Corso di Algoritmi e Strutture Dati 2022/2023

Laboratorio individuale:

Realizzazione di un programma per gestire informazioni relative a film

Obiettivo

L'obiettivo del laboratorio individuale è quello di creare una struttura dati in grado di memorizzare e fornire informazioni relative ad una semplice database di film, contenenti insiemi di attori, registi, produttori e film e relazioni quali "X ha recitato nel film Y", oppure "X ha diretto il film Y" o infine "X ha prodotto il film Y".

Vi vengono forniti quattro file

- main.cpp, per facilitare la fase di test
- movies.h, che stabilisce l'interfaccia a cui devono aderire i metodi che sviluppate
- file di input contenente la lista dei nodi
- file di input contenente la lista delle relazioni

Come nel laboratorio precedente, è richiesta una struttura dati list::List (lista di stringhe) che possa contenere gli output di alcuni dei metodi che implementerete. È ragionevole aspettarsi che la vostra consegna richieda la creazione di ALMENO tre nuovi file: list.h, list.cpp e movies.cpp. Sta a voi suddividere il codice in questi file secondo quanto avete imparato finora.

A voi completa libertà nel progettare una struttura dati che soddisfi i requisiti indicati nel seguito, che sia efficiente ed intelligente, in grado di memorizzare le informazioni date e fornire risposte ad interrogazioni del tipo "Quali attori hanno recitato in un qualsiasi film insieme a X" e similari, come di seguito specificato.

Dati

I dati derivano dal database Movie Graph utilizzato come esempio nel graph DB Neo4j.

In particolare, esistono solo 2 tipologie di nodi: persona e film.

Il file nodi.txt contiene, in ogni riga, il nome ed il tipo di un nodo. Ad esempio

```
Keanu Reeves:persona

When Harry Met Sally:film
```

Una persona può apparire sia come un attore che come regista che come produttore.

I nodi sono collegati tra loro da una serie di relazioni che rappresentano, ad esempio, se un attore ha recitato in un film o se la persona è stato il regista.

Queste informazioni sono contenute nel file archi.txt, una relazione per riga, nel formato persona:film:relazione. Ad esempio:

```
Keanu Reeves:The Matrix Reloaded:recitato_in
Lilly Wachowski:Cloud Atlas:diretto
Rob Reiner:When Harry Met Sally:prodotto
```

I due file forniti contengono, rispettivamente, 171 nodi e 151 archi.

Requisiti

Non esistono relazioni tra due nodi dello stesso tipo, e.g. persona-persona, ma solo persona-film.

Le relazioni sono tutte bidirezionali: ad esempio se un attore X ha interpretato il film Y, il film Y è stato interpretato dall'attore X.

Non è garantito che un film abbia un solo produttore o un solo regista.

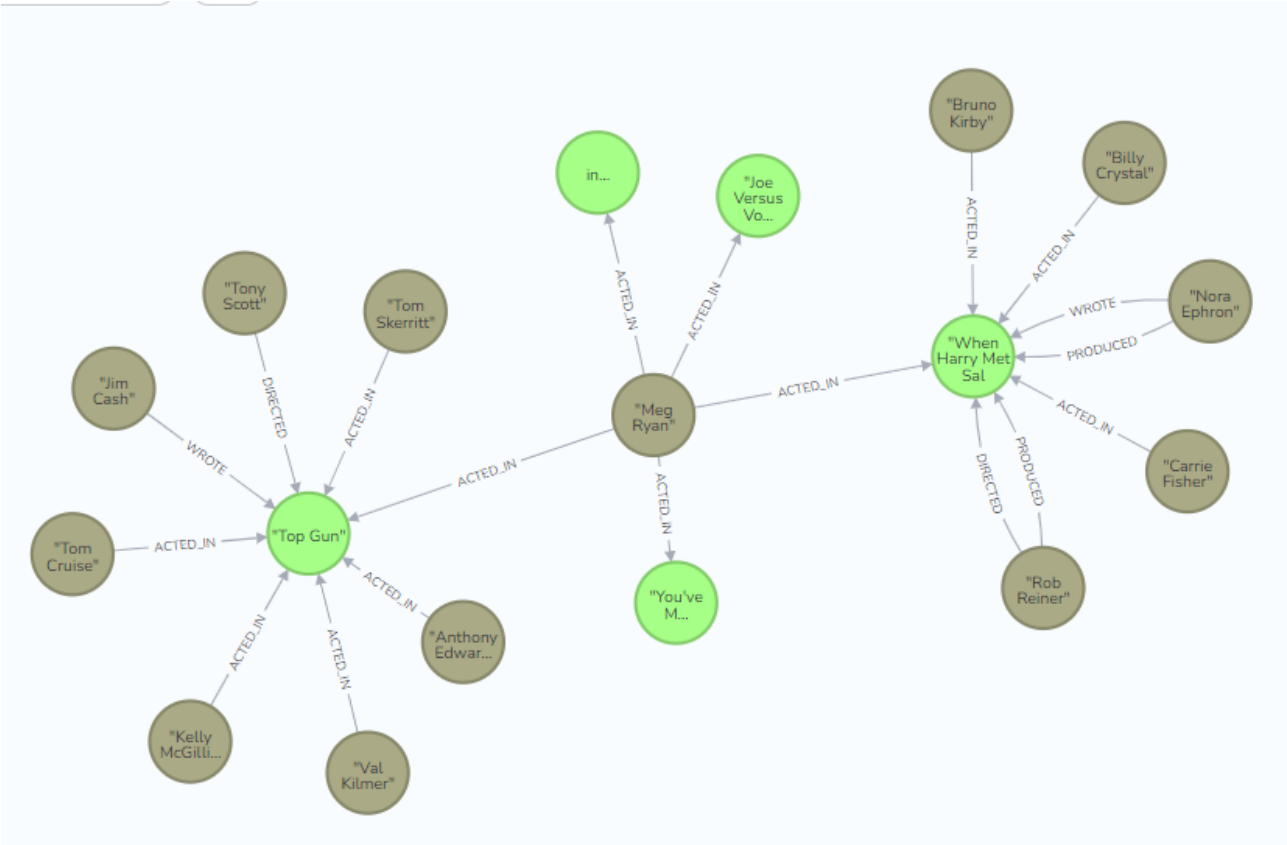
Una persona potrebbe essere sia attore che regista che priduttore.

