## Esercizio

Occorre creare il codice di un progetto Java che implementa la gestione di un **catalogo di libri e ebook**, introducendo una struttura basata su **quattro tipi di eccezioni specifiche**.

**Classi coinvolte:**

1. **Classe `Libro`**:
   - Rappresenta un libro disponibile nella biblioteca.
   - Campi:
     - `titolo` (String): titolo del libro.
     - `autore` (String): autore del libro.
     - `disponibile` (boolean): indica se il libro è disponibile per il prestito.
   - Metodi:
     - Costruttore: inizializza i campi e verifica che titolo e autore siano validi.
     - `prendiInPrestito`: permette di prendere in prestito un libro se disponibile.
2. **Classe `Ebook`** (sottoclasse di `Libro`):
   - Rappresenta un ebook con limitazioni sui prestiti.
   - Campi aggiuntivi:
     - `maxPrestiti` (int): numero massimo di prestiti consentiti.
     - `prestitiCorrenti` (int): numero di prestiti effettuati finora.
   - Metodi:
     - Costruttore: oltre a inizializzare i campi del libro, definisce il limite massimo di prestiti.
     - `prendiInPrestito`: permette di prendere in prestito un ebook se non si supera il limite di prestiti.

**Requisiti:**

1. **Creare quattro eccezioni personalizzate**:
    - **`LibroNonDisponibileException`** : deve essere lanciata quando si tenta di prendere in prestito un libro non disponibile.
    - **`PrestitiMassimiSuperatiException`** : deve essere lanciata quando si supera il limite massimo di prestiti per un ebook.
    - **`TitoloNonValidoException`** : deve essere lanciata quando si cerca di creare un libro o un ebook con un titolo nullo o vuoto.
    - **`AutoreNonValidoException`** : deve essere lanciata quando si cerca di creare un libro o un ebook con un autore nullo o vuoto.
2. **Modificare il costruttore della classe `Libro`** :
    - Aggiungere controlli per verificare che il **titolo** e l'**autore** siano validi (non nulli o vuoti).
    - In caso di titolo non valido, lanciare un'eccezione di tipo `TitoloNonValidoException` .
    - In caso di autore non valido, lanciare un'eccezione di tipo `AutoreNonValidoException` .
3. **Modificare il metodo `prendiInPrestito` nella classe `Libro`** :
    - Se il libro non è disponibile per il prestito ( `disponibile == false` ), lanciare un'eccezione di tipo `LibroNonDisponibileException` .
4. **Modificare il metodo `prendiInPrestito` nella classe `Ebook`** :
    - Aggiungere un controllo per verificare che il numero massimo di prestiti ( `maxPrestiti` ) non sia stato superato.
    - Se il numero massimo di prestiti è stato raggiunto, lanciare un'eccezione di tipo `PrestitiMassimiSuperatiException` .
5. **Nella classe `Main`** :
    - Creare diversi oggetti `Libro` ed `Ebook` e simulare situazioni che scatenino ciascuna delle quattro eccezioni:
        - **`TitoloNonValidoException`** : Provare a creare un libro o un ebook con un titolo nullo o vuoto.
        - **`AutoreNonValidoException`** : Provare a creare un libro o un ebook con un autore nullo o vuoto.
        - **`LibroNonDisponibileException`** : Provare a prendere in prestito un libro non disponibile.
        - **`PrestitiMassimiSuperatiException`** : Provare a prendere in prestito un ebook dopo aver raggiunto il limite massimo di prestiti.
6. **Extra**:

- Usare una libreria di logging (es. LogBack) per stampare i messaggi di errore ogni volta che viene catturata un'eccezione.
- Aggiungere un metodo di utilità in `Main` per stampare lo stato corrente del libro o dell'ebook (es. disponibilità, numero di prestiti correnti, ecc.).