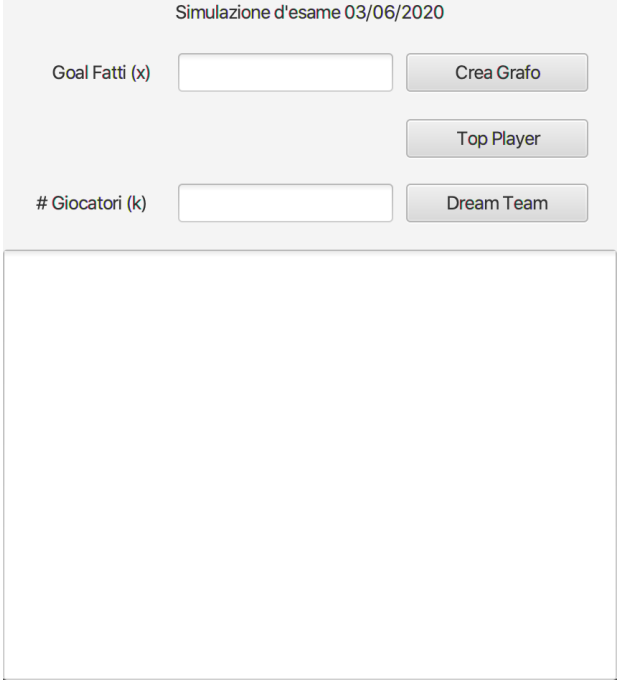
Simulazione d'esame 3/06/2020

Si consideri il database **PremierLeague.sql**, contenente informazioni sul campionato di calcio inglese della stagione 2011/2012. Il database è strutturato secondo il diagramma ER illustrato nella pagina seguente. Si intende costruire un'applicazione JavaFX che permetta di interrogare tale base dati. L'applicazione dovrà svolgere le seguenti funzioni:

PUNTO 1



a. Permettere all'utente di inserire, in un apposito campo di testo, un numero minimo ***x*** di *goal fatti (campo **Goals**, tabella **actions**)*

b. Alla pressione del bottone "Crea Grafo", si crei un grafo semplice, pesato e orientato i cui nodi sono i giocatori che hanno *segnato più di **x** goal in media a partita durante la stagione (provare con valori < 1).*

c. Un arco collega due giocatori (tra quelli precedentemente selezionati) se appartengono a squadre diverse, e sono scesi in campo da "titolari" (campo **starts = 1**, tabella **actions**) in almeno una partita in cui si sono affrontati, e se $\Delta \neq 0$ (v.sotto).
Dati due giocatori, in particolare, l'arco deve essere *orientato* dal giocatore che ha giocato più minuti all'interno di queste partite verso il giocatore che ne ha giocati di meno. I minuti giocati si deducono dal campo **TimePlayed**, tabella **actions**; fare attenzione che possono esistere più partite tra gli stessi giocatori, ed i relativi TimePlayed vanno sommati.
Il peso dell'arco, sempre > 0, rappresenta la differenza dei minuti giocati (Δ) dai due giocatori all'interno delle partite in cui si sono affrontati.
Se tale Δ è uguale a 0, **l'arco non deve essere inserito.**

d. Premendo il bottone "Top-Player", si trovi il giocatore che abbia "battuto", in termini di minuti giocati, il numero maggiore di avversari. Si stampino, contestualmente, gli avversari battuti ordinati in modo decrescente di Δ.
Suggerimenti: in un grafo orientato, è possibile utilizzare i seguenti metodi:
   - *outDegreeOf*: restituisce il numero di archi uscenti da un determinato vertice;
   - *inDegreeOf*: restituisce il numero di archi entranti in un determinato vertice;
   - *outgoingEdgesOf*: restituisce il set di archi uscenti da un determinato vertice;
   - *incomingEdgesOf*: restituisce il set di archi entranti in un determinato vertice;
   - *Graphs.successorListOf*: restituisce la lista di vertici raggiunti dagli outgoingEdges;
   - *Graphs.predecessorListOf*: restituisce la lista di vertici raggiunti dagli incomingEdges.

