

Università di Bologna – Dipartimento di Informatica – Scienza e Ingegneria
Corso di Ingegneria del Software
Anno Accademico 2025-2026
Preappello – 24 novembre 2024 – Durata 45'

1. L'ereditarietà virtuale in C++

- A. È alla base del meccanismo di late binding per la chiamata a funzione.
- B. Non esiste.
- ☒ C. Permette di evitare replica di strutture dati in presenza di ereditarietà multipla con classi base condivise.
- D. Gestisce la reflection con opportune primitive.

2. Si scriva un decorator in Python corretto che stampa un messaggio prima della chiamata di una funzione e alla sua terminazione

3. Il seguente frammento di codice in C++ e in Java ha un comportamento sostanzialmente differente, discutere quale (si ignorino le piccole differenze sintattiche):

```
class A {  
    int x;  
    class B {  
        int y;  
    }  
}
```

4. In UML uno use case

- A. Struttura l'interfaccia grafica del sistema
- ☒ B. Definisce una funzionalità atomica necessaria per l'utente finale
- C. Organizza i casi di test del sistema
- D. Struttura i pacchetti di classi da fornire al cliente

5. Le metaclassi in Java

- ☒ A. Non esistono
- B. Sono alla base del polimorfismo parametrico
- C. Servono nella dependency injection
- D. Permettono l'overloading

6. Si descriva design pattern clone e si dettagli la sua struttura in un linguaggio di programmazione a scelta

Per le domande 6. – 10. si consideri la seguente storia:

Una volta, a Bologna, fecero un palazzo di gelato proprio sulla Piazza Maggiore, e i bambini venivano di lontano a dargli una leccatina. Il tetto era di panna montata, il fumo dei comignoli di zucchero filato, i comignoli di frutta candita. Tutto il resto era di gelato: le porte di gelato, i muri di gelato, i mobili di gelato. Un bambino piccolissimo si era attaccato a un tavolo e gli leccò le zampe una per una, fin che il tavolo gli crollò addosso con tutti i piatti, e i piatti erano di gelato al cioccolato, il più buono. Una guardia del Comune, a un certo punto, si accorse che una finestra si scioglieva. I vetri erano di gelato alla fragola, e si squagliavano in rivoletti rosa. - Presto, - gridò la guardia, - più presto ancora! E giù tutti a leccare più presto, per non lasciar andare perduta una sola goccia di quel capolavoro. - Una poltrona! - implorava una vecchietta, che non riusciva a farsi largo tra la folla, - una poltrona per una povera vecchia. Chi me la porta? Coi braccioli, se è possibile. Un generoso pompiere corse a prenderle una poltrona di gelato alla crema e pistacchio, e la povera vecchietta, tutta beata, cominciò a leccarla proprio dai braccioli. Fu un gran giorno, quello, e per ordine dei dottori nessuno ebbe il mal di pancia. Ancora adesso, (quando i bambini chiedono un altro gelato, i genitori sospirano: - Eh già, per te ce ne vorrebbe un palazzo intero), come quello di Bologna.

6. Si generi il diagramma a classi della storia nel suo complesso estraendo oggetti
7. Si estraggano i metodi
8. Si determinino almeno due storie atomiche estratte dalla storia
9. Si presenti un diagramma di attività desumibile da una parte qualunque della storia

N.B. : fatta eccezione per l'ex.3, che ho sbagliato all'esame e rifatto a casa, il resto è corretto e approvato dal prof. Succi, nonostante io stesso veda, a posteriori, diverse incongruenze nel mio compito.

In generale mi sembra abbia dato priorità alla forma e ai concetti più che ai dettagli (e.g. nel codice "... " come completamento, o qualche svarione nelle classi), quindi questo non è il compito più accurato ma ha valutato a pieni punti egualmente (sono distribuiti uniformemente, 3x10pt).

Per qualsiasi cosa, Telegram: @sandrolebon

2

```
1 import ...
2 def stampa (func):
3     @functool.wrap(func)
4     def wrapper(*args, **kwargs):
5         print("Inizio funzione")
6         res = func(*args, **kwargs)
7         print("Fine funzione")
8         return res
9     return wrapper
```

