanno sostenuto l'esame. Di ogni corso di laurea interessa il codice e il nome. Di ogni corso interessa il nome e la disciplina a cui appartiene (ad esempio: matematica, fisica, informatica, ecc.).

Ruoli nelle associazioni

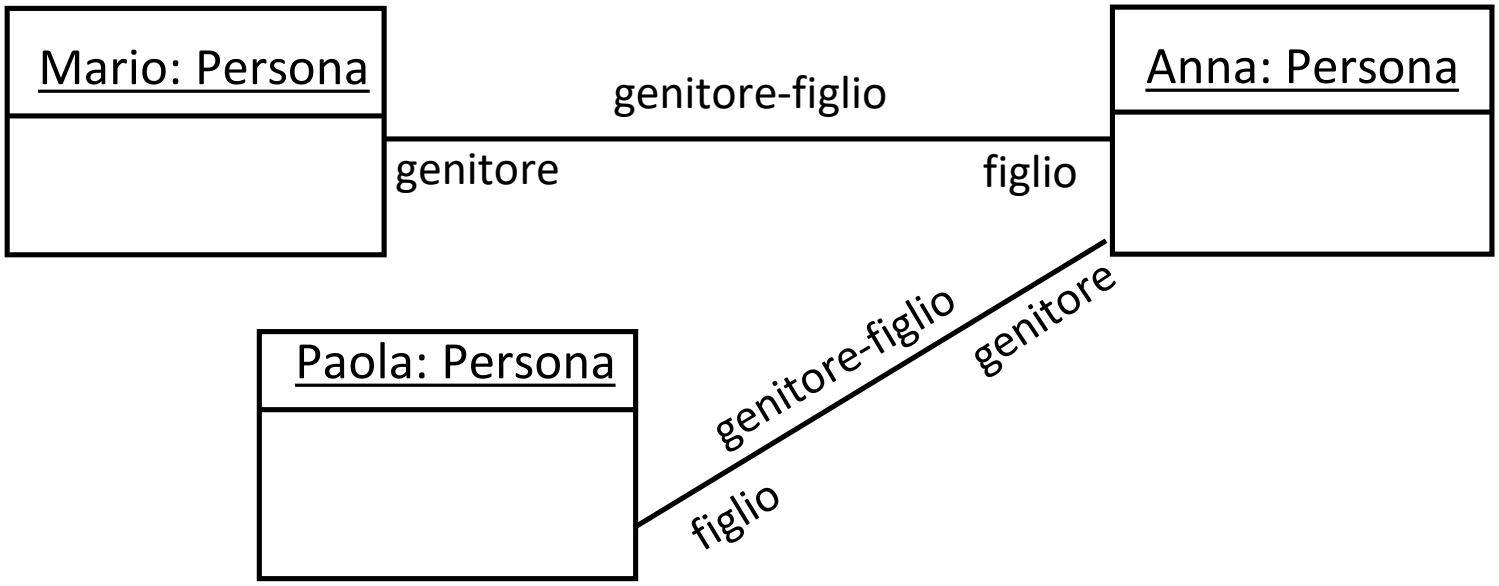
- È possibile aggiungere alla associazione informazione che specificano il ruolo delle classi



- Il ruolo è un nome posizionato lungo la linea dell'associazione, vicino alla classe a cui si riferisce
- Nell'esempio, dipendente è il ruolo che la persona gioca nell'associazione "lavora-in" con Azienda

Ruoli e Link

- I ruoli sono spesso necessari nei link:

