

# Esempi di domande di ingegneria del software

Paolo Ciancarini, Simone Giannone, Giancarlo Succi

Revisione del 14 settembre 2025

## Indice

<b>1</b>	<b>Introduzione</b>	<b>3</b>
<b>2</b>	<b>Domande chiuse sul processo</b>	<b>4</b>
2.1	Domande . . . . .	4
2.2	Risposte corrette . . . . .	10
<b>3</b>	<b>Domande chiuse sul prodotto</b>	<b>13</b>
3.1	Domande . . . . .	13
3.2	Risposte Corrette . . . . .	26
<b>4</b>	<b>Domande aperte sul prodotto</b>	<b>33</b>
<b>5</b>	<b>Domande aperte sul modellamento data una storia</b>	<b>34</b>
5.1	Esempio di domande . . . . .	34
5.2	Esempio di storie . . . . .	34
5.2.1	Esempio di una storia: da “Il funerale della volpe” di Gianni Rodari . . . . .	34
5.2.2	Esempio di una storia: da “Distrazione interplanetaria” di Gianni Rodari . . . . .	35
5.2.3	Esempio di una storia: da “Il Paese con la “S” davanti” di Italo Calvino . . . . .	35
5.2.4	Esempio di una storia: da “Gli amori difficili” di Italo Calvino . . . . .	36

## 1 Introduzione

Il presente documento contiene una serie di domande che possono capitare nei test di Ingegneria del Software nel Corso di Laurea in Informatica dell'Alma Mater – Università di Bologna, tipicamente per gli studenti del primo appello della prima sessione. Questo insieme deve servire come strumento di esercitazione e fa riferimento a programmi che sono variati nel tempo e quindi non è assolutamente rappresentativo del contenuto di un eventuale test (che potrebbe anche non esserci) nell'anno accademico in cui il lettore sfoglia queste pagine.

## 2 Domande chiuse sul processo

### 2.1 Domande

- Ps.1. Nel metodo MoSCoW utilizzato per la gestione dei requisiti, cosa rappresenta la categoria “Should Have”?
- A. Requisiti importanti ma non critici, che possono essere posticipati se necessario.
  - B. Funzionalità non essenziali ma che migliorano l’esperienza utente.
  - C. Funzionalità opzionali che possono essere implementate se c’è tempo.
  - D. Requisiti essenziali che devono essere soddisfatti per il successo del progetto.
- Ps.2. Il “triangolo di ferro” del project management rappresenta i tre vincoli fondamentali di un progetto. Quali sono?
- A. Qualità, Comunicazione, Budget.
  - B. Tempo, Costo, Scopo.
  - C. Team, Product Owner, Scrum Master.
  - D. Customer, Solution, Endeavour.
- Ps.3. Cos’è lo “scope” di un progetto agile?
- A. Gli stakeholder del progetto.
  - B. Il product backlog.
  - C. Il numero finale degli sprint.
  - D. L’architettura del prodotto.
- Ps.4. Quale ruolo è assente in Scrum?
- A. Product owner.
  - B. Scrum master.
  - C. Project manager.
  - D. Team member.
- Ps.5. Cos’è il Minimal Viable Product?
- A. Il prodotto con il più alto ritorno sugli investimenti rispetto al rischio.
  - B. Il prodotto al termine del primo sprint
  - C. Il prodotto al termine dell’ultimo sprint
  - D. Il prodotto al termine dello sprint più recente
- Ps.6. Che cos’è la velocità di un team Scrum?
- A. Il numero medio di story point “fatti” in uno sprint.
  - B. Il numero medio di user story “fatti” in uno sprint.

- C. Il numero di story point “fatti” nell’ultimo sprint.
  - D. Il numero di story point “fatti” nello sprint in corso.
- Ps.7. Come si calcola lo sforzo necessario allo sviluppo di un prodotto software?
- A. Il costo in euro di un prodotto software.
  - B. La durata in giorni dello sviluppo.
  - C. La durata dello sviluppo moltiplicata per le persone coinvolte nello sviluppo.
  - D. La somma dei tempi dedicati allo sviluppo da parte di ciascun sviluppatore.
- Ps.8. Come si chiama il testing in cui si usa il sorgente del programma?
- A. Blackbox.
  - B. Whitebox.
  - C. Debugging.
  - D. Validazione.
- Ps.9. Quale delle seguenti affermazioni descrive meglio il compito principale del project manager di un prodotto software?
- A. Assicurarsi che ogni sviluppatore scriva il codice meno costoso possibile nel minor tempo possibile.
  - B. Gestire i requisiti parlando con gli stakeholder e analizzandoli col team di sviluppo.
  - C. Gestire il testing in modo automatico cercando di risparmiare tempo e diminuire effort.
  - D. Assicurarsi che il prodotto sia completato nei tempi previsti, rimanendo entro il budget e soddisfacendo i requisiti richiesti.
- Ps.10. Quale delle seguenti caratteristiche identifica meglio un processo di sviluppo agile?
- A. La suddivisione del lavoro in iterazioni brevi e incrementali con feedback frequenti dagli stakeholder.
  - B. L’uso delle user story per definire completamente i requisiti di prodotto prima di iniziare lo sviluppo.
  - C. La pianificazione accurata di tutte le attività necessarie al prodotto prima dell’avvio dello sviluppo.
  - D. L’individuazione di un singolo responsabile che prenda tutte le decisioni tecniche e di progetto.
- Ps.11. In un design orientato agli oggetti, cos’è un "componente riusabile"?
- A. Una parte del codice progettata per essere facilmente integrabile in contesti diversi.

- B. Un template HTML collegato a un controller.
  - C. Una classe che eredita da se stessa.
  - D. Un modulo Python contenente solo costanti.
- Ps.12. Quale tra questi è un vantaggio dei test automatici?
- A. Aumentano il coupling tra i moduli.
  - B. Rallentano il ciclo di sviluppo.
  - C. Eliminano la necessità del debugging.
  - D. Permettono di individuare rapidamente regressioni nel codice.
- Ps.13. Qual è l'obiettivo della tecnica del "Mocking" nei test?
- A. Creare ambienti virtuali.
  - B. Misurare la copertura del codice.
  - C. Simulare comportamenti di componenti esterni.
  - D. Gestire errori in tempo reale.
- Ps.14. Cosa caratterizza un buon design modulare?
- A. Moduli altamente accoppiati tra loro.
  - B. Codice duplicato nei vari moduli.
  - C. Assenza di interfacce.
  - D. Alta coesione e basso accoppiamento.
- Ps.15. A cosa serve un diagramma delle classi UML?
- A. A visualizzare l'output di un programma.
  - B. A rappresentare le relazioni tra classi e oggetti.
  - C. A disegnare wireframe per interfacce grafiche.
  - D. A testare il comportamento delle classi.
- Ps.16. Qual è lo scopo principale della progettazione guidata dai test (TDD)?
- A. Produrre test in ritardo rispetto allo sviluppo.
  - B. Guidare la scrittura del codice attraverso i test.
  - C. Scrivere meno codice possibile.
  - D. Testare l'usabilità dell'interfaccia utente.
- Ps.17. Qual è lo scopo principale della fase di analisi dei requisiti?
- A. Definire l'architettura del sistema.
  - B. Comprendere cosa il sistema deve fare.
  - C. Scrivere i test di unità.
  - D. Stimare il tempo di esecuzione del software.
- Ps.18. Quale dei seguenti modelli di sviluppo è caratterizzato da fasi sequenziali e ben definite?

