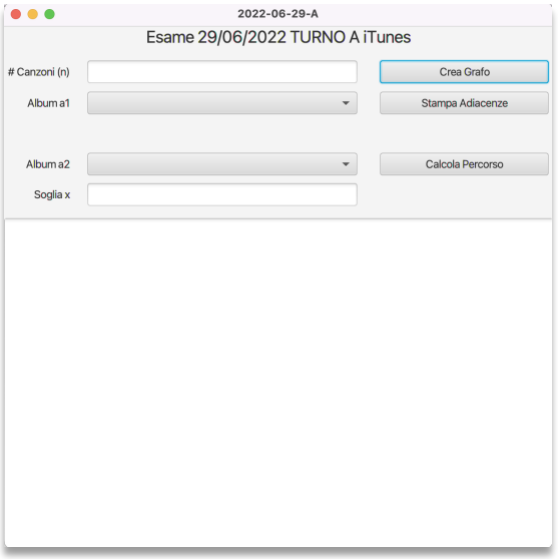
Esame del 29/06/2022 – Turno A

Si consideri il database **iTunes.sql**, presente nella cartella "database" del progetto base e tratto dalla piattaforma iTunes di Apple. Esso contiene informazioni su artisti (Artist), album musicali (Album) e canzoni (Track), ed è stato estratto a partire dai dati di un utente reale. Il diagramma ER del database è illustrato nella pagina seguente.

Si intende costruire un'applicazione JavaFX che permetta di che svolga le seguenti funzioni:

PUNTO 1



a. Permettere all'utente di inserire, nell'apposito campo di testo, un valore numerico *n*.
b. Alla pressione del bottone "Crea Grafo", si crei un grafo **semplice, orientato e pesato,** i cui vertici sono tutti gli album musicali (tabella *Album*) che contengono più di *n* canzoni (*Track*).
c. Per ogni coppia di album a1-a2, si consideri il loro **Δ canzoni**, definito come la differenza tra il numero di canzoni di a1 e il numero di canzoni di a2. I due album sono collegati tra di loro da un arco se e solo se tale differenza è **diversa da 0.** L'eventuale arco è orientato dall'album con meno canzoni verso l'album con più canzoni, e il peso, sempre positivo, è definito come il valore assoluto di **Δ canzoni.**
d. Si definisca il "bilancio" di un vertice come la differenza tra la somma dei pesi dei suoi archi entranti e la somma dei pesi dei suoi archi uscenti. Permettere all'utente di selezionare, dall'apposita tendina, un album *a1* tra quelli presenti nel grafo (elencati in ordine alfabetico di titolo). Alla pressione del bottone "Stampa Adiacenze", si si stampino tutti i nodi *successori* di *a1* in ordine decrescente di bilancio (vedere gli screenshot alle pagine seguenti per un esempio di stampa corretto).