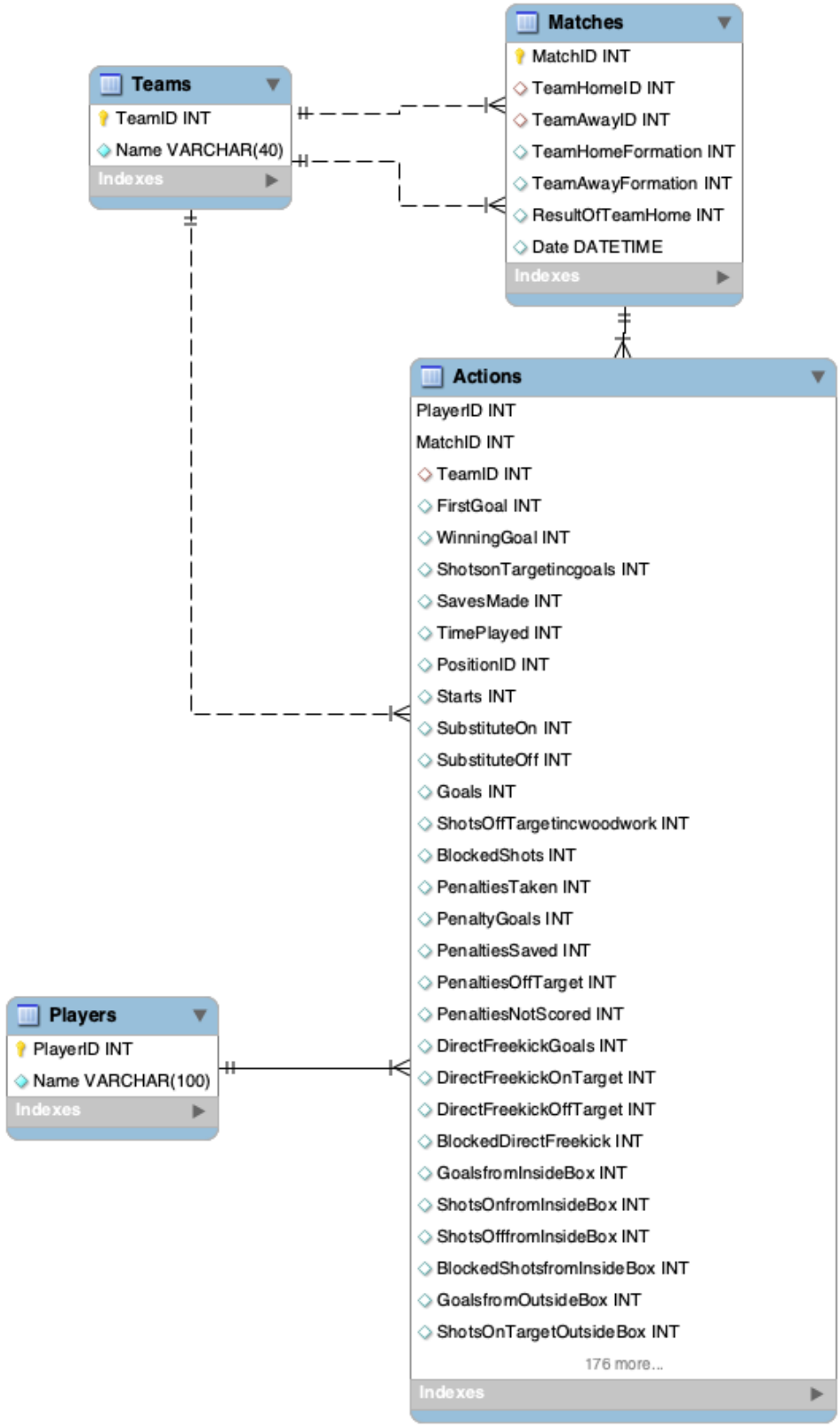
PUNTO 2

a. Permettere all'utente di inserire un numero di giocatori ***k*** (**Suggerimento**: usare valori di K piccoli, < 5).
b. Alla pressione del bottone "Dream-Team", si individui, tramite un algoritmo ricorsivo, un insieme di ***k*** giocatori (il "dream-team"), facenti parte di quelli selezionati nel punto precedente, che massimizzi il *grado di titolarità* del dream-team stesso. Il *grado di titolarità* di ogni singolo giocatore, in particolare, è dato dalla differenza tra il peso dei suoi archi uscenti (i minuti che ha giocato in più dei suoi avversari) ed il peso degli archi entranti (i minuti che ha giocato in meno).
   N.B: i giocatori che entrano a far parte del dream-team possono far parte di squadre diverse. La tabella "teams" non ha nulla a che vedere con il dream-team.
c. Quando un giocatore entra a far parte del dream-team, tutti i giocatori che quest'ultimo ha "battuto", in termini di minuti giocati, durante la stagione non possono più essere aggiunti al dream-team.
d. Si stampi il "dream-team" ottenuto e il suo *grado di titolarità* (definito come la somma dei gradi di titolarità dei giocatori che ne fanno parte).

