1 Teoria

- Che cos'è un flusso (stream) in Java? Qual è la differenza tra flussi di byte e di caratteri?
- Quali classi Java vengono utilizzate per la lettura e la scrittura di file di testo?
- Come si crea un file in Java? Quali metodi sono disponibili per verificare l'esistenza di un file?
- Come si legge un file di testo riga per riga in Java? Quali classi e metodi si utilizzano?
- Come si scrive su un file di testo in Java? Quali classi e metodi si utilizzano?
- Cos'è il buffering in I/O? Quali vantaggi offre l'uso di buffer nella lettura e scrittura di file?
- Come si gestisce la chiusura automatica delle risorse in Java durante le operazioni di I/O?
- Quali sono le differenze tra le classi `File`, `FileReader`, `BufferedReader`, `FileWriter` e `BufferedWriter`?
- Cos'è il package `java.nio.file` e come si differenzia dal package `java.io` nella gestione dei file?
- Come si gestiscono le eccezioni comuni durante le operazioni di I/O, come `IOException` e `FileNotFoundException`?

2 Gestione dei File

1. **Creazione di un file e scrittura di dati**
   Scrivi un programma che crea un file chiamato `esempio.txt` e vi scrive la stringa "Ciao, mondo!" al suo interno.
2. **Lettura di un file riga per riga**
   Scrivi un programma che apre il file `esempio.txt` e stampa ogni riga del file sulla console.
3. **Scrittura di più righe su un file**
   Scrivi un programma che scrive tre righe di testo nel file `esempio.txt`, sovrascrivendo il contenuto esistente.
4. **Aggiunta di contenuto a un file esistente**
   Scrivi un programma che apre il file `esempio.txt` in modalità append e aggiunge una nuova riga di testo alla fine del file.

5. **Lettura di un file utilizzando `Files.readAllLines`**
   Scrivi un programma che legge tutte le righe del file `esempio.txt` utilizzando il metodo `Files.readAllLines` e le stampa sulla console.

6. **Verifica dell'esistenza di un file**
   Scrivi un programma che verifica se il file `esempio.txt` esiste e stampa un messaggio appropriato sulla console.

7. **Copia di un file**
   Scrivi un programma che copia il contenuto di `esempio.txt` in un nuovo file chiamato `copia.txt`.

8. **Eliminazione di un file**
   Scrivi un programma che elimina il file `copia.txt` se esiste.

9. **Lettura di un file con `Scanner`**
   Scrivi un programma che legge il contenuto del file `esempio.txt` utilizzando la classe `Scanner` e lo stampa sulla console.

10. **Gestione delle eccezioni durante le operazioni di I/O**
    Scrivi un programma che tenta di aprire il file `esempio.txt` e gestisce eventuali eccezioni `IOException` e `FileNotFoundException` con messaggi appropriati.