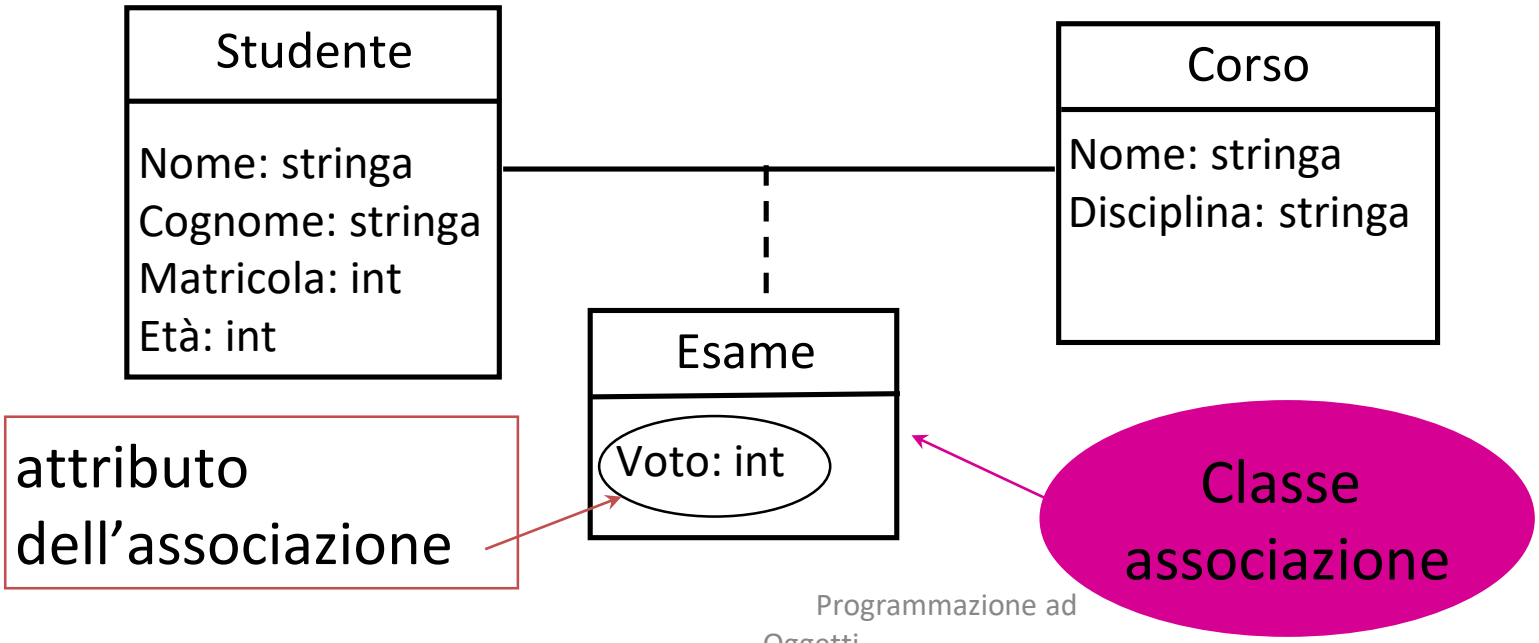


- Se non fossero indicati i ruoli nell'associazione "genitore-figlio", non sapremmo interpretare correttamente i link che sono istanze dell'associazione

Attributi di associazione

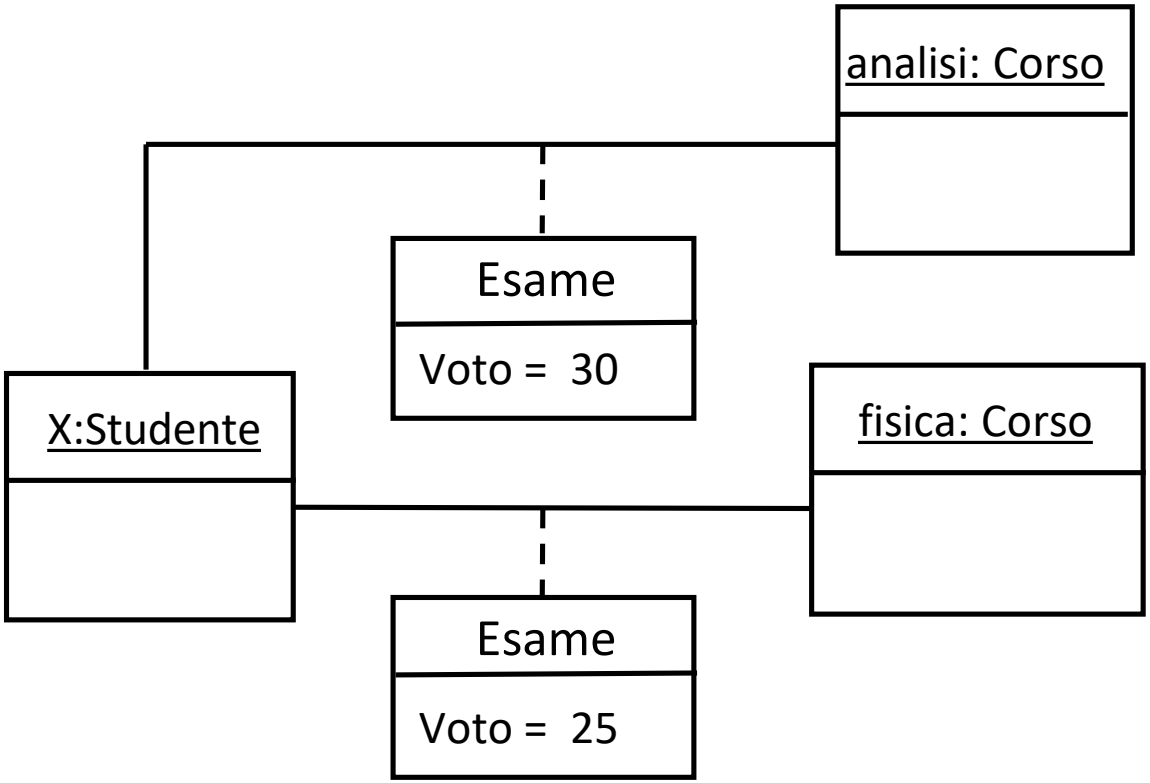
Le associazioni possono avere attributi che associano ad ogni link che è istanza dell'associazione un valore di un determinato tipo.

Esempio: Voto non è una proprietà nè di Studente, nè di Corso, ma della associazione Esame tra Studente e Corso.