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.

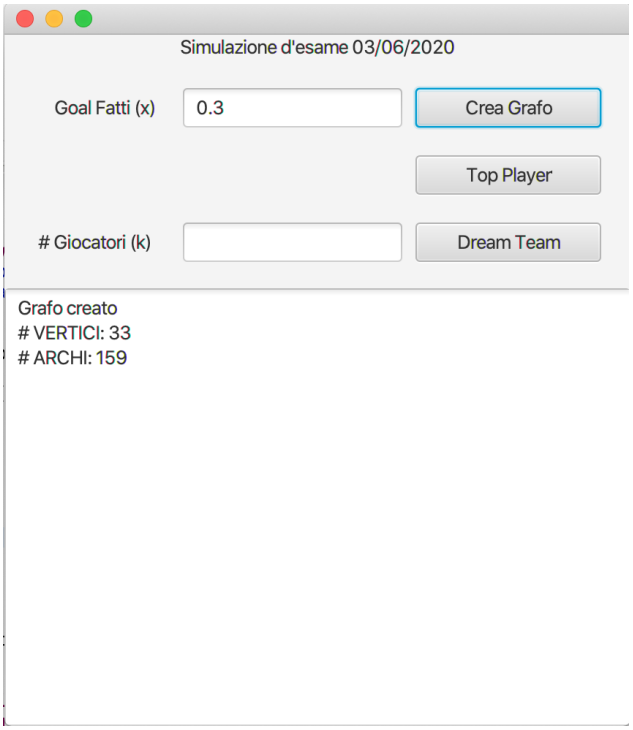
Tutti i possibili errori di immissione, validazione dati, accesso al database, ed algoritmici devono essere gestiti, non sono ammesse eccezioni generate dal programma. Nell'ultima pagina sono disponibili due esempi di risultati per controllare la propria soluzione.

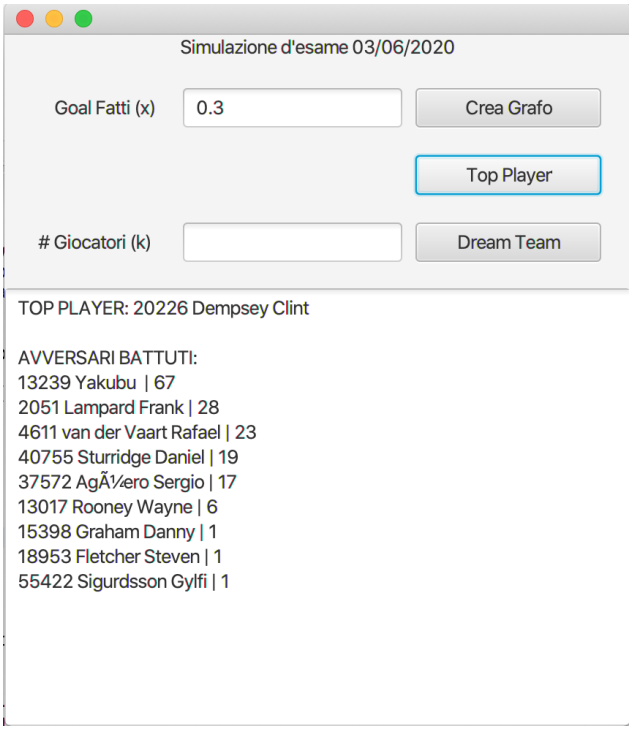
Le tabelle **Teams**, **Players** e **Matches** contengono rispettivamente informazioni su squadre, giocatori e partite. Ogni riga della tabella **Actions**, invece, contiene le informazioni statistiche di un determinato giocatore in una determinata partita (minuti giocati, goal fatti, ecc.)



ESEMPI DI RISULTATI PER CONTROLLARE LA PROPRIA SOLUZIONE:

x = 0.3





x = 0.5