- A. Modello Incrementale.
  - B. Modello a spirale.
  - C. Modello a cascata.
  - D. Modello Agile.
- Ps.19. Perché è utile creare una documentazione dei requisiti?
- A. Per decidere il linguaggio di programmazione.
  - B. Per generare automaticamente il codice sorgente.
  - C. Per evitare l'uso di librerie esterne.
  - D. Per avere una base condivisa tra sviluppatori e stakeholder.
- Ps.20. Qual è l'obiettivo del testing nel processo di sviluppo software?
- A. Ridurre la dimensione del codice.
  - B. Eliminare le dipendenze tra moduli.
  - C. Produrre documentazione automatica.
  - D. Verificare che il software si comporti come atteso.
- Ps.21. In Scrum, cosa rappresenta il Product Backlog?
- A. L'elenco dei bug da correggere.
  - B. L'elenco ordinato delle funzionalità e modifiche richieste al prodotto.
  - C. Il diario delle attività svolte dallo Scrum Master.
  - D. Il piano di progetto iniziale.
- Ps.22. Chi è responsabile della massimizzazione del valore del prodotto in Scrum?
- A. Il team di sviluppo.
  - B. Il project manager.
  - C. Il Product Owner.
  - D. Il cliente.
- Ps.23. Quale delle seguenti attività avviene tipicamente alla fine di uno Sprint in Scrum?
- A. Stima dei costi del progetto.
  - B. Ridefinizione della visione del prodotto.
  - C. Sprint Review e Sprint Retrospective.
  - D. Disegno dell'architettura del sistema.
- Ps.24. In cosa consiste la Daily Scrum?
- A. Una revisione dettagliata del codice.
  - B. Un incontro con gli stakeholder.
  - C. Una breve riunione quotidiana per pianificare il lavoro delle prossime 24 ore.

- D. Una presentazione dei risultati ottenuti al cliente.
- Ps.25. Quale delle seguenti non è una delle responsabilità dello Scrum Master?
- A. Decidere quali funzionalità implementare.
  - B. Facilitare le cerimonie Scrum.
  - C. Aiutare a rimuovere gli impedimenti al lavoro del team.
  - D. Promuovere la comprensione di Scrum nel team.
- Ps.26. Qual è la durata raccomandata per uno Sprint in Scrum?
- A. 1 giorno.
  - B. 1 mese e mezzo.
  - C. Massimo 1 mese.
  - D. Almeno 3 mesi.
- Ps.27. Qual è uno dei principali vantaggi dell'uso di Scrum?
- A. Riduzione automatica del debito tecnico.
  - B. Feedback frequente e continuo da parte degli stakeholder.
  - C. Rimozione del bisogno di documentazione.
  - D. Codice generato automaticamente ad ogni sprint.
- Ps.28. Qual è l'obiettivo della Retrospective in Scrum?
- A. Pianificare le funzionalità dello sprint successivo.
  - B. Identificare miglioramenti nel processo di lavoro del team.
  - C. Presentare il lavoro finito agli stakeholder.
  - D. Raccogliere i bug segnalati dagli utenti.
- Ps.29. In Scrum, chi partecipa alla Sprint Planning?
- A. Solo lo Scrum Master e il Product Owner.
  - B. Solo il team di sviluppo.
  - C. Solo il cliente.
  - D. Scrum Master, Product Owner e team di sviluppo.
- Ps.30. Quale delle seguenti è una caratteristica di un buon requisito software?
- A. Essere vago e aperto a interpretazioni.
  - B. Essere chiaro, misurabile e verificabile.
  - C. Essere scritto in linguaggio tecnico.
  - D. Essere modificabile solo dal team di sviluppo.
- Ps.31. Cosa si intende per 'prototipazione' nel contesto dello sviluppo software?
- A. Scrivere codice direttamente in produzione.
  - B. Definire il piano economico del progetto.



- C. Costruire versioni semplificate del software per raccogliere feedback.
  - D. Creare la documentazione finale del sistema.
- Ps.32. Cosa rappresenta una 'user story' in Scrum?
- A. Un diagramma dei flussi di dati.
  - B. Una descrizione sintetica di una funzionalità dal punto di vista dell'utente.
  - C. Un requisito funzionale formale.
  - D. Un caso di test per l'integrazione.
- Ps.33. Quale delle seguenti attività rientra nella gestione della qualità del software?
- A. Assegnare i task al team di sviluppo.
  - B. Stimare i costi delle licenze software.
  - C. Definire metriche per valutare il codice.
  - D. Scrivere la documentazione utente.
- Ps.34. Qual è un vantaggio dell'uso della modellazione UML nel processo di sviluppo?
- A. Elimina la necessità del testing.
  - B. Facilita la comunicazione tra membri del team e stakeholder.
  - C. Riduce i tempi di compilazione.
  - D. Permette di testare automaticamente il codice.
- Ps.35. La relazione include per un use case
- A. definisce un'estensione atomica del mio use case.
  - B. definisce un use case che è una specializzazione atomica dello use case di partenza.
  - C. definisce una funzionalità coesa dello use case e quindi separata.
  - D. definisce una relazione di ereditarietà tra use case.
- Ps.36. La relazione extend per un use case
- A. definisce un'estensione atomica del mio use case.
  - B. definisce un use case che è una specializzazione atomica dello use case di partenza.
  - C. crea con lo use case di partenza un nuovo use case.
  - D. definisce una relazione di ereditarietà tra use case.
- Ps.37. Un use case è
- A. un qualunque requisito.
  - B. un caso di uso dell'interfaccia utente accettato dall'utente.
  - C. un caso di uso atomico che dà valore all'utente.
  - D. un requisito semplice del sistema validato dall'utente.

## 2.2 Risposte corrette

- Ps.1. Nel metodo MoSCoW utilizzato per la gestione dei requisiti, cosa rappresenta la categoria "Should Have"?
- ✓ A. Requisiti importanti ma non critici, che possono essere posticipati se necessario.
- Ps.2. Il "triangolo di ferro" del project management rappresenta i tre vincoli fondamentali di un progetto. Quali sono?
- ✓ B. Tempo, Costo, Scopo.
- Ps.3. Cos'è lo "scope" di un progetto agile?
- ✓ B. Il product backlog.
- Ps.4. Quale ruolo è assente in Scrum?
- ✓ C. Project manager.
- Ps.5. Cos'è il Minimal Viable Product?
- ✓ A. Il prodotto con il più alto ritorno sugli investimenti rispetto al rischio.
- Ps.6. Che cos'è la velocità di un team Scrum?
- ✓ A. Il numero medio di story point "fatti" in uno sprint.
- Ps.7. Come si calcola lo sforzo necessario allo sviluppo di un prodotto software?
- ✓ C. La durata dello sviluppo moltiplicata per le persone coinvolte nello sviluppo.
- Ps.8. Come si chiama il testing in cui si usa il sorgente del programma?
- ✓ B. Whitebox.
- Ps.9. Quale delle seguenti affermazioni descrive meglio il compito principale del project manager di un prodotto software?
- ✓ D. Assicurarsi che il prodotto sia completato nei tempi previsti, rimanendo entro il budget e soddisfacendo i requisiti richiesti.
- Ps.10. Quale delle seguenti caratteristiche identifica meglio un processo di sviluppo agile?
- ✓ A. La suddivisione del lavoro in iterazioni brevi e incrementali con feedback frequenti dagli stakeholder.
- Ps.11. In un design orientato agli oggetti, cos'è un "componente riusabile"?
- ✓ A. Una parte del codice progettata per essere facilmente integrabile in contesti diversi.

