Prova d'esame del 02/07/2018 - Turno A

Si consideri il database "extflightdelays", contenente informazioni su aeroporti, compagnie aeree, orari di partenza e di arrivo dei voli effettuati negli Stati Uniti durante il 2015. Il database (disponibile su Kaggle all'indirizzo: https://www.kaggle.com/usdot/flight-delays/data) è strutturato secondo il diagramma ER della pagina seguente.

Si intende costruire un'applicazione JavaFX che permetta di interrogare tale base dati, e calcolare informazioni a proposito dei voli tra i diversi aeroporti. L'applicazione dovrà svolgere le seguenti funzioni:

PUNTO 1

- a. Permettere all'utente di inserire una distanza media minima x (in miglia) e di selezionare il bottone "Analizza aeroporti".
- b. Alla pressione del bottone, creare un grafo che rappresenti gli aeroporti collegati da almeno un volo, e distanti in media almeno x miglia.
 Il grafo deve essere semplice, non orientato e pesato, i vertici devono rappresentare gli aeroporti, mentre gli archi devono indicare le

Esame 02-07-2018 TURNO A		
Disanza minima		Analizza aeroporti
Aeroporto di partenza	•	Aeroporti connessi
Miglia disponibili		Cerca itinerario

rotte tra gli aeroporti collegati tra di loro da almeno un volo. Il peso dell'arco rappresenta la <u>distanza media</u> tra i due aeroporti, calcolata come la media del campo *DISTANCE* di ciascun volo che li collega (poiché il grafo non è orientato, considerare tutti i voli in entrambe le direzioni: **A->B** e **B->A**). L'arco tra due aeroporti deve essere aggiunto solo se la distanza media è superiore a **x**. Per semplicità, si supponga che le distanze salvate nel database siano già espresse in miglia.

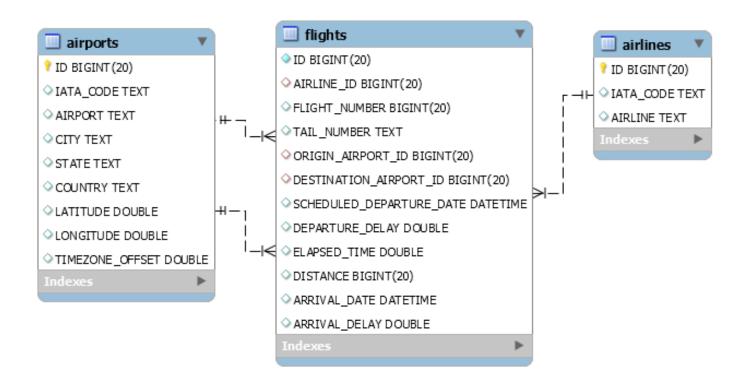
- c. Permettere all'utente di selezionare da un menu a tendina uno degli aeroporti presenti nel grafo (a1).
- d. Alla pressione del bottone "Aeroporti connessi", stampare l'elenco degli aeroporti adiacenti a quello selezionato, in ordine decrescente di distanza.

PUNTO 2

- a. L'utente inserisce nell'apposita casella il numero totale di miglia disponibili che è disposto a percorrere.
- Alla pressione del bottone "Cerca itinerario", il programma dovrà cercare l'itinerario di viaggio per cui l'utente possa visitare il maggior numero di città con le miglia disponibili, a partire dall'aeroporto di partenza a1 selezionato nel punto 1c. Tramite un algoritmo ricorsivo agente sul grafo creato al punto 1, trovare un cammino semplice e non ciclico, con il maggiore numero di nodi e tale da rispettare il vincolo sul numero massimo di miglia.
 - In particolare, dato un aeroporto, il passeggero sceglie la prossima destinazione tra tutte quelle disponibili, consumando un numero di miglia pari al peso dell'arco relativo alla tratta selezionata. Ignorare le date e gli orari di partenza e supporre che il passeggero non passi mai due volte per lo stesso aeroporto e non debba tornare a casa.
- c. Al termine della ricerca, il programma deve stampare l'itinerario, indicando gli aeroporti utilizzati e la distanza totale percorsa.

Nella realizzazione del codice, si lavori a partire dalle classi (Bean e DAO, FXML) e dal database contenuti nel progetto di base. È ovviamente permesso aggiungere o modificare classi e metodi.

Tutti i possibili errori di immissione, validazione dati, accesso al database, ed algoritmici devono essere gestiti, non sono ammesse eccezioni generate dal programma.



La tabella *airports* riporta tutti gli aeroporti presenti sul territorio statunitense, la tabella *airlines* tutte le compagnie aeree operanti, mentre la tabella *flights* contiene informazioni sui voli effettuati da una specifica compagnia aerea tra una coppia di aeroporti, riportando le date di partenza, di arrivo ed eventuali ritardi.