Esempi

Nella figura seguente un esempio relativo al film "When Harry Met Sally" tratto dal database originale.

Nel nostro caso consideriamo solo le relazioni "recitato_in", "diretto", "prodotto".

Nella figura successive una vista dei dati incentrati sull'attrice "Meg Ryan". Lei ha interpretato 5 film, di due dei quali sono mostrate le altre persone coinvolte.



Se volessimo chiedere "restituisci tutti gli attori che hanno interpretato un film insieme a Meg Ryan e quale", otterremo una lista del tipo

Sleepless in Seattle,Victor Garber
Sleepless in Seattle,Tom Hanks
Sleepless in Seattle,Bill Pullman
Sleepless in Seattle,Rita Wilson
Sleepless in Seattle,Rosie O'Donnell
You've Got Mail,Tom Hanks
You've Got Mail,Parker Posey
You've Got Mail,Greg Kinnear
You've Got Mail,Steve Zahn
You've Got Mail,Dave Chappelle
Top Gun,Val Kilmer
Top Gun,Tom Skerritt
Top Gun,Kelly McGillis
Top Gun,Tom Cruise
Top Gun,Anthony Edwards
When Harry Met Sally,Carrie Fisher

When Harry Met Sally,Billy Crystal
When Harry Met Sally,Bruno Kirby
Joe Versus the Volcano,Tom Hanks
Joe Versus the Volcano,Nathan Lane
In grassetto potete vedere la corrispondenza tra la figura ed i dati. È stato chiesto chi ha recitato, per cui sono state considerate solo le relazioni “acted_in” dei soli film in cui uno degli attori era “Meg Ryan”.

Nota: come vedete nel seguito, questa query è puramente illustrativa e non fa parte delle operazioni richieste.

Operazioni richieste

a. leggi un database (2 operandi: nome_file_nodi nome_file_archi)
b. aggiungi un nodo corrispondente a una persona (1 operando: nome)
c. aggiungi un nodo corrispondente a un film (1 operando: nome)
d. stampa il contenuto del database - in modo sensato
e. numero di persone
f. numero di film
g. numero di relazioni "ha recitato in"
h. numero di relazioni "ha diretto"
i. numero di relazioni "ha prodotto"
j. stampa gli attori che hanno recitato in almeno un film con un attore (1 operando: nome)
k. stampa gli attori che hanno recitato in film con un dato produttore (1 operando: nome)
l. stampa gli attori che hanno recitato in film con un dato regista (1 operando: nome)
m. numero di Bacon di un attore (1 operando: nome)
n. aggiungi un nodo corrispondente a una relazione (3 operandi separati da ':', persona:film:relazione, come nei file di input)

Numero di Bacon (https://it.wikipedia.org/wiki/Kevin_Bacon#Il_numero_di_Bacon): numero di "gradi di separazione" tra un attore e Kevin Bacon (KB), dove KB stesso ha numero di Bacon 0, un attore che ha recitato con KB ha come numero di Bacon 1 e in generale un attore X ha come numero di Bacon 1 + (minimo tra i numeri di Bacon degli attori che hanno recitato in un film in cui ha recitato X).

Consegna

Un file zip contenente

- Tutti i file necessari a compilare il programma
- Una relazione (formato txt, doc o pdf) in cui viene descritta e discussa sia la struttura dati che la sua complessità e la strategia seguita per il calcolo del numero di Bacon. Includete nella relazione anche il comando usato per compilare il vostro codice.
- Non è consigliato, ma è permesso, cambiare i file main.cpp e movies.h forniti. Il comportamento del programma risultante deve comunque essere uguale a quello specificato da main.cpp. Se cambiate questi file, è necessario includere una spiegazione nella relazione.

Discusso vuol dire “Ho scelto di rappresentare i nodi come elementi di un vettore **perché**…”.

Punti e Valutazione

Il punteggio massimo di quattro punti si raggiunge non solo se le funzioni e le strutture dati sono corrette, ma anche se sono implementate in modo efficiente e non ci sono problemi “di stile”.

In particolare, sono elementi apprezzati ai fini della valutazione (l’elenco non è esaustivo):

- corretta indentazione del codice,
- identificatori significativi (guai a riempirsi di variabili aux)
- introduzione di funzioni ausiliarie quando appropriato
- commenti significativi (non devono essere "traduzioni" del codice in lingua corrente o frasi equivalenti a qui c’è scritto che qui c’è scritto solo questo, ma spiegazioni ad alto livello dell'algoritmo eseguito e del motivo per cui avete fatto le vostre scelte).

La consegna è obbligatoria solo per coloro che hanno sottoscritto il patto.

Questa volta il termine del 19 giugno 23:55 è categorico.