- Ps.12. Quale tra questi è un vantaggio dei test automatici?
- ✓D. Permettono di individuare rapidamente regressioni nel codice.
- Ps.13. Qual è l'obiettivo della tecnica del "Mocking" nei test?
- ✓C. Simulare comportamenti di componenti esterni.
- Ps.14. Cosa caratterizza un buon design modulare?
- ✓D. Alta coesione e basso accoppiamento.
- Ps.15. A cosa serve un diagramma delle classi UML?
- ✓B. A rappresentare le relazioni tra classi e oggetti.
- Ps.16. Qual è lo scopo principale della progettazione guidata dai test (TDD)?
- ✓B. Guidare la scrittura del codice attraverso i test.
- Ps.17. Qual è lo scopo principale della fase di analisi dei requisiti?
- ✓B. Comprendere cosa il sistema deve fare.
- Ps.18. Quale dei seguenti modelli di sviluppo è caratterizzato da fasi sequenziali e ben definite?
- ✓C. Modello a cascata.
- Ps.19. Perché è utile creare una documentazione dei requisiti?
- ✓D. Per avere una base condivisa tra sviluppatori e stakeholder.
- Ps.20. Qual è l'obiettivo del testing nel processo di sviluppo software?
- ✓D. Verificare che il software si comporti come atteso.
- Ps.21. In Scrum, cosa rappresenta il Product Backlog?
- ✓B. L'elenco ordinato delle funzionalità e modifiche richieste al prodotto.
- Ps.22. Chi è responsabile della massimizzazione del valore del prodotto in Scrum?
- ✓C. Il Product Owner.
- Ps.23. Quale delle seguenti attività avviene tipicamente alla fine di uno Sprint in Scrum?
- ✓C. Sprint Review e Sprint Retrospective.
- Ps.24. In cosa consiste la Daily Scrum?
- ✓C. Una breve riunione quotidiana per pianificare il lavoro delle prossime 24 ore.

- Ps.25. Quale delle seguenti non è una delle responsabilità dello Scrum Master?
- ✓A. Decidere quali funzionalità implementare.
- Ps.26. Qual è la durata raccomandata per uno Sprint in Scrum?
- ✓C. Massimo 1 mese.
- Ps.27. Qual è uno dei principali vantaggi dell'uso di Scrum?
- ✓B. Feedback frequente e continuo da parte degli stakeholder.
- Ps.28. Qual è l'obiettivo della Retrospective in Scrum?
- ✓B. Identificare miglioramenti nel processo di lavoro del team.
- Ps.29. In Scrum, chi partecipa alla Sprint Planning?
- ✓D. Scrum Master, Product Owner e team di sviluppo.
- Ps.30. Quale delle seguenti è una caratteristica di un buon requisito software?
- ✓B. Essere chiaro, misurabile e verificabile.
- Ps.31. Cosa si intende per 'prototipazione' nel contesto dello sviluppo software?
- ✓C. Costruire versioni semplificate del software per raccogliere feedback.
- Ps.32. Cosa rappresenta una 'user story' in Scrum?
- ✓B. Una descrizione sintetica di una funzionalità dal punto di vista dell'utente.
- Ps.33. Quale delle seguenti attività rientra nella gestione della qualità del software?
- ✓C. Definire metriche per valutare il codice.
- Ps.34. Qual è un vantaggio dell'uso della modellazione UML nel processo di sviluppo?
- ✓B. Facilita la comunicazione tra membri del team e stakeholder.
- Ps.35. La relazione include per un use case
- ✓C. definisce una funzionalità coesa dello use case e quindi separata.
- Ps.36. La relazione extend per un use case
- ✓A. definisce un'estensione atomica del mio use case.
- Ps.37. Un use case è
- ✓C. un caso di uso atomico che dà valore all'utente.

## 3 Domande chiuse sul prodotto

### 3.1 Domande

Pd.1. Il decorator in Python

- A. implementa nel linguaggio il design pattern del decorator.
- B. permette la decorazione del codice con commenti necessari alla auto-documentazione.
- C. è una funzione di ordine superiore.
- D. definisce l'interfaccia con il C++.

Pd.2. I template in C++

- A. sono una modalità corretta e completa di definire generici.
- B. simulano la presenza di generici tramite la definizione di puntatori a void.
- C. definiscono classi astratte che permettono di definire comportamenti analoghi ai generici.
- D. producono una moltiplicazione di codice oggetto se istanziati su tipi multipli.

Pd.3. Qual è lo scopo principale di un caso d'uso in un processo di sviluppo software?

- A. Descrivere come il sistema è implementato internamente.
- B. Fornire una rappresentazione grafica della struttura dei dati.
- C. Rappresentare le interazioni tra gli attori e il sistema per soddisfare un obiettivo specifico.
- D. Valutare la performance del sistema in ambiente di produzione.

Pd.4. Quale affermazione descrive meglio la specializzazione dei template in C++?

- A. I template sono modelli di codice precompilati da usare in fase di inclusione di header.
- B. I template in C++ funzionano esattamente come i generici in Java, con type erasure a runtime.
- C. I template emulano il polimorfismo parametrico generando classi o funzioni specifiche durante la compilazione, sulla base di modelli esistenti.
- D. I template usano la specializzazione per creare gerarchie di classi simili all'ereditarietà in fase di esecuzione.

Pd.5. Una funzione virtuale in UML

- A. può limitare il numero di repliche di campi in una classe base.

- B. permette l'overriding.
  - C. permette l'overloading.
  - D. è una forma di eredità che sembra esistere ma poi si dissolve.
- Pd.6. Con quale strumento in C si può approssimare il polimorfismo parametrico?
- A. Con le macro.
  - B. Con l'inclusione.
  - C. Con i puntatori a funzione.
  - D. Non è mai possibile.
- Pd.7. L'ereditarietà virtuale
- A. è la tipica forma di ereditarietà che implementa il late binding.
  - B. è una forma di ereditarietà che non permette di istanziare oggetti della classe.
  - C. è una forma di ereditarietà che serve ad evitare la moltiplicazione di istanze della classe.
  - D. non esiste.
- Pd.8. Un oggetto è
- A. un modello di una classe.
  - B. un'istanza di una classe.
  - C. una classe.
  - D. una metaclassa.
- Pd.9. UML è l'acronimo per
- A. Universal Modeling Language.
  - B. Unified Modeling Language.
  - C. Universal Modeling Level.
  - D. Unified Modeling Level.
- Pd.10. Qual è la principale differenza tra requisiti funzionali e requisiti non funzionali nel contesto dello sviluppo software?
- A. I requisiti funzionali descrivono come il software deve essere implementato, mentre i requisiti non funzionali descrivono cosa il software deve fare.
  - B. I requisiti funzionali sono sempre obbligatori, mentre i requisiti non funzionali sono opzionali.
  - C. I requisiti funzionali definiscono i comportamenti e le operazioni del sistema, mentre i requisiti non funzionali specificano vincoli o caratteristiche di qualità del sistema.