Attributi nei link

Ovviamente, se una associazione ha un attributo, ogni link che è istanza dell'associazione avrà un valore per quell'attributo



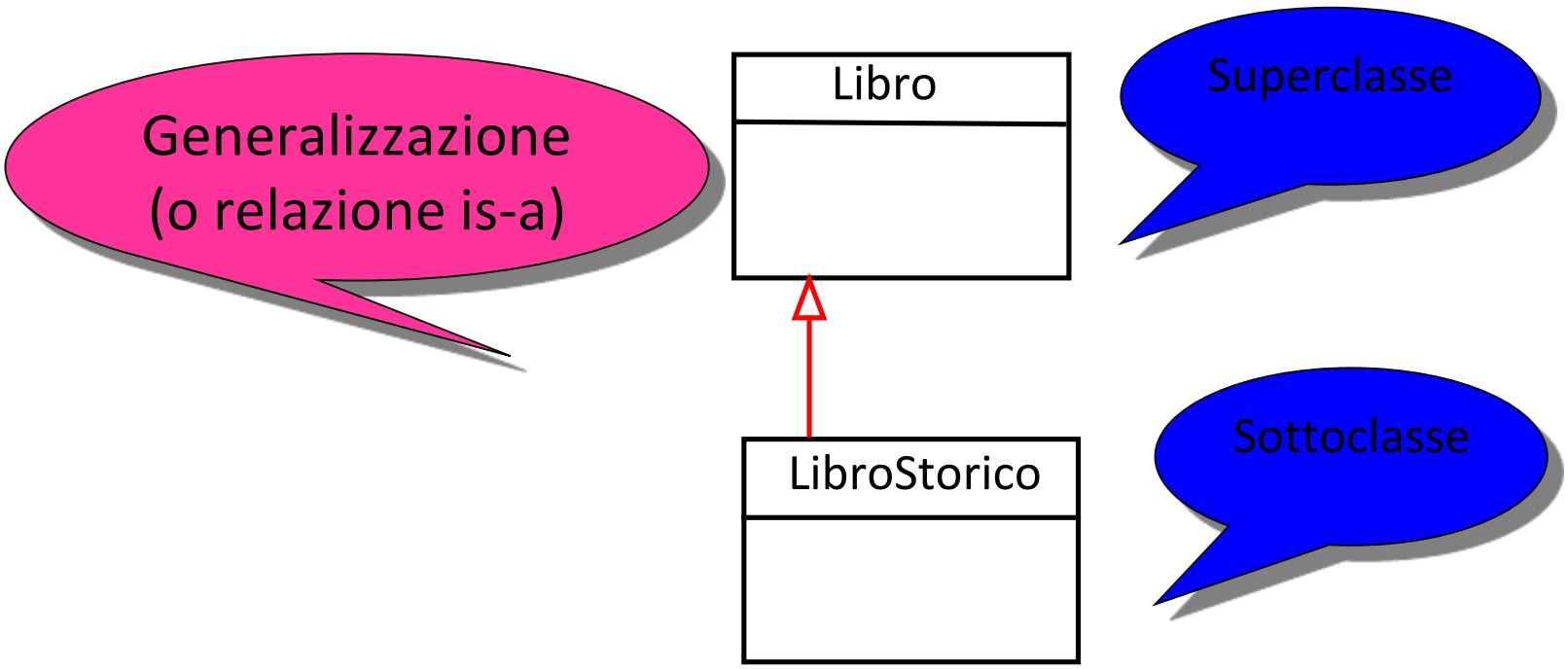
Esercizio 3

Tracciare il diagramma delle classi corrispondenti alle seguenti specifiche:

Si vogliono modellare le persone (con nome, cognome, età), le città di nascita (con nome, numero di abitanti, e sindaco, compresa l'indicazione dell'anno di elezione), e le province in cui si trovano le città (con nome, anno di istituzione, e presidente, con l'anno di elezione e lista politica in cui è stato eletto).

Generalizzazione in UML

- La generalizzazione coinvolge una superclasse ed una o più sottoclassi (classi derivate): ogni istanza di ciascuna sottoclasse è anche istanza della superclasse
- Quando la sottoclasse è una, la generalizzazione è una relazione is-a tra la sottoclasse e la superclasse