PUNTO 2

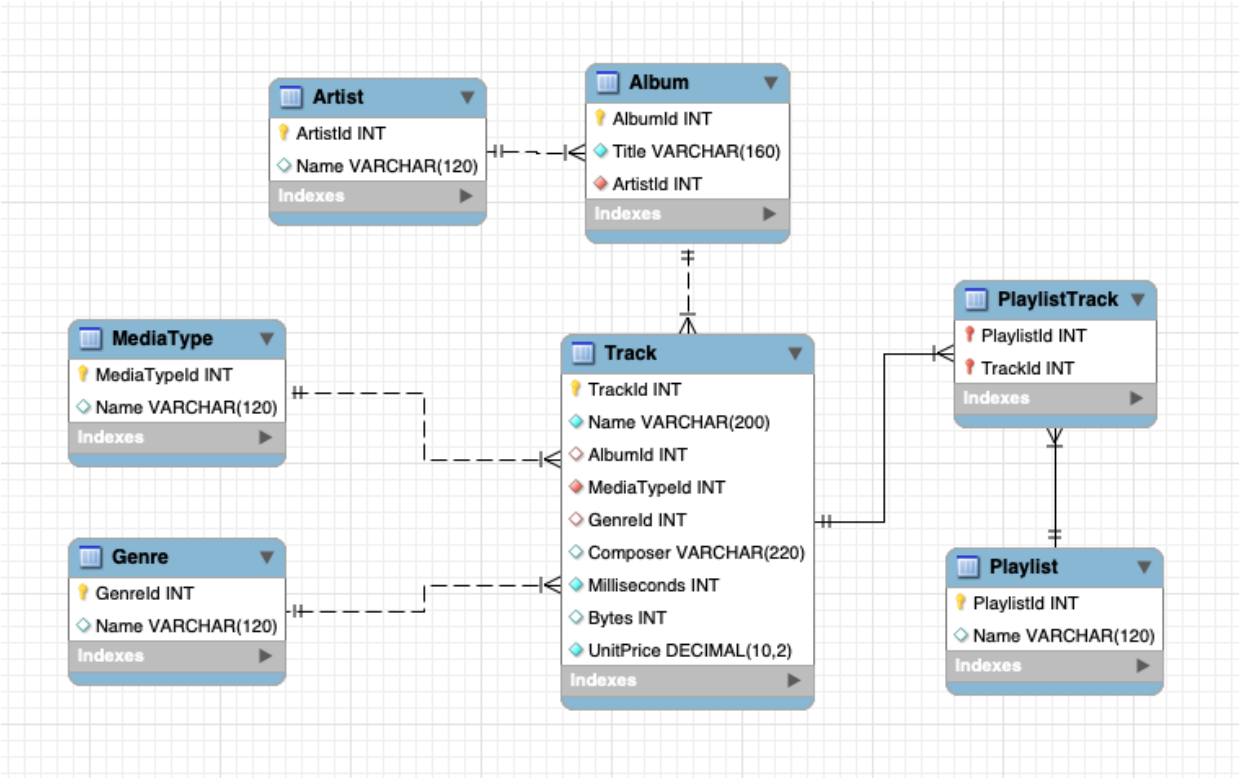
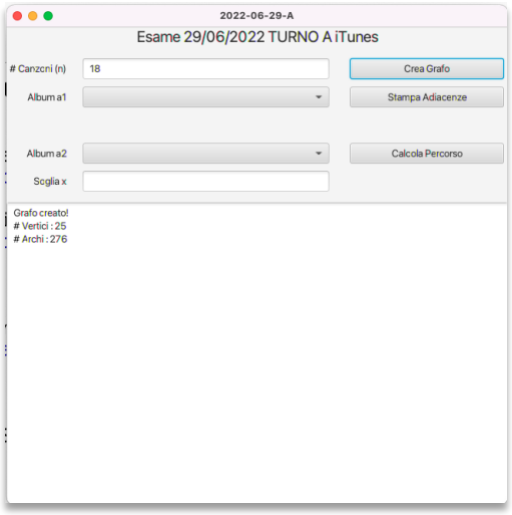
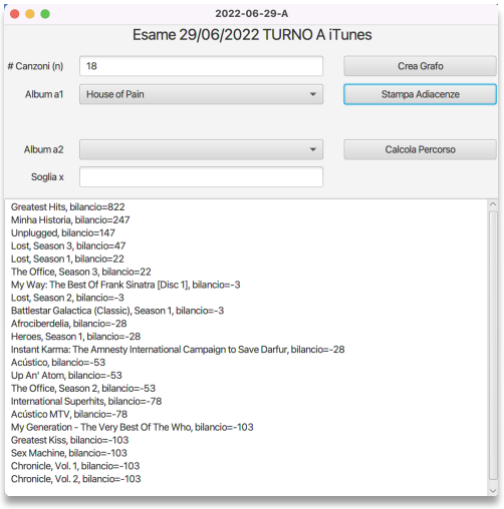
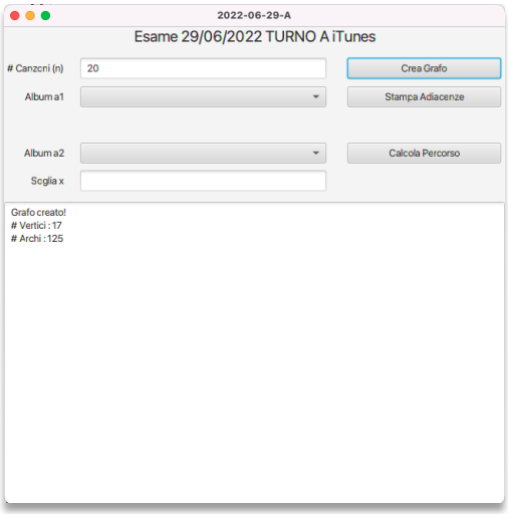
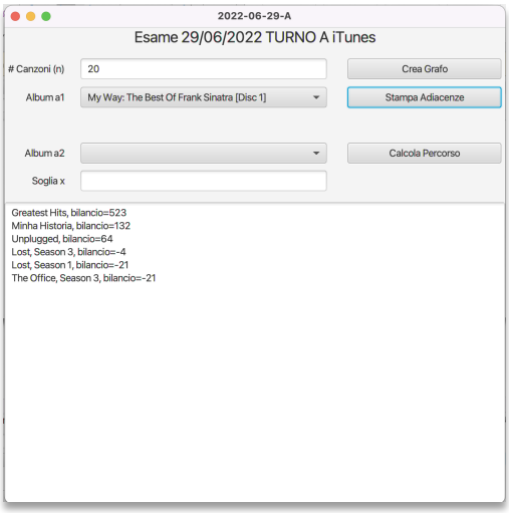
Permettere all'utente di inserire una soglia numerica *x*, e di selezionare, dall'apposita tendina, un ulteriore album *a2* tra quelli presenti nel grafo. Alla pressione del bottone "Calcola Percorso", trovare e stampare (se esiste) un cammino semplice sul grafo calcolato nel punto 1 che abbia le seguenti caratteristiche:

- parta da *a1* (selezionato al punto 1d) e termini in *a2*;
- attraversi solo archi con peso maggiore o uguale a *x*;
- tocchi il maggior numero di vertici che hanno un "bilancio" maggiore di quello del vertice di partenza *a1* (per il calcolo del "bilancio" di un vertice si veda il punto 1d).

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. Nelle pagine seguenti, sono disponibili due esempi di risultati per controllare la propria soluzione.

Le tabelle **Artist**, **Album** e **Track** contengono rispettivamente informazioni su artisti, album musicali e canzoni contenute. Ogni canzone è collegata a un genere (tabella **Genre**) ed è memorizzata con uno specifico formato (tabella **MediaType**). La tabella **Playlist** contiene informazioni sulle playlist salvate dall'utente. La tabella **PlaylistTrack** modella la relazione molti a molti tra playlist e canzoni.



ESEMPI DI RISULTATI PER CONTROLLARE LA PROPRIA SOLUZIONE: