Esercitazione

Esercizio Libreria

- Una libreria ha deciso di informatizzare il proprio archivio delle giacenze. Si è creato un file in cui, per ogni libro disponibile in magazzino, sono stati inseriti i seguenti elementi:

- titolo del libro; - autore; - editore; - numero di copie disponibili.

La libreria ha la necessità di effettuare le seguenti operazioni:

- cercare tutti i libri di un particolare autore;
- cercare tutti i libri che contengono una determinata stringa nel titolo;
- cercare i libri che hanno il numero di copie massimo;
- listare tutti i libri per cui il numero di copie disponibili è al di sotto di una determinata soglia.
- aggiornare il numero di copie di un libro. Il libro da aggiornare è identificato dal titolo.

L'operazione di aggiornamento deve essere propagata al file.

Esercizio Libreria

- Creare inoltre una classe main per testare le classi ed i metodi definiti. In particolare occorre
  - leggere da un file contenente i dati dei 5 libri mostrati di seguito;
  - visualizzare i libri scritti da Camilleri;
  - visualizzare i libri che hanno la parola sogni nel titolo;
  - visualizzare il libro con il numero massimo di copie;
  - visualizzare i libri che hanno meno di 15 copie disponibili;
  - aggiungere 30 copie al libro sogni rossi.

Una voce di notte
Camilleri Andrea
Sellerio Editore Palermo
44
Fai bei sogni
Gramellini Massimo
Longanesi
14

Esercizio

- Implementare e testare una classe **`Bank`** che contenga un vettore di oggetti di tipo **`BankAccount`** e abbia i metodi:
  - addAccount(initialBalance, customerName)
  - deposit(account, amount)
  - withdraw(account, amount)
  - getBalance(account)
  - transfer(fromAccount, toAccount, amount)

Esercizio

- Si supponga di voler progettare e implementare parte del sistema informativo di una azienda di trasporti su rotaia. Per ogni treno occorrerà tenere traccia delle stazioni di fermata, della stazione di partenza e di quella di arrivo, oltre che dei relativi orari. Occorre poi che ad ogni treno sia associato il numero dei posti a sedere disponibili e il numero totale di chilometri percorsi. Nei treni espressi, infine, è previsto anche un servizio ristorante, e anche per questo servizio è necessario tenere traccia del numero di posti disponibili.
- Un utente di questo sistema informativo potrebbe essere interessato a determinare il numero di fermate effettuate da ciascun treno. Inoltre, chi utilizza tale sistema informativo potrebbe essere interessato a determinare il massimo ricavo realizzabile nell'erogazione di questo servizio. Tale ricavo dipende chiaramente da un parametro, ovvero dal prezzo che ogni passeggero dovrà pagare per percorrere un chilometro. Nei treni espressi occorrerà tenere conto anche del ricavo che si presume di ottenere in ogni chilometro da ognuno dei posti disponibili nel vagone ristorante (anch'esso fornito come parametro).

Esercizio

- Implementare le classi:
- Purse:
  - addCoin(Coin)
  - getTotal()
- Coin:
  - getValue()
  - getName()
- Mettere le classi in un pacchetto “it.unisa.prog2.money” e testare le classi con una classe di collaudo MoneyTest del pacchetto di default
- Dare delle precondizioni ragionevoli per i metodi e testarle con delle asserzioni

Esercizio

- Implementare e testare la classe ContoCorrente (pacchetto “money”)
- deposita(double importo)
  - pre-condizione importo >= 0
- preleva(double importo)
  - pre-condizione importo <= saldo
- restituisciSaldo()
- restituisciNumeroConto()

- Ogni conto corrente ha un numero progressivo che lo identifica, restituito da restituisciNumeroConto()