Ereditarietà in UML

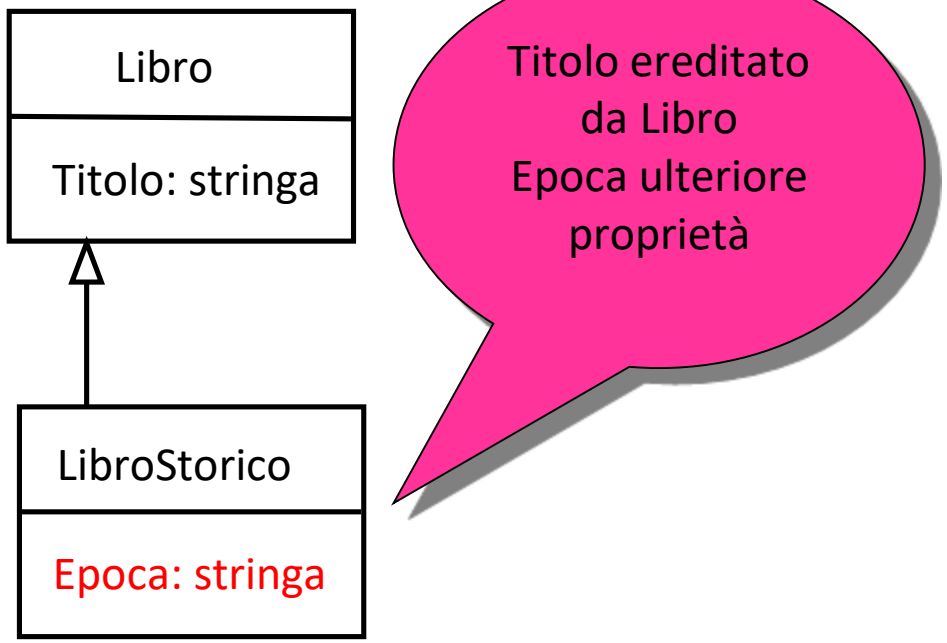
Principio di ereditarietà: ogni proprietà della superclasse è anche una proprietà della sottoclasse, e non si riporta esplicitamente nel diagramma

Dal fatto che

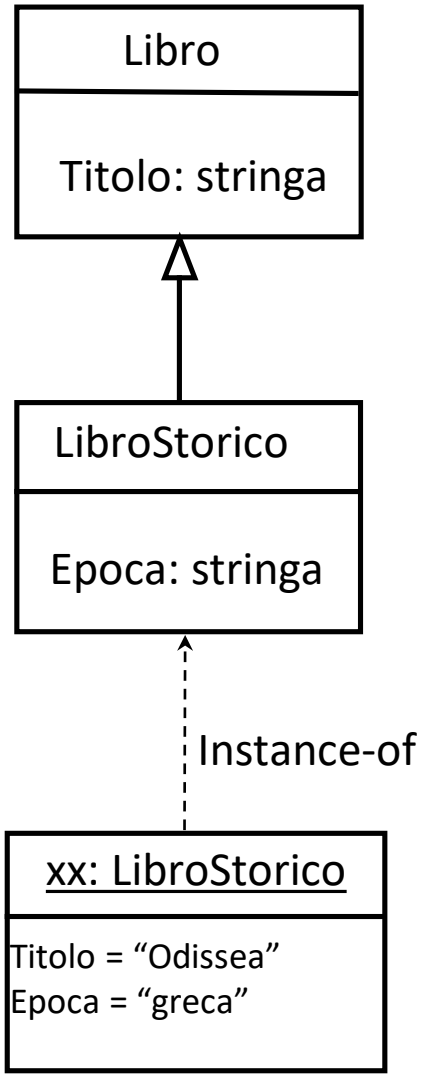
1. Ogni istanza di Libro ha una Titolo
2. Ogni istanza di LibroStorico è una istanza di Libro

segue logicamente che

3. Ogni istanza di LibroStorico ha un Titolo

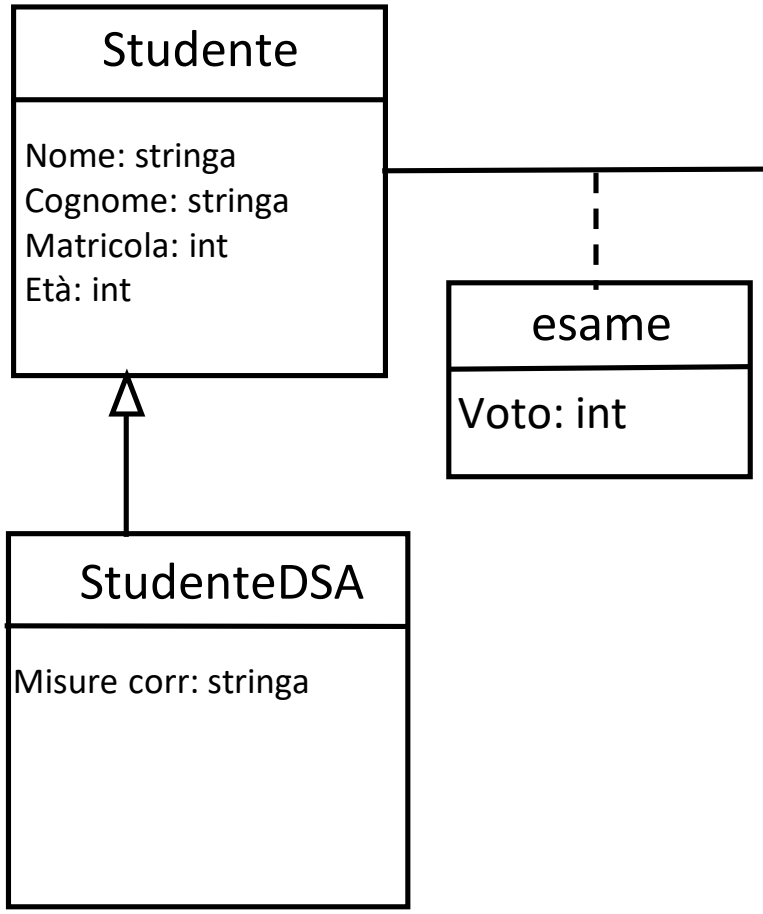


Ereditarietà in UML: istanze



- xx è istanza sia di LibroStorico (classe più specifica) sia di Libro
- due classi diverse non sono necessariamente disgiunte: Libro e LibroStorico non sono disgiunte
- xx ha un valore per tutti gli attributi di LibroStorico, sia quelli propri, sia quelli ereditati dalla classe Libro