- D. I requisiti funzionali riguardano solo l'interfaccia utente, mentre i requisiti non funzionali riguardano solo il backend del sistema.

Pd.11. Il late binding permette

- A. la definizione di modelli generici.
- B. la definizioni di più funzioni con nomi identici ma parametri diversi risolta durante la compilazione.
- C. la definizioni di più funzioni con nomi identici ma parametri diversi risolta durante l'esecuzione.
- D. l'associazione del corpo di una funzione al suo nome durante l'esecuzione.

Pd.12. L'overloading in Python

- A. dà un errore di compilazione.
- B. è strutturato come in C++.
- C. richiede una fase di preprocessing.
- D. è gestito tramite un decorator.

Pd.13. Qual è uno dei principi base di Python secondo il Zen of Python?

- A. La velocità è tutto.
- B. Complicato è meglio di complesso.
- C. Esplicito è meglio che implicito.
- D. Più codice, meglio è.

Pd.14. Quale tra questi NON è un paradigma supportato da Python?

- A. Scripting.
- B. Logico.
- C. Imperativo.
- D. Funzionale.

Pd.15. Cosa succede se dimentichi l'indentazione dopo un'istruzione if in Python?

- A. Viene ignorata l'istruzione.
- B. Si esegue comunque.
- C. Si genera un SyntaxWarning.
- D. Si genera un IndentationError.

Pd.16. Come viene gestito il typing in Python?

- A. Statico e forte.
- B. Dinamico e debole.
- C. Statico e debole.
- D. Dinamico e forte.

- Pd.17. Qual è il risultato dell'operazione `9//4` in Python?
- A. 2
  - B. 2.25
  - C. 3
  - D. Errore.
- Pd.18. Quale tra queste affermazioni è corretta riguardo le stringhe in Python?
- A. Le stringhe sono mutevoli.
  - B. Si può modificare un singolo carattere.
  - C. Le stringhe sono immutabili.
  - D. Le stringhe supportano operatori di assegnamento.
- Pd.19. Cosa succede se si tenta di accedere all'indice 20 di una stringa lunga 20 caratteri?
- A. Restituisce il carattere finale.
  - B. Solleva un `IndexError`.
  - C. Restituisce `None`.
  - D. Solleva un `ValueError`.
- Pd.20. Quale tra questi tipi di collezioni è immutabile in Python?
- A. `Tuple`.
  - B. `Set`.
  - C. `Dictionary`.
  - D. `List`.
- Pd.21. Quale funzione viene usata per generare una sequenza di numeri interi in Python?
- A. `list()`
  - B. `seq()`
  - C. `range()`
  - D. `series()`
- Pd.22. Qual è la differenza tra `pop()` e `remove()` in una lista?
- A. `pop()` rimuove per valore, `remove()` per indice.
  - B. `pop()` rimuove per indice, `remove()` per valore.
  - C. Entrambe rimuovono per indice.
  - D. Nessuna differenza.
- Pd.23. Cosa ritorna `range(2, 10, 3)` una volta convertito in lista?
- A. `[2, 4, 6, 8, 10]`



- B. [2, 5, 8]
- C. [2, 3, 4, 5, 6, 7, 8, 9]
- D. [2, 5, 7, 9]

Pd.24. Cosa rappresenta l'eccezione `StopIteration` in un iteratore?

- A. L'iteratore ha raggiunto la fine della sequenza.
- B. L'iteratore ha trovato un errore nei dati.
- C. L'iterazione non è mai iniziata.
- D. L'oggetto non è iterabile.

Pd.25. Come si chiama una funzione senza nome in Python?

- A. `anon()`
- B. `ghost`
- C. `lambda`
- D. `inline`

Pd.26. Cosa fa il decoratore `@staticmethod`?

- A. Rende un metodo accessibile solo dalla classe.
- B. Consente di chiamare il metodo senza passare `self` o `cls`.
- C. Rende il metodo protetto.
- D. Aggiunge il metodo al costruttore.

Pd.27. Qual è l'ordine delle regole di scoping in Python (LEGB)?

- A. Local, External, Global, Builtin
- B. Local, Enclosing, Global, Builtin
- C. Last, Enclosing, Global, Builtin
- D. Last, External, Global, Builtin

Pd.28. A cosa serve la parola chiave `global` in Python?

- A. A creare una variabile visibile solo localmente.
- B. A limitare l'uso di variabili globali in funzioni.
- C. A riferirsi a una variabile definita fuori dal blocco corrente.
- D. A esportare variabili in altri moduli.

Pd.29. Cosa succede ai parametri con valori predefiniti mutabili in Python?

- A. Vengono copiati ogni volta.
- B. Mantengono il loro stato tra chiamate successive.
- C. Generano un errore se modificati.
- D. Vengono convertiti in valori immutabili.

Pd.30. Cosa rappresenta `type(x)` quando `x` è una funzione?

- A. method
  - B. builtin\_function\_or\_method
  - C. function
  - D. lambda
- Pd.31. Dopo aver consumato un generatore in Python, cosa succede?
- A. Può essere riutilizzato infinite volte.
  - B. Viene automaticamente rigenerato.
  - C. Viene convertito in lista.
  - D. È esaurito e non può più essere usato.
- Pd.32. Quale metodo magico permette di sovraccaricare l'operatore '+'?
- A. `__sum__`
  - B. `__plus__`
  - C. `__add__`
  - D. `__concat__`
- Pd.33. Qual è il vantaggio dell'uso di una metaclassa Singleton generalizzata?
- A. Permette di creare oggetti da qualsiasi classe base.
  - B. Garantisce che esista solo un'istanza di qualsiasi classe che la usa.
  - C. Migliora la velocità di esecuzione.
  - D. Impedisce la modifica degli attributi.
- Pd.34. Cosa rappresenta il concetto di "interfaccia" in ambito di programmazione orientata agli oggetti?
- A. Una libreria esterna che estende le funzionalità.
  - B. Un contratto che specifica un insieme di metodi che una classe deve implementare.
  - C. Una classe concreta astratta.
  - D. Una classe che eredita da più superclassi.
- Pd.35. Qual è lo scopo principale della separazione tra interfaccia e implementazione?
- A. Rendere più difficile la manutenzione del codice.
  - B. Migliorare le performance in compilazione.
  - C. Ridurre l'accoppiamento tra le componenti.
  - D. Garantire la compatibilità con il sistema operativo.
- Pd.36. Quando si usa il principio di Dependency Inversion?
- A. Per implementare ricorsione.
  - B. Per ereditare proprietà da una superclasse.

- C. Quando si devono ottimizzare cicli annidati.
- D. Quando si vogliono dipendenze da astrazioni e non da classi concrete.

Pd.37. Perché è utile utilizzare il principio di Single Responsibility?

- A. Ogni classe ha una sola ragione di cambiamento.
- B. Aumenta la velocità di compilazione.
- C. Riduce la quantità di codice scritto.
- D. Evita l'uso delle classi astratte.

Pd.38. Qual è il risultato dell'operazione `7 // 3` in Python?

- (a) 2.3333
- (b) 3
- (c) 1
- (d) 2

Pd.39. Cosa succede se si prova ad assegnare un nuovo carattere a un indice di una stringa in Python?

- A. La stringa viene modificata correttamente.
- B. Viene sollevato un `TypeError`.
- C. La stringa viene copiata e modificata.
- D. Non succede nulla.

