Biblioteca comunale

Progetto di Programmazione dell'Appello di Febbraio 2017

Introduzione

In questo esame vi chiederemo di scrivere un insieme di classi per la modellazione di una biblioteca comunale.

Classi

Descriviamo ora nei dettagli come realizzare le classi necessarie al progetto. Se lo ritenete, potete apportare cambiamenti alle segnature dei metodi proposti e anche – se lo ritenete – alla struttura delle classi, restando ovviamente all'interno dei requisiti esposti nel paragrafo introduttivo.

N.B. (1): Oltre ai metodi qui indicati, aggiungete a ogni classe un valido metodo `toString()` e, dove lo ritenete, anche un metodo `equals` e un metodo `hashCode` (con le segnature appropriate e che soddisfino i contratti previsti).

N.B. (2): Nel seguito, useremo il termine *lista di X* per indicare un qualunque tipo che consenta di conservare delle sequenze di oggetti di tipo `X`: potete implementare una lista con un array (`X[]`) oppure usando ad esempio delle `ArrayList<X>` (pacchetto `java.util`).

Classe Data

Sebbene nella libreria standard di Java esistano molte classi per la rappresentazione e manipolazione delle date, per scopi didattici vi chiediamo in primo luogo di creare una classe di nome `Data` per rappresentare le date. Una `Data` rappresenta una data (intesa come giorno, mese e anno), e la classe deve disporre dei seguenti costruttori e metodi:

- `Data(int g, int m, int a)`: costruisce una data con giorno (compreso fra 1 e 31), mese (compreso fra 1 e 12) e anno specificati. Non è richiesto (sebbene sia considerato apprezzabile) che il metodo sollevi delle eccezioni nel caso che gli argomenti non siano corretti.
- `Data()`: costruisce la data corrente. Per ottenere il giorno corrente potete invocare:

```
Calendar.getInstance().get(Calendar.DAY_OF_MONTH);
```

La classe `Calendar` è nel pacchetto `java.util`. Similmente l'anno può essere ottenuto come

```
Calendar.getInstance().get(Calendar.YEAR);
```

Per il mese potete chiamare

```
Calendar.getInstance().get(Calendar.MONTH);
```

ma ricordatevi che il valore restituito dall'ultima chiamata è 0 per gennaio, 1 per febbraio ecc.

- `int giorniDaEpoca()`: restituisce i giorni trascorsi dal 1/1/1970 alla data su cui è invocato.
- `int giorniDa(Data d)`: restituisce i giorni trascorsi dalla data `d` a quella su cui il metodo è invocato. Il valore restituito è negativo se la data su cui è invocato viene prima di `d`.
- `boolean vienePrimaDi(Data d)`: restituisce `true` se e solo se la data su cui è invocata viene prima della data `d`.
- `Data successiva()`: restituisce la data ottenuta come il giorno 1 del mese `m+2` (dove `m` è il mese della data su cui è invocato). Ad esempio, se lo invocate sulla data 3/4/2015, la data successiva è il 1/6/2015. Se lo invocate sulla data 17/11/2016, la data successiva è il 1/1/2017.

Classe Libro

Le istanze di questa classe rappresentano i libri posseduti dalla biblioteca. Ogni libro è caratterizzato da alcuni dati, cioè:

- il `titolo` (una stringa)
- gli `autori` (un array di stringhe, ognuna delle quali rappresenta il nome di un autore)
- un `identificativo` (un numero intero che rappresenta il libro, e che è diverso per libri diversi)
- un booleano `prestabile` che è `true` se e solo se è possibile dare in prestito questo libro.

Inserite nella classe i seguenti costruttori e metodi:

- `Libro(String titolo, String[] autori, boolean prestabile)`: costruisce un libro con i dati indicati
- `Libro(String titolo, String autore, boolean prestabile)`: costruisce un libro con un solo autore
- `Libro(String titolo, String[] autori)`: costruisce un libro prestabile con i dati indicati
- `Libro(String titolo, String autore)`: costruisce un libro prestabile con un solo autore.

Classe Utente

Le istanze di questa classe rappresentano gli utenti della biblioteca. Un utente è costituito da:

- un `nome` (una stringa)
- un `cognome` (una stringa)
- una `dataDiNascita` (una data)
- un `identificativo` (un numero intero che rappresenta l'utente, e che è diverso per utenti diversi).

Inserite nella classe i costruttori e i metodi che ritenete opportuni.

Classe Prestito

Un prestito rappresenta il prestito di un libro a un certo utente, attuato da una certa data a un'altra data. La classe ha i seguenti costruttori e metodi:

- `Prestito(Libro x, Utente u, Data da, Data a)` : crea un prestito per il libro `x` , che viene dato all'utente `u` dalla data `da` (compresa) alla data `a` (esclusa).
- `Prestito(Libro x, Utente u, Data da)` : crea un prestito per il libro `x` , che viene dato all'utente `u` dalla data `da` (compresa) alla data ottenuta invocando il metodo `successiva()` su `da` .
- `boolean riguardaLibro(Libro t)` : restituisce `true` se e solo se il prestito su cui è invocato riguarda il libro passato come argomento.
- `boolean riguardaData(Data d)` : restituisce `true` se e solo se la data passata come argomento rientra nel periodo per cui vale questo prestito.

Classe Biblioteca

Una biblioteca ha i seguenti campi:

- un `nome` (il nome della biblioteca, una stringa)
- una lista di libri posseduti (all'inizio vuota)
- una lista di utenti (all'inizio vuota)
- una lista di prestiti (all'inizio vuota).

Oltre al costruttore (che prende come unico argomento il nome della biblioteca), dovete implementare i seguenti metodi:

- `void nuovoUtente(Utente x)` : aggiunge l'utente specificato.
- `void nuovoLibro(Libro x)` : aggiunge il libro specificato.
- `int numeroUtenti()` : restituisce il numero di utenti.
- `int numeroLibri()` : restituisce il numero di libri.
- `Libro[] cercaLibro(String x)` : restituisce l'array di libri che contengono nel titolo la stringa `x` .
- `boolean disponibile(Libro x)` : restituisce true se e solo se il libro è prestabile e nessun prestito che riguarda questo libro copre la data di oggi.

- `int quantiPrestiti(Utente x)` : restituisce il numero di prestiti dell'utente `x` alla data attuale.
- `boolean daiInPrestito(Libro x, Utente u, Data finoA)` : controlla se il libro `x` può essere dato in prestito oggi e se l'utente `u` non ha più di quattro prestiti attivi; se una di queste due condizioni è falsa, restituisce `false` . Altrimenti, crea il prestito con i dati indicati (da oggi fino alla data `finoA`) e l'aggiunge alla lista dei prestiti.
- `boolean daiInPrestito(Libro x, Utente u)` : come sopra, ma stavolta la data fino a cui dura il prestito è la data successiva a quella attuale (nel senso del metodo `successiva()` di `Data`).
- `Utente chiHaInPrestito(Libro x)` : restituisce l'utente che ha attualmente in prestito il libro indicato (oppure `null` se il libro non è attualmente in prestito). Per ottenere questo risultato, occorre scandire la lista dei prestiti, guardare quali sono quelli che riguardano il libro `x` e vedere se uno dei prestiti copre la data attuale.