Ereditarietà sulle associazioni



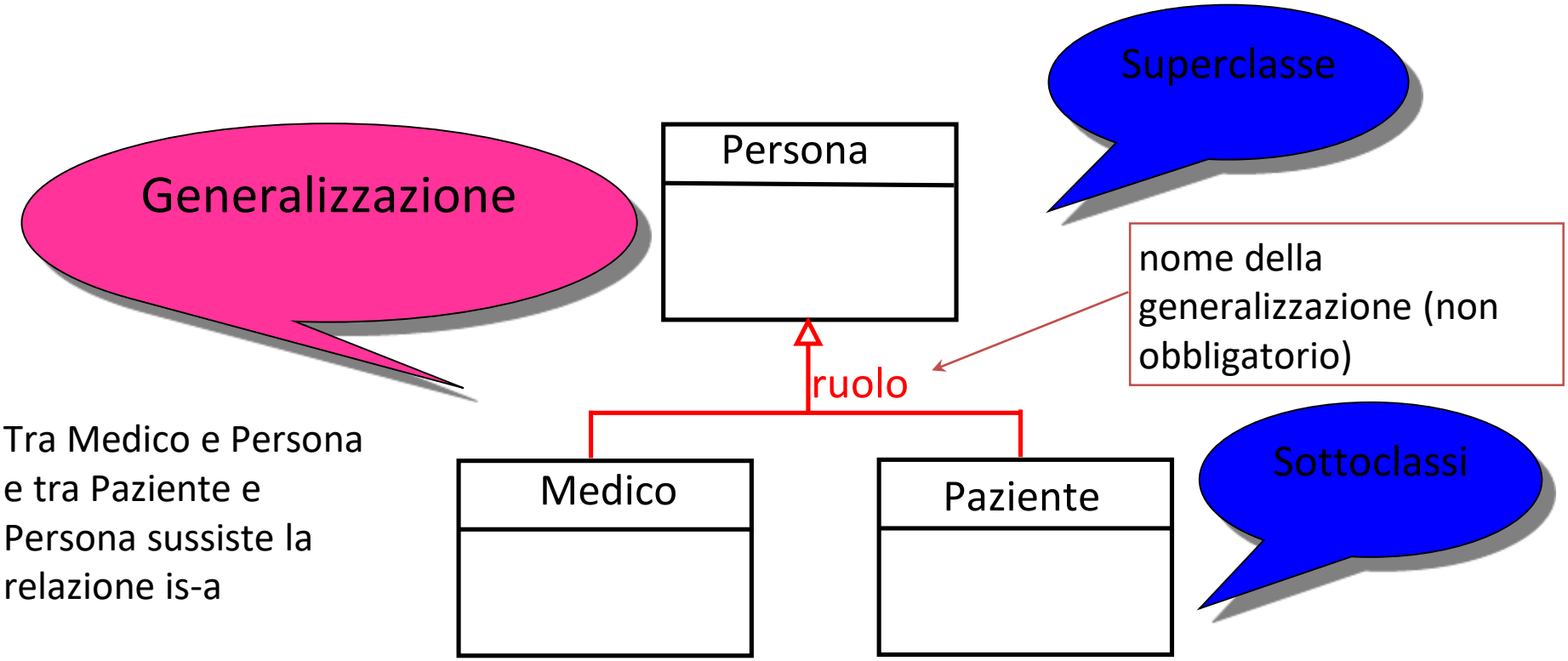
1. Ogni istanza di Studente può essere coinvolta in un numero qualunque di istanze della associazione “esame”
2. Ogni istanza di StudenteDSA è una istanza di Studente