Pd.40. Quale operatore in Python serve a concatenare due stringhe?

- A. `*`
- B. `+`
- C. `&`
- D. `//`

Pd.41. Come si scrive una stringa multilinea in Python?

- A. Usando il carattere `\n`.
- B. Usando triple virgolette `'''...'''` o `"""..."""`.
- C. Usando la funzione `multistring()`.
- D. Con la concatenazione di più stringhe singole.

Pd.42. Cosa produce la slice `[::-1]` applicata a una stringa?

- A. Ritorna la stessa stringa.
- B. Ritorna una copia della stringa.
- C. Ritorna la stringa invertita.
- D. Ritorna la stringa con solo caratteri pari.

Pd.43. Quale errore viene sollevato quando si accede a un indice fuori dal range di una stringa?

- A. `TypeError`
- B. `ValueError`
- C. `IndexError`
- D. `KeyError`

Pd.44. Che differenza c'è tra gli operatori `+` e `*` usati con le stringhe?

- A. `+` ripete la stringa, `*` concatena.
- B. `+` concatena le stringhe, `*` ripete la stringa un certo numero di volte.
- C. Entrambi concatenano.
- D. Entrambi ripetono la stringa.

Pd.45. Cosa succede con questo codice?

```
c = 'Hello' ' ' 'World'
print(c)
```

- A. Errore di sintassi.
- B. Stampa "HelloWorld" senza spazio.
- C. Stampa "Hello World" con spazio.
- D. Nessuna stampa.

Pd.46. Come funziona la slice `[start:end:step]`?

- A. Restituisce una sottostringa da `start` a `end` escluso, saltando `step` caratteri.
- B. Restituisce la stringa invertita.
- C. Restituisce una copia della stringa.
- D. Restituisce una lista di caratteri.

Pd.47. Qual è la differenza principale tra una stringa e una lista in Python?

- A. Le stringhe sono mutabili, le liste sono immutabili.
- B. Le stringhe sono immutabili, le liste sono mutabili.
- C. Nessuna differenza.
- D. Le liste contengono solo numeri, le stringhe solo caratteri.

Pd.48. Quale tra le seguenti affermazioni è vera riguardo le liste in Python?

- A. Le liste sono immutabili e ordinate.
- B. Le liste sono mutabili e ordinate.
- C. Le liste non supportano duplicati.
- D. Le liste sono immutabili e non ordinate.

Pd.49. Cosa succede assegnando una lista ad un'altra variabile in Python?

- A. Viene creata una copia profonda della lista.

- B. Viene creata una copia superficiale della lista.
- C. Entrambe le variabili puntano alla stessa lista.
- D. Viene generato un errore.

Pd.50. Quale metodo permette di aggiungere un elemento alla fine di una lista?

- A. `add()`
- B. `insert()`
- C. `append()`
- D. `extend()`

Pd.51. Cosa succede se si modifica un elemento interno (ad esempio una lista) contenuto in una tupla?

- A. La tupla diventa mutabile.
- B. La modifica genera un errore.
- C. L'elemento interno si modifica ma la tupla resta immutabile.
- D. La tupla si copia automaticamente prima della modifica.

Pd.52. Quale tra le seguenti affermazioni sui set è corretta?

- A. I set sono ordinati e possono contenere duplicati.
- B. I set sono non ordinati e non possono contenere duplicati.
- C. I set possono contenere liste come elementi.
- D. I set supportano l'operatore `+` per concatenazione.

Pd.53. Cosa succede se si prova ad aggiungere una lista come elemento di un set?

- A. La lista viene aggiunta correttamente.
- B. Il set diventa mutabile.
- C. Si genera un errore di tipo "unhashable type".
- D. La lista viene convertita in tupla e aggiunta.

Pd.54. Come si crea una copia profonda (deep copy) di una lista annidata in Python?

- A. Usando il metodo `copy()` della lista.
- B. Usando l'operatore di assegnazione `=`.
- C. Usando il modulo `copy` con `copy.deepcopy()`.
- D. Non è possibile creare copie profonde di liste annidate.

Pd.55. Qual è il risultato di questa operazione su una lista `aList = [1, 2, 3, 4]`: `aList.pop(2)`?

- A. Rimuove e ritorna l'elemento con valore 2.
- B. Rimuove e ritorna l'elemento all'indice 2.
- C. Rimuove l'elemento 2 senza ritornarlo.

- D. Genera un errore.
- Pd.56. Quale tra le seguenti affermazioni descrive correttamente la differenza tra tuple e liste?
- A. Tuple sono mutabili, liste sono immutabili.
  - B. Tuple sono immutabili, liste sono mutabili.
  - C. Entrambe sono mutabili ma con sintassi diversa.
  - D. Entrambe sono immutabili ma con sintassi diversa.
- Pd.57. Cosa succede se si concatena una tupla con un intero in Python?
- A. L'intero viene aggiunto come elemento.
  - B. Viene generato un errore di tipo.
  - C. L'intero viene convertito in tupla automaticamente.
  - D. La tupla diventa mutabile.
- Pd.58. Nel contesto delle API di OpenAI, quale ruolo serve a definire le regole e il comportamento del modello?
- A. `user`
  - B. `assistant`
  - C. `system`
  - D. `developer`
- Pd.59. Quale informazione è contenuta nel campo `usage` della risposta API di OpenAI?
- A. Lo stile della risposta
  - B. Il numero di token utilizzati
  - C. Il livello di temperatura
  - D. Il modello usato
- Pd.60. A cosa serve un `PromptTemplate` in LangChain?
- A. Gestire la memoria conversazionale
  - B. Definire una catena di agenti
  - C. Creare prompt strutturati e parametrizzati
  - D. Collegare database esterni
- Pd.61. Qual è la funzione principale della Memory in LangChain?
- A. Salvare lo stato del modello
  - B. Ricordare il contesto delle conversazioni precedenti
  - C. Generare automaticamente prompt
  - D. Aumentare la velocità di calcolo

- Pd.62. In CrewAI, quale elemento rappresenta una singola unità di lavoro assegnata a un agente?
- A. Crew
  - B. Role
  - C. Task
  - D. Goal
- Pd.63. Qual è la differenza principale tra LangChain e CrewAI?
- A. LangChain si concentra su pipeline e strumenti, CrewAI sulla coordinazione multi-agente
  - B. LangChain è solo per traduzioni, CrewAI per generazione di testo
  - C. LangChain è un modello di linguaggio, CrewAI è un framework
  - D. LangChain funziona senza API, CrewAI solo con API OpenAI
- Pd.64. Cosa rappresenta il campo `finish_reason` in una risposta API di OpenAI?
- A. Il modello è andato in errore
  - B. Il motivo per cui il modello ha terminato la generazione
  - C. La quantità di token generati
  - D. La temperatura usata nella generazione
- Pd.65. Per quale motivo può essere utile lo streaming delle risposte di un LLM?
- A. Per ridurre il costo dei token
  - B. Per migliorare la sicurezza dei prompt
  - C. Per ridurre la latenza e visualizzare il testo man mano che viene generato
  - D. Per salvare automaticamente le risposte nel database
- Pd.66. In LangChain, cosa fa una `LLMChain`?
- A. Collega un prompt a un modello e gestisce input e output
  - B. Genera automaticamente nuove API
  - C. Ottimizza l'uso dei token
  - D. Traduce i messaggi in diverse lingue
- Pd.67. Qual è il vantaggio principale di usare Agents in LangChain?
- A. Consentono di ridurre i costi computazionali
  - B. Eliminano la necessità di definire prompt
  - C. Consentono al modello di scegliere dinamicamente quali strumenti utilizzare
  - D. Rendono l'LLM più veloce

