Esercizio Libro

Scrivere una classe Libro contenente i seguenti campi.

Dati:

- autori: String
- titolo: String
- annoDiPubblicazione: Integer
- codiceISBN: String

Metodi:

- costruttore con tre argomenti indicanti autori, titolo e codice ISBN
  - (il valore di default dell'anno di pubblicazione **deve essere** pari a `Integer.MAX_VALUE`);
- `toString()`: che restituisce una stringa nel formato "autori, titolo, anno, ISBN"
  - (cioè **i quattro campi** dati separati da virgola+spazio) se l'anno è diverso da `Integer.MAX_VALUE`
  - e "**autori, titolo, ISBN**" (cioè i tre campi dati separati da virgola+spazio) se l'anno è `Integer.MAX_VALUE`.
- `cambiaAnno(a)`: che, dato un anno a, imposta l'anno di pubblicazione del libro con il valore dell'argomento a;
- `inAnno(a)`: che, dato un anno di pubblicazione a, verifica che il libro sia stato pubblicato in quell'anno;
- `stessoAnno(l)`: che, dato un libro l, verifica che il libro sia stato pubblicato nello stesso anno di l;
- `cambiaAnno(l)`: che, dato un libro l, imposta l'anno di pubblicazione del libro con quella del libro l;

Scrivere la soluzione in modo da poter compilare il seguente programma di prova senza doverlo modificare:

```
public class ProvaEserc1 {
    public static void main (String[] args) {

        Libro l1 = new Libro("J.D. Salinger", "Il giovane Holden",
"123456789");
        Libro l2 = new Libro("Agatha Christie", "Dieci piccoli indiani",
"987654321");
        Libro l3 = new Libro("Antoine de Saint-Exupéry", "Il Piccolo
Principe", "000000001");

        String  s = l2.toString();
        boolean b = s.equals("Agatha Christie, Dieci piccoli indiani,
987654321");
        System.out.println("Test 1. Metodo toString(): "+b);

        l2.cambiaAnno(1939);
        b = l2.toString().equals("Agatha Christie, Dieci piccoli indiani,
1939, 987654321");
        System.out.println("Test 2. Metodi toString() e cambiaAnno(a):
```

```
"+b);

        b = l1.inAnno(Integer.MAX_VALUE) && !l1.inAnno(2020);
        System.out.println("Test 3. Metodo inAnno(a): "+b);

        b = l1.stessoAnno(l3) && !l1.stessoAnno(l2);
        System.out.println("Test 4. Metodo stessoAnno(l): "+b);

        l2.cambiaAnno(l1);
        b = l2.toString().equals("Agatha Christie, Dieci piccoli indiani,
987654321");
        System.out.println("Test 5. Metodo cambiaAnno(l): "+b);
    }
}
```

Esercizio Stanza Prenotazione

Realizzare le classi **Stanza** e **Prenotazione**, che rappresentano una camera d'albergo e una prenotazione per la camera.

Il metodo **riserva** di Stanza accetta un nome, la data di inizio e di fine prenotazione, e restituisce un oggetto di tipo **Prenotazione**.

Se la camera è occupata in una delle giornate richieste, il metodo lancia un'eccezione.

Per semplicità, una data è rappresentata da un numero intero tra 0 a 365.

Il metodo **prenotazioni()** di **Stanza** consente di scorrere l'elenco delle prenotazioni, in ordine cronologico.

L'implementazione deve rispettare il seguente esempio d'uso:

Esempio d'uso:

```
Stanza r = new Stanza();

Prenotazione p1 = r.riserva("Mario Rossi", 105, 120);
Prenotazione p2 = r.riserva("Giuseppe Verdi", 5, 20);
Prenotazione p3 = r.riserva("Brad Pitt", 20, 22);
Prenotazione p4 = r.riserva("Angiolina Jolie", 200, 222);

for (Prenotazione p: r . prenotazioni ()){
    System.out.println(p.getName());
}
```

Output atteso:

- Mario Rossi
- Giuseppe Verdi
- Brad Pitt

- Angiolina Jolie

upgrade

Gestire la data con la classe **LocalDate**

Esercizio Curriculum

Un oggetto **Curriculum** rappresenta una sequenza di lavori, ognuno dei quali è un'istanza della classe **Job**.

Il costruttore di **Curriculum** accetta il nome di una persona. Il metodo **addJob** aggiunge un lavoro alla sequenza, caratterizzato da una descrizione e dall'anno di inizio, restituendo un nuovo oggetto di tipo **Job**.

Infine, la classe **Job** offre il metodo **next**, che restituisce in tempo costante il lavoro successivo nella sequenza (oppure **null**).

Implementare le classi Curriculum e Job, rispettando il seguente caso d'uso.

Caso d'uso:

```
Curriculum cv = new Curriculum("Umberto Eco");
Curriculum.Job j1 = cv.addJob("Insegnante", 1980);
Curriculum.Job j2 = cv.addJob("Scrittore", 1990);
Curriculum.Job j3 = cv.addJob("Linguista", 2000);
System.out.println(j2 .next()) ;
System.out.println(j3 .next()) ;
```

Output:

- Linguista: 2000
- null

Esercizio enumerazioni

Implementare la classe enumerata **Status**, che rappresenta le 4 modalità di un programma di *instant messaging*: **ONLINE, BUSY, HIDDEN e OFFLINE**.

Il metodo **isVisible** restituisce vero per i primi due e falso per gli altri.

Il metodo **canContact** accetta un altro oggetto **Status** x e restituisce vero se un utente in questo stato può contattare un utente nello stato x e cioè se questo stato è diverso da **OFFLINE** e lo stato x è visibile.

Esempio d'uso:

```
Status a = Status.BUSY, b = Status.HIDDEN;
System.out.println(a. isVisible ()) ;
```

```
System.out.println(a.canContact(b));
```

Output:

- true
- false