Esercitazione OOP 20230613

Si definisca un insieme di classi ed interfacce per rappresentare gli abbonamenti degli utenti di un Teatro

Requirements

Ogni **abbonato** è un utente con nome (String), età (int), email (String). Email e' un identificativo univoco per l'utente
Ogni **abbonamento** è caratterizzato da

- Abbonato
- codiceAbbonamento univoco (decidere autonomamente il formato e le strategie per mantenere univocità)
- un posto
- costoAbbonamento (double) calcolato come verrà specificato di seguito.

Il costo dell'abbonamento per un ogni abbonato dipenderà posto occupato (vedi seguito) e servizi aggiuntivi che può richiedere; specifici tipi di abbonato possono ottenere riduzioni in base al loro profilo (tipo)

Ci sono diversi profili speciali di abbonati, ciascuno può usufruire di particolari servizi e/o agevolazioni:

- Abbonato standard che non ha diritto a detrazioni
- studenti universitari e anziani (età>75), possono usufruire di riduzioni del prezzo e in base al loro tipo possono accedere ai servizi di cui si dirà nel seguito

Per ciascun tipo di abbonato occorre lanciare delle eccezioni in caso di inserimento di valori errati

Il teatro ha TOT posti divisi in 3 fasce (TOT=n°poltronissime + n°normali+ n°Galleria):
Per semplicità non ci si occuperà dell'assegnazione dello specifico posto. E' possibile ad esempio usare tre contatori per mantenere il memorizzare il numero di posti nelle 3 fasce.

Per l'abbonamento c'è un PrezzoBase, ma il costoAbbonamento va calcolato in funzione di possibili riduzioni o incrementi dovuti a profilo abbonato, al tipo di posto scelto e ai servizi addizionali richiesti

Le poltronissime, avranno un incremento prezzo di 40% rispetto al prezzo base dell'abbonamento, e i posti in galleria avranno un decremento del 30% rispetto al prezzo base.
Gli studenti universitari non hanno accesso alle poltronissime. I posti in galleria sono accessibili ad abbonati senza detrazioni e studenti universitari, non ad anziani.
Anziani e Studenti universitari usufruiscono di una detrazione del costo dell'abbonamento rispettivamente del 30% e 40% del costo totale.

Qui si parlerà del servizio parcheggio e del servizio culturale cui hanno accesso solo alcuni tipi di abbonati.

Il teatro ha un parcheggio con MAX posti
Possono richiedere un posto del parcheggio 1) gli anziani senza pagare prezzo aggiuntivo e 2) gli abbonati standard pagando un prezzo aggiuntivo di ParkFee euro.

Per la richiesta posto va implementato un metodo: requestParkPass che, se il posto è stato assegnato scrive su System.out: "pass abbonamento "+ abbonamento.toString() e restituisce un booleano che indica se l'operazione ha successo o meno

Il servizio culturale è riservato a AbbonatiStandard aggiungendo una CultureFee al costo totale e a Studenti Universitari senza costi aggiuntivi

Il teatro gestisce i propri abbonamenti mediante un applicativo che:

- Abbona i vari tipi di utente mantenendo le informazioni in una Map<EmailUtente, Abbonamento>
- Mantiene le informazioni di stato per assegnare codice abbonamento per riempire il teatro (per semplicità numero progressivo)
- Deve poter recuperare un abbonamento in base all'abbonato
- Deve poter permettere ad un abbonato di recedere dall'abbonamento
- Deve mantenere il guadagno del teatri accumulato sul costo degli abbonamenti
  Deve poter eseguire statistiche: valutare l'età media degli abbonati, ottenere il numero di studenti universitari
- Deve restituire la lista di tutti gli abbonati con riduzione

---

Gestione posti per singola rappresentazione:

I posti rimanenti possono essere occupati per le singole serate.
Per semplicità qui si prenotano solo biglietti in base alla disponibilità rimasta dagli abbonamenti. Invece di calcolarla dalla dimensione della Map di cui sopra, supponiamo sia dato da una configurazione nota con **MaxGallery** di posti disponibili in galleria (per semplicità supporremo pari a 10), **MaxFirst** di poltronissime disponibili (per semplicità supporremo pari a 10), **MaxNormal** (per semplicità supporremo pari a 10) di posti normali disponibili.

Tutte le informazioni relative alle prenotazioni a singoli eventi vengono mantenute all'interno di un oggetto Buffer che contiene un Map<String, Seat> in cui la chiave è un'informazione sull'utente (es. mail+num_utente) e il valore è il Seat(es. <Subscriber10, First>, <Subscriber11,Gallery>…). L'oggetto Buffer può essere modificato fino al momento dell'evento in cui gli utenti acquistano effettivamente il biglietto all'ingresso del teatro.
N.B. possono essere prenotati per non più di MaxXXX ticket di tipo XXX.

L'oggetto Buffer viene gestito concorrentemente da 4 Thread :

- un Produttore inserisce un utente con relativo ticket nel buffer se c'è spazio e non è gia presente. Il posto inserito è determinato casualmente tra i 3 delineati (N.B. come detto occorre tenere conto della capienza , cioè dei limiti sui tipi di posti e della quantità già prenotata per ogni tipo di posto). va in sleep per un periodo compreso tra 1 e 3 sec ed alla fine restituisce il numero posto assegnato
- un Consumatore che cancella una prenotazione (qualsiasi) e va in sleep per 5 sec

- 2 Thread, uno di amministrazione del teatro e l'altro di segreteria si comportano da Lettori eseguendo 2 attività diverse:
  - Il Thread amministrazione calcola l'introito complessivo attuale previsto (somma il costo dei biglietti di tutti quelli che hanno attualmente prenotato e calcola 80% supponendo che un 20% non verrà all'acquisto)
  - Il Thread Segreteria stampa a schermo l'intero contenuto di Map (in accordo all'algoritmo scrittori/lettori)
  - In entrambi i casi i thread vanno in sleep per un periodo compreso tra 10 e 30 sec

Non trattiamo l'acquisto effettivo al Botteghino la sera dell'evento.