- Pd.68. Qual è lo scopo dei Tools in LangChain?
- A. Esporre funzioni esterne che il modello può invocare
  - B. Memorizzare conversazioni precedenti
  - C. Fornire dataset per il training
  - D. Modificare la temperatura del modello
- Pd.69. In CrewAI, cosa definisce il parametro `goal` di un agente?
- A. Le API a cui l'agente può accedere
  - B. L'obiettivo esplicito che guida le decisioni dell'agente
  - C. Il tipo di memoria utilizzata
  - D. La sequenza dei task da eseguire
- Pd.70. In CrewAI, come possono essere eseguiti i task?
- A. In modo sequenziale o in parallelo
  - B. Solo in ordine gerarchico
  - C. Solo uno alla volta
  - D. Sempre in background
- Pd.71. Cosa rappresenta un Crew in CrewAI?
- A. Un singolo agente specializzato
  - B. Una memoria condivisa tra più task
  - C. Un insieme di agenti e task coordinati
  - D. Un tool esterno integrato nel sistema
- Pd.72. Quali sono le tre fasi principali dell'esecuzione di un programma Python?
- A. Parsing, Ottimizzazione, Linking
  - B. Interpreter Initialization, Compilazione in bytecode, Bytecode Interpretation
  - C. Preprocessing, Compilazione in Assembly, Esecuzione
  - D. Traduzione in C, Compilazione, Runtime
- Pd.73. Qual è il compito principale della fase di Interpreter Initialization?
- A. Generare i file `.pyc`
  - B. Caricare moduli built-in, gestire memoria e creare il GIL
  - C. Disassemblare il bytecode
  - D. Avviare la compilazione JIT
- Pd.74. Cosa rappresenta un frame durante l'esecuzione del bytecode?
- A. Una struttura che contiene solo variabili globali
  - B. Un'unità di esecuzione con codice, scope e stack



- C. Una copia del sorgente `.py`
- D. Un dizionario temporaneo con i built-in

Pd.75. Come funziona la call stack in Python?

- A. Ogni funzione aggiunge un frame; al ritorno il frame viene rimosso
- B. Tutte le funzioni condividono un unico frame
- C. La call stack non esiste, si usano solo dizionari di contesto
- D. Le funzioni ricorsive sovrascrivono i frame precedenti

Pd.76. Qual è la funzione del modulo `dis` in Python?

- A. Disassemblare il bytecode in opcodes leggibili
- B. Ottimizzare la compilazione
- C. Mostrare i moduli built-in
- D. Gestire i riferimenti degli oggetti

Pd.77. Quale meccanismo di gestione memoria usa CPython?

- A. Garbage collection generazionale puro
- B. Reference counting con garbage collector per cicli
- C. Manuale come in C
- D. Unicamente mark-and-sweep

Pd.78. Perché esiste il Global Interpreter Lock (GIL)?

- A. Per velocizzare il calcolo parallelo
- B. Per semplificare la gestione memoria, evitando race condition
- C. Per supportare librerie C esterne
- D. Per ridurre la latenza del bytecode

Pd.79. Come fanno librerie come NumPy a ottenere parallelismo nonostante il GIL?

- A. Usano Python puro senza bytecode
- B. Rilasciano temporaneamente il GIL in codice C esterno
- C. Compilano direttamente in Assembly
- D. Ignorano completamente la call stack

Pd.80. Qual è la caratteristica principale della Python Virtual Machine?

- A. È un interprete stack-based che esegue bytecode
- B. È un compilatore che genera file eseguibili nativi
- C. È una macchina a registri per CPU multi-core
- D. Funziona solo come ottimizzatore di memoria

Pd.81. Qual è lo scopo del value stack nella VM?

- A. Conservare moduli built-in
- B. Salvare costanti globali
- C. Tenere operandi e risultati temporanei durante l'esecuzione
- D. Gestire i thread del GIL

### 3.2 Risposte Corrette

Ps.1. Il decorator in Python

- ✓C. è una funzione di ordine superiore.

Ps.2. I template in C++

- ✓D. producono una moltiplicazione di codice oggetto se istanziati su tipi multipli.

Ps.3. Qual è lo scopo principale di un caso d'uso in un processo di sviluppo software?

- ✓C. Rappresentare le interazioni tra gli attori e il sistema per soddisfare un obiettivo specifico.

Ps.4. Quale affermazione descrive meglio la specializzazione dei template in C++?

- ✓C. I template emulano il polimorfismo parametrico generando classi o funzioni specifiche durante la compilazione, sulla base di modelli esistenti.

Ps.5. Una funzione virtuale in UML

- ✓B. permette l'overriding.

Ps.6. Con quale strumento in C si può approssimare il polimorfismo parametrico?

- ✓A. Con le macro.

Ps.7. L'ereditarietà virtuale

- ✓C. è una forma di ereditarietà che serve ad evitare la moltiplicazione di istanze della classe.

Ps.8. Un oggetto è

- ✓B. un'istanza di una classe.

Ps.9. UML è l'acronimo per

- ✓B. Unified Modeling Language.

- Ps.10. Qual è la principale differenza tra requisiti funzionali e requisiti non funzionali nel contesto dello sviluppo software?
- ✓C. I requisiti funzionali definiscono i comportamenti e le operazioni del sistema, mentre i requisiti non funzionali specificano vincoli o caratteristiche di qualità del sistema.
- Ps.11. Il late binding permette
- ✓D. l'associazione del corpo di una funzione al suo nome durante l'esecuzione.
- Ps.12. L'overloading in Python
- ✓D. è gestito tramite un decorator.
- Ps.13. Qual è uno dei principi base di Python secondo il Zen of Python?
- ✓C. Esplicito è meglio che implicito.
- Ps.14. Quale tra questi NON è un paradigma supportato da Python?
- ✓B. Logico.
- Ps.15. Cosa succede se dimentichi l'indentazione dopo un'istruzione if in Python?
- ✓D. Si genera un IndentationError.
- Ps.16. Come viene gestito il typing in Python?
- ✓D. Dinamico e forte.
- Ps.17. Qual è il risultato dell'operazione  $9//4$  in Python?
- ✓A. 2
- Ps.18. Quale tra queste affermazioni è corretta riguardo le stringhe in Python?
- ✓C. Le stringhe sono immutabili.
- Ps.19. Cosa succede se si tenta di accedere all'indice 20 di una stringa lunga 20 caratteri?
- ✓B. Solleva un IndexError.
- Ps.20. Quale tra questi tipi di collezioni è immutabile in Python?
- ✓A. Tuple.
- Ps.21. Quale funzione viene usata per generare una sequenza di numeri interi in Python?
- ✓C. range()