quindi

3. Ogni istanza di StudenteDSA può essere coinvolta in un numero qualunque di istanze della associazione “esame”

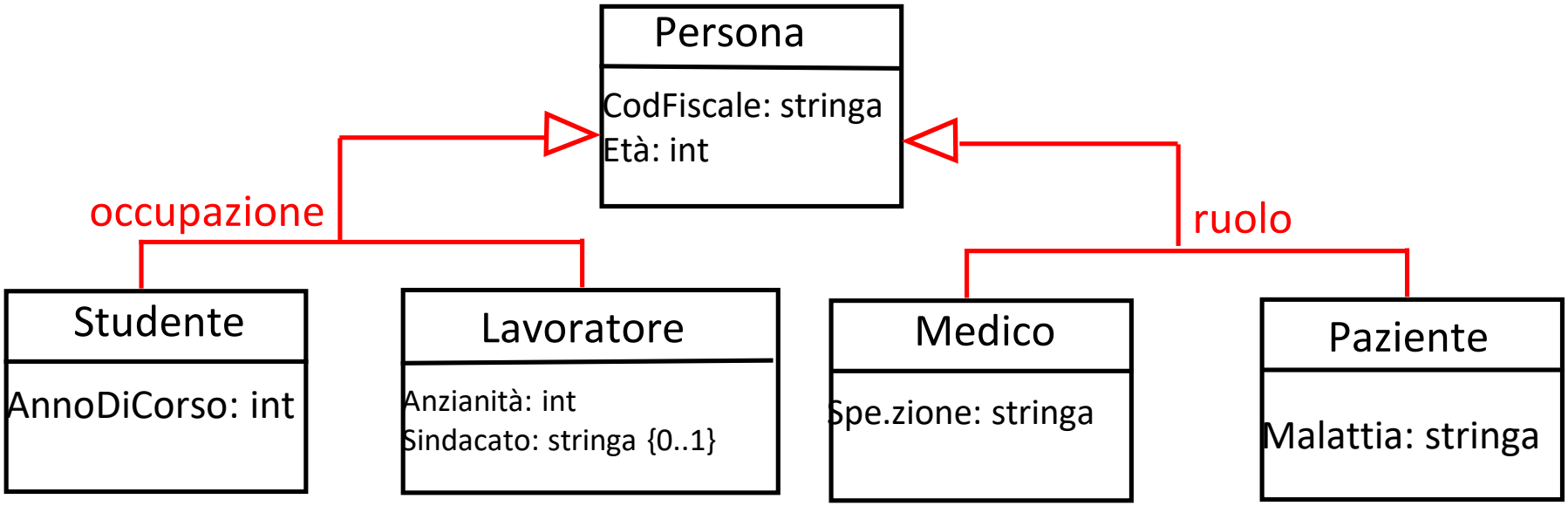
Generalizzazione in UML

La superclasse può anche generalizzare diverse sottoclassi rispetto ad un unico criterio (indicato con un nome)



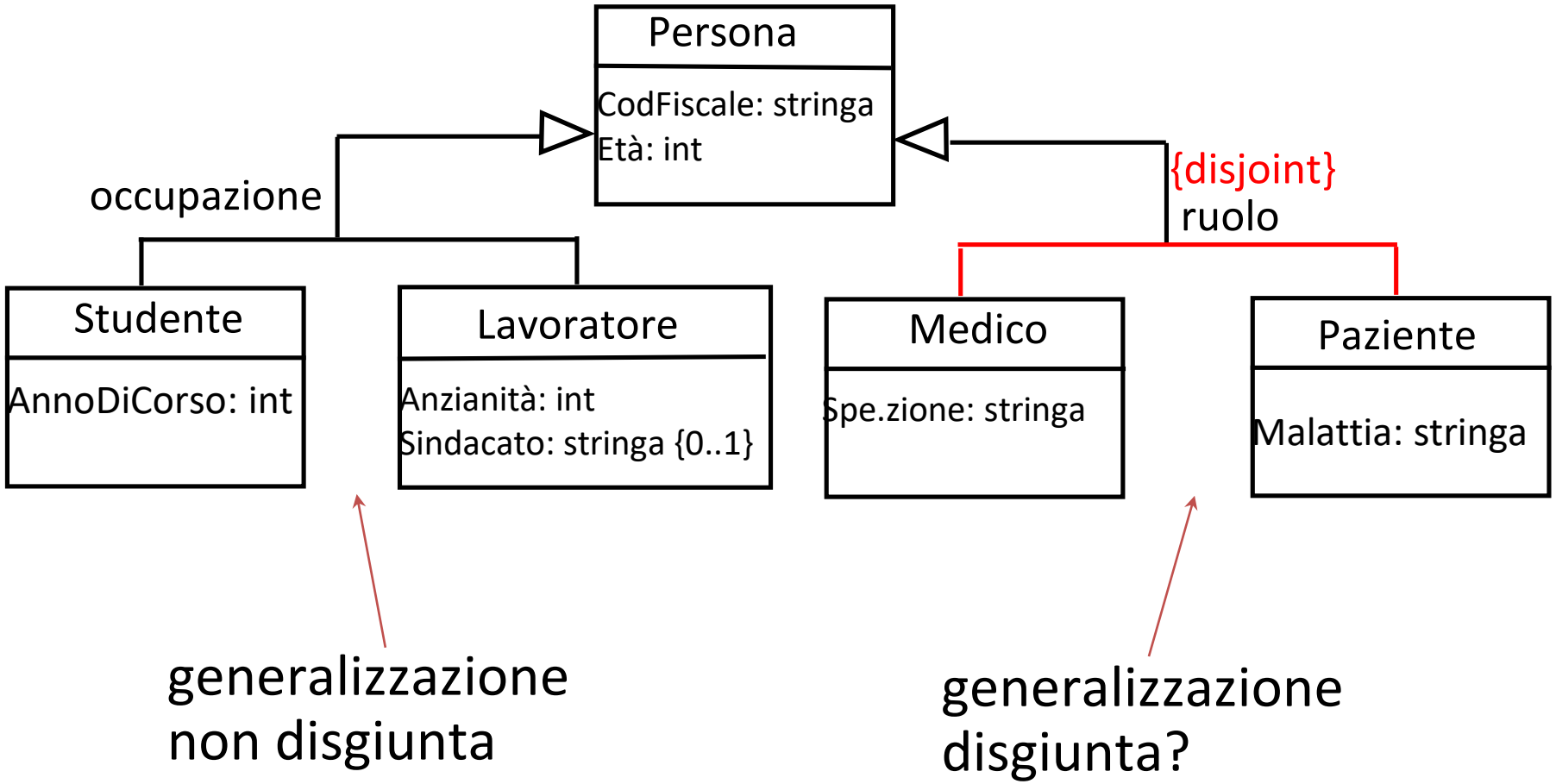
Diverse generalizzazioni della stessa classe