3

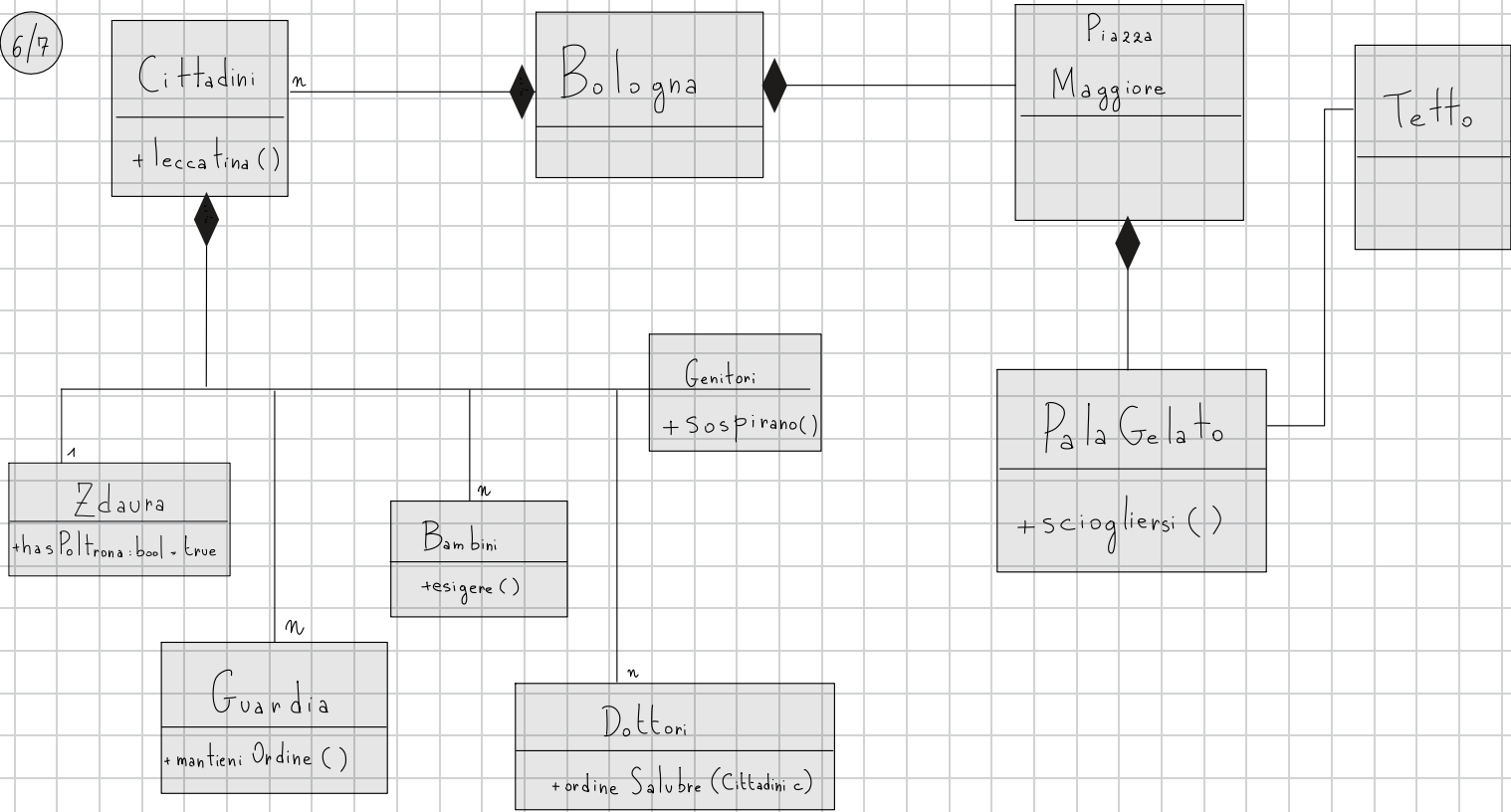
In Java, "class B" è una Inner Class : possiede un riferimento implicito all'istanza di A e può accedere direttamente alla variabile "x".
In C++, "class B" è solo una classe definita nel namespace di A (simile ad una "static class" in Java) : non ha alcun legame con l'istanza di A, quindi non può accedere a "x" a meno che non le venga passato esplicitamente un oggetto di tipo A.

6

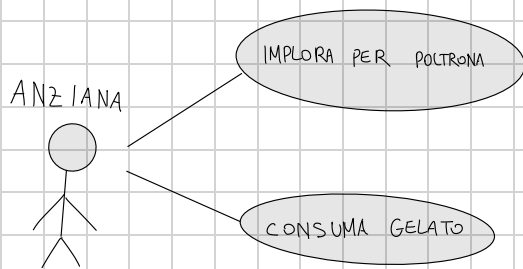
Grazie al design pattern clone è possibile copiare un oggetto anziché crearlo da zero, con notevoli vantaggi.

```
1 import copy
2 class Test:
3     def __init__(self, test): ...
4     def clone(self)
5     return copy.deepcopy(self)
6
7 Originale = Test("test", ...)
8 Clonato = originale.clone()
```

6/7



8



9