- Ps.22. Qual è la differenza tra `pop()` e `remove()` in una lista?
- ✓B. `pop()` rimuove per indice, `remove()` per valore.
- Ps.23. Cosa ritorna `range(2, 10, 3)` una volta convertito in lista?
- ✓B. `[2, 5, 8]`
- Ps.24. Cosa rappresenta l'eccezione `StopIteration` in un iteratore?
- ✓A. L'iteratore ha raggiunto la fine della sequenza.
- Ps.25. Come si chiama una funzione senza nome in Python?
- ✓C. `lambda`
- Ps.26. Cosa fa il decoratore `@staticmethod`?
- ✓B. Consente di chiamare il metodo senza passare `self` o `cls`.
- Ps.27. Qual è l'ordine delle regole di scoping in Python (LEGB)?
- ✓B. Local, Enclosing, Global, Builtin
- Ps.28. A cosa serve la parola chiave `global` in Python?
- ✓C. A riferirsi a una variabile definita fuori dal blocco corrente.
- Ps.29. Cosa succede ai parametri con valori predefiniti mutabili in Python?
- ✓B. Mantengono il loro stato tra chiamate successive.
- Ps.30. Cosa rappresenta `type(x)` quando `x` è una funzione?
- ✓C. `function`
- Ps.31. Dopo aver consumato un generatore in Python, cosa succede?
- ✓D. È esaurito e non può più essere usato.
- Ps.32. Quale metodo magico permette di sovraccaricare l'operatore `'+'`?
- ✓C. `__add__`
- Ps.33. Qual è il vantaggio dell'uso di una metaclassa Singleton generalizzata?
- ✓B. Garantisce che esista solo un'istanza di qualsiasi classe che la usa.
- Ps.34. Cosa rappresenta il concetto di "interfaccia" in ambito di programmazione orientata agli oggetti?
- ✓B. Un contratto che specifica un insieme di metodi che una classe deve implementare.

Ps.35. Qual è lo scopo principale della separazione tra interfaccia e implementazione?

✓C. Ridurre l'accoppiamento tra le componenti.

Ps.36. Quando si usa il principio di Dependency Inversion?

✓D. Quando si vogliono dipendenze da astrazioni e non da classi concrete.

Ps.37. Perché è utile utilizzare il principio di Single Responsibility?

✓A. Ogni classe ha una sola ragione di cambiamento.

Ps.38. Qual è il risultato dell'operazione `7 // 3` in Python?

✓D. 2

Ps.39. Cosa succede se si prova ad assegnare un nuovo carattere a un indice di una stringa in Python?

✓B. Viene sollevato un `TypeError`.

Ps.40. Quale operatore in Python serve a concatenare due stringhe?

✓B. +

Ps.41. Come si scrive una stringa multilinea in Python?

✓B. Usando triple virgolette `'''...'''` o `"""..."""`.

Ps.42. Cosa produce la slice `[::-1]` applicata a una stringa?

✓C. Ritorna la stringa invertita.

Ps.43. Quale errore viene sollevato quando si accede a un indice fuori dal range di una stringa?

✓C. `IndexError`

Ps.44. Che differenza c'è tra gli operatori + e \* usati con le stringhe?

✓B. + concatena le stringhe, \* ripete la stringa un certo numero di volte.

Ps.45. Cosa succede con questo codice?

```
c = 'Hello' ' ' 'World'
print(c)
```

✓C. Stampa "Hello World" con spazio.

Ps.46. Come funziona la slice `[start:end:step]`?

✓A. Restituisce una sottostringa da `start` a `end` escluso, saltando `step` caratteri.

- Ps.47. Qual è la differenza principale tra una stringa e una lista in Python?
- ✓B. Le stringhe sono immutabili, le liste sono mutabili.
- Ps.48. Quale tra le seguenti affermazioni è vera riguardo le liste in Python?
- ✓B. Le liste sono mutabili e ordinate.
- Ps.49. Cosa succede assegnando una lista ad un'altra variabile in Python?
- ✓C. Entrambe le variabili puntano alla stessa lista.
- Ps.50. Quale metodo permette di aggiungere un elemento alla fine di una lista?
- ✓C. `append()`
- Ps.51. Cosa succede se si modifica un elemento interno (ad esempio una lista) contenuto in una tupla?
- ✓C. L'elemento interno si modifica ma la tupla resta immutabile.
- Ps.52. Quale tra le seguenti affermazioni sui set è corretta?
- ✓B. I set sono non ordinati e non possono contenere duplicati.
- Ps.53. Cosa succede se si prova ad aggiungere una lista come elemento di un set?
- ✓C. Si genera un errore di tipo "unhashable type".
- Ps.54. Come si crea una copia profonda (deep copy) di una lista annidata in Python?
- ✓C. Usando il modulo `copy` con `copy.deepcopy()`.
- Ps.55. Qual è il risultato di questa operazione su una lista `aList = [1, 2, 3, 4]`: `aList.pop(2)`?
- ✓B. Rimuove e ritorna l'elemento all'indice 2.
- Ps.56. Quale tra le seguenti affermazioni descrive correttamente la differenza tra tuple e liste?
- ✓B. Tuple sono immutabili, liste sono mutabili.
- Ps.57. Cosa succede se si concatena una tupla con un intero in Python?
- ✓B. Viene generato un errore di tipo.
- Ps.58. Nel contesto delle API di OpenAI, quale ruolo serve a definire le regole e il comportamento del modello?
- ✓C. `system`

- Ps.59. Quale informazione è contenuta nel campo **usage** della risposta API di OpenAI?
- ✓B. Il numero di token utilizzati
- Ps.60. A cosa serve un **PromptTemplate** in LangChain?
- ✓C. Creare prompt strutturati e parametrizzati
- Ps.61. Qual è la funzione principale della Memory in LangChain?
- ✓B. Ricordare il contesto delle conversazioni precedenti
- Ps.62. In CrewAI, quale elemento rappresenta una singola unità di lavoro assegnata a un agente?
- ✓C. Task
- Ps.63. Qual è la differenza principale tra LangChain e CrewAI?
- ✓A. LangChain si concentra su pipeline e strumenti, CrewAI sulla coordinazione multi-agente
- Ps.64. Cosa rappresenta il campo **finish\_reason** in una risposta API di OpenAI?
- ✓B. Il motivo per cui il modello ha terminato la generazione
- Ps.65. Per quale motivo può essere utile lo streaming delle risposte di un LLM?
- ✓C. Per ridurre la latenza e visualizzare il testo man mano che viene generato
- Ps.66. In LangChain, cosa fa una **LLMChain**?
- ✓A. Collega un prompt a un modello e gestisce input e output
- Ps.67. Qual è il vantaggio principale di usare Agents in LangChain?
- ✓C. Consentono al modello di scegliere dinamicamente quali strumenti utilizzare
- Ps.68. Qual è lo scopo dei Tools in LangChain?
- ✓A. Esporre funzioni esterne che il modello può invocare
- Ps.69. In CrewAI, cosa definisce il parametro **goal** di un agente?
- ✓B. L'obiettivo esplicito che guida le decisioni dell'agente
- Ps.70. In CrewAI, come possono essere eseguiti i task?
- ✓A. In modo sequenziale o in parallelo