La stessa superclasse può avere diverse generalizzazioni indipendenti, corrispondenti a diversi criteri.



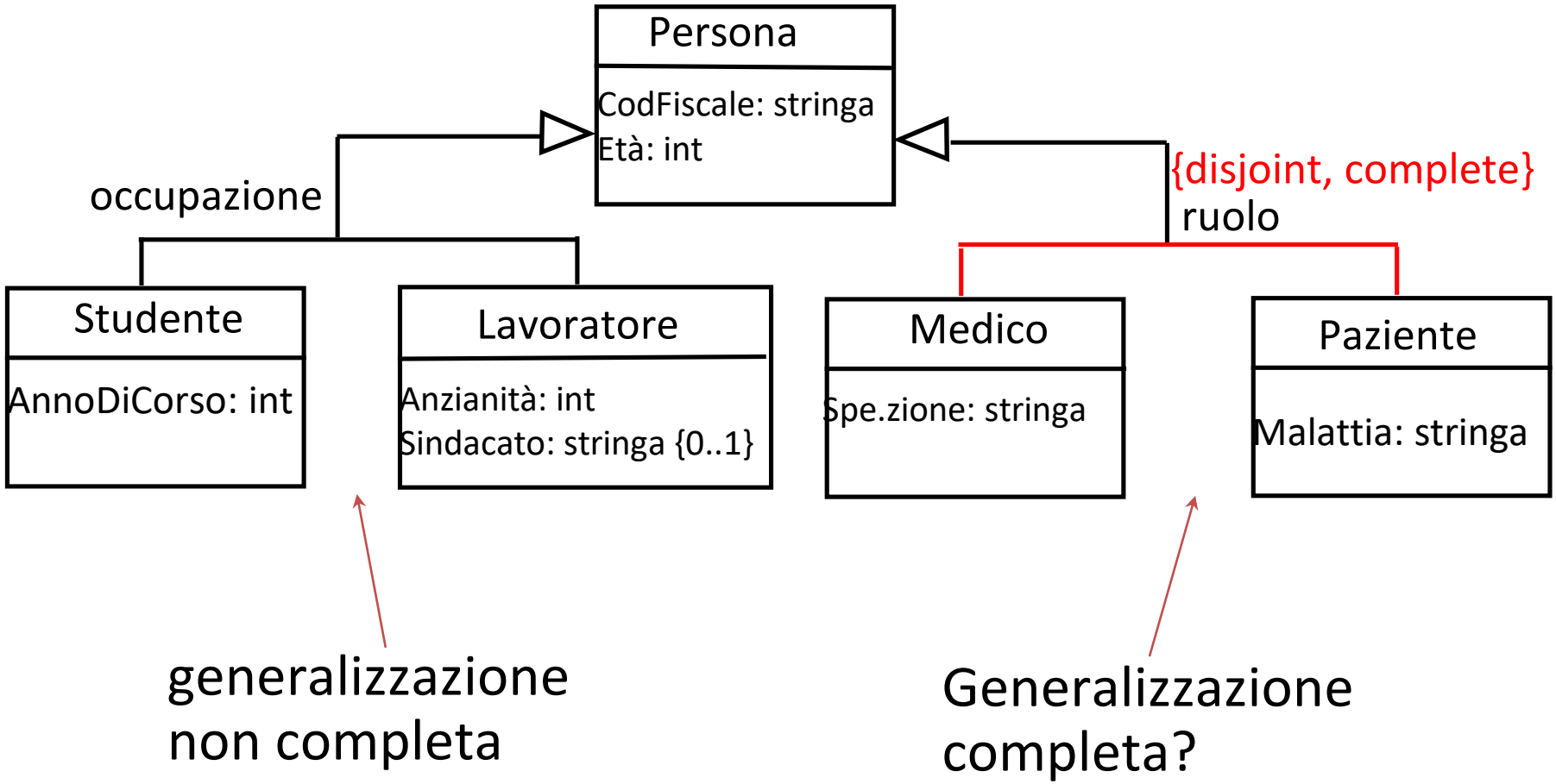
Generalizzazioni disgiunte

Una generalizzazione può essere disgiunta (le sottoclassi sono disgiunte a coppie) o no



