1 Teoria

- Che cos'è un design pattern? Perché è importante usarli nello sviluppo software?
- Descrivi il pattern Singleton. Qual è il suo scopo principale?
- Quali sono i principali problemi che il Singleton risolve?
- Come si implementa correttamente un Singleton in Java? Quali sono le varianti comuni?
- Quali sono i potenziali problemi del Singleton in contesti multithreading e come si possono risolvere?
- Che cos'è un Iterator? Qual è il suo scopo nei pattern di progettazione?
- Come funziona il pattern Iterator? Quali metodi fondamentali contiene un Iterator?
- Quali sono i vantaggi dell'uso di un Iterator rispetto all'accesso diretto agli elementi di una collezione?
- In Java, quali interfacce standard definiscono l'Iterator?
- Puoi fare un esempio di utilizzo di un Iterator per scorrere una collezione?

2 Esercizi sul Singleton

1. **Implementazione Singleton semplice**
   Scrivi una classe `Configurazione` che implementa il pattern Singleton. Assicurati che ci sia un solo oggetto istanziato e che l'accesso sia tramite un metodo statico.
2. **Singleton thread-safe**
   Modifica la classe `Configurazione` per renderla thread-safe usando la sincronizzazione.
3. **Singleton con inizializzazione lazy**
   Implementa un Singleton che istanzia l'oggetto solo al primo utilizzo (lazy initialization).
4. **Test del Singleton**
   Scrivi un breve programma che prova a creare più istanze della classe Singleton e verifica che si tratti della stessa istanza.
5. **Singleton e serializzazione**
   Spiega come garantire che un Singleton rimanga unico anche dopo la serializzazione e deserializzazione.

3 Esercizi sull'Iterator

1. **Uso di Iterator su ArrayList**
   Crea un `ArrayList` di interi e usa un Iterator per stampare tutti gli elementi.

2. **Iterator su una struttura dati personalizzata**
   Definisci una semplice classe `ListaDiNomi` che contiene una lista interna di stringhe. Implementa un metodo che restituisce un Iterator per scorrere i nomi.

3. **Rimozione di elementi con Iterator**
   Usa un Iterator su una collezione per rimuovere tutti gli elementi che soddisfano una certa condizione (ad esempio numeri pari).

4. **Implementazione manuale di un Iterator**
   Crea una classe `Contatore` che contiene un array di numeri interi e implementa manualmente un Iterator per scorrere questi numeri.

5. **Confronto tra for-each e Iterator**
   Scrivi un breve testo che spiega le differenze tra l'uso del ciclo for-each e l'uso esplicito di un Iterator. Quando è preferibile usare uno rispetto all'altro?