- Ps.71. Cosa rappresenta un Crew in CrewAI?
- ✓C. Un insieme di agenti e task coordinati
- Ps.72. Quali sono le tre fasi principali dell'esecuzione di un programma Python?
- ✓B. Interpreter Initialization, Compilation in bytecode, Bytecode Interpretation
- Ps.73. Qual è il compito principale della fase di Interpreter Initialization?
- ✓B. Caricare moduli built-in, gestire memoria e creare il GIL
- Ps.74. Cosa rappresenta un frame durante l'esecuzione del bytecode?
- ✓B. Un'unità di esecuzione con codice, scope e stack
- Ps.75. Come funziona la call stack in Python?
- ✓A. Ogni funzione aggiunge un frame; al ritorno il frame viene rimosso
- Ps.76. Qual è la funzione del modulo `dis` in Python?
- ✓A. Disassemblare il bytecode in opcodes leggibili
- Ps.77. Quale meccanismo di gestione memoria usa CPython?
- ✓B. Reference counting con garbage collector per cicli
- Ps.78. Perché esiste il Global Interpreter Lock (GIL)?
- ✓B. Per semplificare la gestione memoria, evitando race condition
- Ps.79. Come fanno librerie come NumPy a ottenere parallelismo nonostante il GIL?
- ✓B. Rilasciano temporaneamente il GIL in codice C esterno
- Ps.80. Qual è la caratteristica principale della Python Virtual Machine?
- ✓A. È un interprete stack-based che esegue bytecode
- Ps.81. Qual è lo scopo del value stack nella VM?
- ✓C. Tenere operandi e risultati temporanei durante l'esecuzione



## 4 Domande aperte sul prodotto

- Ap.1. Si descriva il design pattern factory method e si evidenzi la sua struttura in un linguaggio di programmazione a scelta.
- Ap.2. Si descriva il design pattern del prototype e si evidenzi la sua struttura specifica in Java.
- Ap.3. Si descriva il design pattern del singleton e si evidenzi la sua struttura specifica in Python.
- Ap.4. Si descriva il design pattern del singleton e si definisca il codice associato in un linguaggio di programmazione che permetta di implementare le caratteristiche di questo pattern.
- Ap.5. Si descriva il design pattern del clone e si evidenzi la sua struttura in un linguaggio di programmazione a scelta.
- Ap.6. Si dettagli la differenza tra overloading e overriding.

## 5 Domande aperte sul modellamento data una storia

### 5.1 Esempio di domande

- Md.1. Si generi il diagramma a classi in Python della storia nel suo complesso.
- Md.2. Si estraggano gli oggetti e i metodi.
- Md.3. Si generi il diagramma a classi della storia nel suo complesso estraendo gli oggetti.
- Md.4. Si estraggano i metodi.
- Md.5. Si determinino almeno due storie atomiche estratte dalla descrizione.
- Md.6. Si presenti un diagramma di attività in UML desumibile da una parte qualunque della descrizione.
- Md.7. Si descriva il design pattern del singleton e si evidenzi la sua struttura specifica in Python.

### 5.2 Esempio di storie

#### 5.2.1 Esempio di una storia: da “Il funerale della volpe” di Gianni Rodari

Una volta le galline trovarono la volpe in mezzo al sentiero. Aveva gli occhi chiusi, la coda non si muoveva. - È morta, è morta - gridarono le galline. - Facciamole il funerale. Difatti suonarono le campane a morto, si vestirono di nero e il gallo andò a scavare la fossa in fondo al prato. Fu un bellissimo funerale e i pulcini portavano i fiori. Quando arrivarono vicino alla buca la volpe saltò fuori dalla cassa e mangiò tutte le galline. La notizia volò di pollaio in pollaio. Ne parlò perfino la radio, ma la volpe non se ne preoccupò. Lasciò passare un po' di tempo, cambiò paese, si sdraiò in mezzo al sentiero e chiuse gli occhi. Vennero le galline di quel paese e subito gridarono anche loro: - È morta, è morta! Facciamole il funerale. Suonarono le campane, si vestirono di nero e il gallo andò a scavare la fossa in mezzo al granoturco. Fu un bellissimo funerale e i pulcini cantavano che si sentivano in Francia. Quando furono vicini alla buca, la volpe saltò fuori dalla cassa e mangiò tutto il corteo. La notizia volò di pollaio in pollaio e fece versare molte lacrime. Ne parlò anche la televisione, ma la volpe non si prese paura per nulla. Essa sapeva che le galline hanno poca memoria e campò tutta la vita facendo la morta. E chi farà come quelle galline vuol dire che non ha capito la storia...

### 5.2.2 Esempio di una storia: da “Distrazione interplanetaria” di Gianni Rodari

Chissà se a quest'ora su Marte,  
su Mercurio o Nettuno,  
qualcuno  
in un banco di scuola  
sta cercando la parola  
che gli manca  
per cominciare il tema  
sulla pagina bianca.  
E certo nel cielo di Orione,  
dei Gemelli, del Leone,  
un altro dimentica  
nel calamaio  
i segni d'interpunzione...  
come faccio io.  
Quasi lo sento  
Lo scricchiolio  
Di un pennino  
In fondo al firmamento:  
in un minuscolo puntino  
della Via Lattea  
un minuscolo scolaretto  
sul suo libro di storia  
disegna un pupazzetto.  
Lo sa che non sta bene  
E anch'io lo so:  
ma rideremo insieme  
quando lo incontrerò.

### 5.2.3 Esempio di una storia: da “Il Paese con la “S” davanti” di Italo Calvino

Giovannino Perdigiorno era un grande viaggiatore. Viaggia e viaggia, capitò nel paese con l'esse davanti.  
“Ma che razza di paese è?” domandò a un cittadino che prendeva il fresco sotto un albero. Il cittadino, per tutta risposta, cavò di tasca un temperino e lo mostrò bene aperto sul palmo della mano.  
“Vede questo?”  
“E' un temperino”  
“Tutto sbagliato. Invece è uno stemperino, cioè un temperino con l'esse davanti. Serve a far ricrescere le matite, quando sono consumate, ed è molto utile nelle scuole”.  
“Magnifico”. disse Giovannino. “E poi?”  
“Poi abbiamo lo staccapanni”.

“Vorrà dire l’attaccapanni”.

“L’attaccapanni serve a ben poco, se non avete il cappotto da attaccarci. Col nostro staccapanni è tutto diverso. Lì non bisogna attaccarci niente, c’è già tutto attaccato. Se avete bisogno di un cappotto andate lì e lo staccate. Chi ha bisogno di una giacca, non deve mica andare a comprarla: passa dallo staccapanni e la stacca. C’è lo staccapanni d’estate e quello d’inverno, quello per uomo e quello per signora. Così si risparmiano tanti soldi”.

#### **5.2.4 Esempio di una storia: da “Gli amori difficili” di Italo Calvino**

L’isolotto aveva rive alte, di roccia. Sopra cresceva la macchia fila e bassa della vegetazione che resiste vicino al mare. Nel cielo volavano i gabbiani. Era una piccola isola vicino alla costa, deserta, incolta: in mezz’ora si poteva farne il giro in barca, o anche in canotto di gomma, come quello di quei due che venivano avanti, l’uomo pagaiando tranquillo, la donna coricata a prendere il sole. Avvicinandosi l’uomo tese l’orecchio. - Cos’hai sentito? - chiese lei. - Silenzio, - lui disse. - Le isole hanno un silenzio che si sente. Di fatto, ogni silenzio consiste della rete di rumori minuti che l’avvolge: il silenzio dell’isola si staccava da quello del calmo mare circostante perché era percorso da fruscii vegetali, da versi d’uccelli o da un improvviso frullo d’ali.