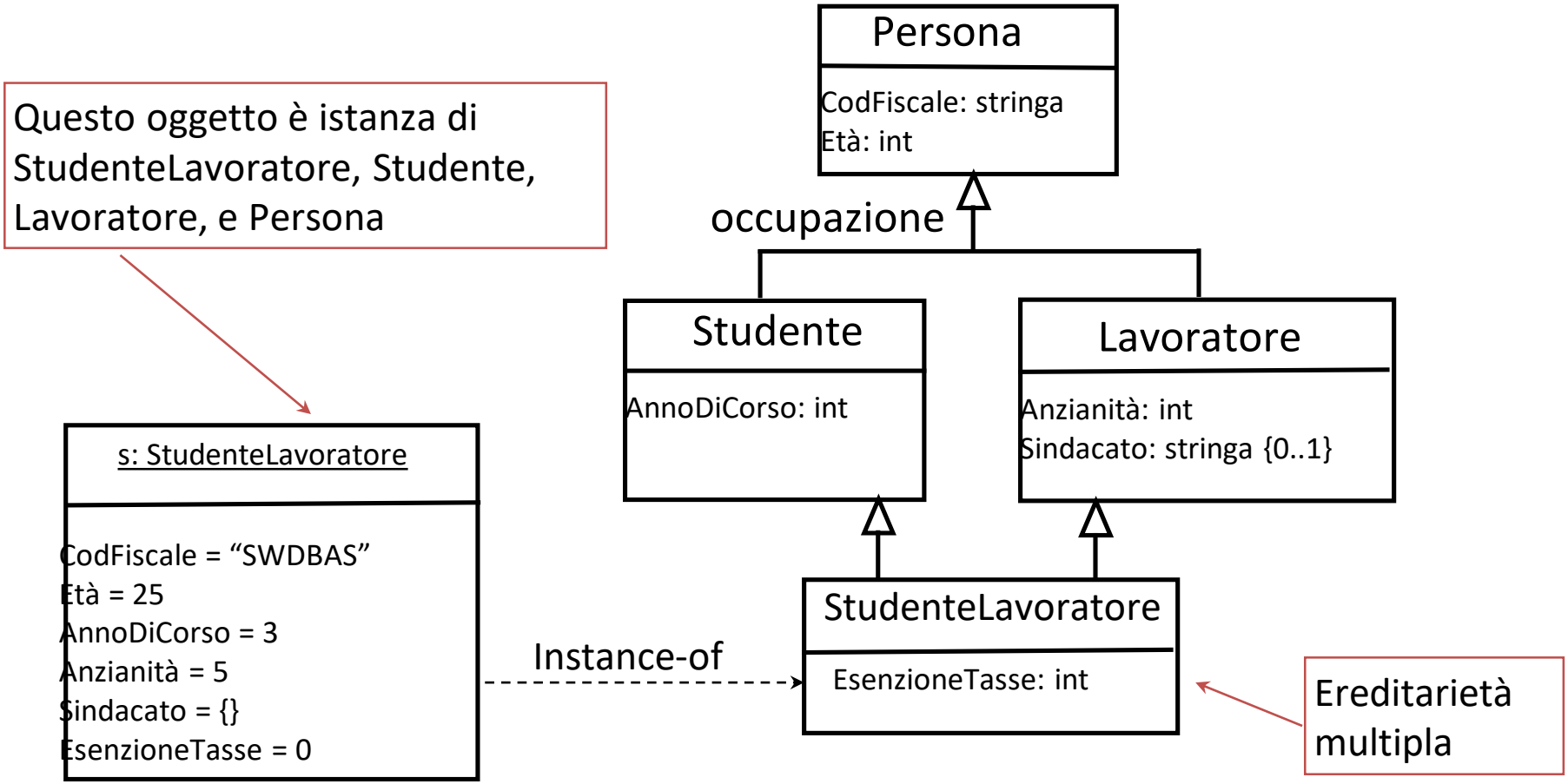
Generalizzazioni complete

Una generalizzazione può essere completa (l'unione delle istanze delle sottoclassi è uguale all'insieme delle istanze della superclasse) o no



Ereditarietà multipla

Per far sì che un oggetto sia istanza di una sola classe più specifica, due sottoclassi non disgiunte possono avere istanze comuni solo se hanno esplicitamente una sottoclasse comune (ereditarietà multipla)



Specializzazione

In una generalizzazione la sottoclasse non solo può avere proprietà aggiuntive rispetto alla superclasse, ma può anche specializzare le proprietà ereditate dalla superclasse.

- Specializzazione di un attributo: Gli attributi della sottoclasse prendono un sottinsieme dei valori degli attributi della superclasse.