Tema 3 - Internet Movie Database

Responsabili:

- Armand Nicolicioiu [mailto:armand.nicolicioiu@gmail.com]
- Andrei Petre [mailto:p31andrei+sd@gmail.com]

Data publicarii: 1 mai, ora: 23:55

Deadline: 23 mai, ora 23:55

Modificări și actualizări

- **3 mai**
  - adaugare exemplu de input/output
  - adaugat schelet de cod cu un singur test, cel din input/output
- **6 mai**
  - actualizat deadline
  - adăugat noi răspunsuri în FAQ
  - scos bonus sistem recomandari

Obiective

În urma realizării acestei teme:

- veti invata cum sa adaptati structuri de date cunoscute la cerinte mai complicate
- veti invata sa apreciati diferentele de timp si memorie consumata intre diverse structuri de date
- veti exersa lucrul colaborativ in echipe de cate doi
- veti putea lucra cu elemente colaborative pentru usurarea muncii (e.g git pentru versionare de cod)
- veti putea folosi diverse structuri de date cu arbori pentru a rezolva problemele eficient
- veti putea invata cate ceva despre cum functioneaza un sistem de recomandari clasic

Intro

Internet Movie Database (IMDb) este o bază de date online ce conține informații legate de filme, programe de televiziune și jocuri video. A fost lansat în 1990 de către programatorul Col Needham, iar în 1998 a devenit o filială a Amazon Inc.

În prezent numără 4.2 milioane de titluri și 7.8 milioane de personalități. De asemenea, are 75 de milioane de utilizatori înregistrați.

Pe vară, am reusit sa obtinem un internship pe backend pentru IMDb. Avem acces la baza lor de date din toate timpurile (omg, ce onoare). Scopul acestui internship este să construim un set de structuri de date eficiente si să le refolosim cat mai mult posibil – codul nostru să fie generic și reutulizabil de alți colegi/interni în viitor. Comentariile ne vor ajuta atata pe noi cât și pe cei care vor citi codul nostru după ce ne-am terminat internship-ul (retineti că un cod e citit de mult mai multe ori decât e scris – inclusiv de către voi, dacă reveniți peste 2-3 luni să recitiți ce ați scris).

Datele din IMDb la care avem acces și sunt relevante pentru noi sunt:

- colecția de filme
- lista de utilizatori și rating-urile oferite de aceștia unor filme

Un film are următoarele informații:

- nume
- id (un string)
- data apariției (ca unix timestamp)

- regizorul filmului
- listă actori
- o listă de categorii

Un utilizator are urmatoarele informatii:

- nume
- id

Un actor are urmatoarele informatii:

- nume
- id

Pe langa aceste informatii care sunt deja indexate in baza de date, sistemul nostru poate primi noi informatii in timp real:

- un nou film tocmai a fost adaugat (cu toate metadatele de mai sus)
- un nou rating a fost adaugat/actualizat/sters de catre un utilizator unui anumit film (care trebuie sa existe)

Cerinta

Veți primi o succesiune de interogări la care va trebui să răspundeți cât mai rapid. De asemenea, va trebui să actualizați baza de date pe măsură ce se primesc informații noi (pentru punctaj maxim va fi nevoie de un mod eficient). Query-urile si actualizările pot și vor fi intercalate! Beware!

Folisiti scheletul de cod pentru a incepe. Modificarile voastre trebuie sa fie in alte fisiere `.h` aferente, nu in main.cpp sau in Makefile. Daca vreti sa faceti compilarea cu mai putine warning-uri pana terminati tema, puteti sa scoateti flag-ul `-Wextra` temporar, insa pe vmchecker vom compila cu toate flag-urile de warning-uri (folosind Makefile-ul primit in schelet).

Operatii de adaugare/modificare

- `add_movie` are semnatura un pic mai lunga:

```
void add_movie(string movie_name, string movie_id, int timestamp, vector<string> categories,
               string director_name, vector<string> actors_ids)
```

- `void add_actor(string actor_id, string name)`
- `void add_user(string user_id, string name)`
- `void add_rating(string user_id, string movie_id, int rating)`
  - Un user poate adauga cel mult un rating pentru un film.
- `void update_rating(string user_id, string movie_id, int rating)`
- `void remove_rating(string user_id, string movie_id)`

Operatii de interogare

- `string get_rating(string movie_id)`
  - Intoarce rating-ul unui film, reprezentand media tuturor rating-urilor date de utilizatori pentru filmul respectiv.
- `string get_longest_career_actor()`
  - Intoarce id-ul actorului cu cea mai lunga activitate (diferenta dintre data de aparitie a celui mai recent film in care a jucat si data de aparitie a primului film). In caz de egalitate, se va intoarce primul ordonat crescator dupa id.
  - Daca un actor a jucat in exact un film, atunci vom considera activitatea sa ca fiind 0. Daca toti actorii au activitate de lungime 0, se aplica prioritatea de mai sus in care il vom intoarce pe primul ordonat crescator dupa id.

    - Daca nu aveti niciun actor care sa fi jucat in vreun film, intoarceti `none`.
- `string get_most_influential_director()`
    - Intoarce numele regizorului care a lucrat cu cei mai multi actori. In caz de egalitate, se va intoarce primul ordonat crescator dupa nume.
- `string get_best_year_for_category(string category)`
    - Intoarce anul in care rating-ul mediu al filmelor din categoria ceruta este maxim. In caz de egalitate, se va intoarce anul minim.
- `string get_2nd_degree_colleagues(string actor_id)`
    - Un actor Y este o legatura de gradul 1 cu X doar daca au jucat impreuna in cel putin un film. Legatura de gradul 2 inseamna acei actori Z care au jucat impreuna cu un actor de gradul 1 al lui X dar nu cu X direct.
    - Intoarce o lista de actori ordonati crescator dupa id separati prin spatiu
- `string get_top_k_most_recent_movies(int k)`
    - Intoarce un string de forma `movie_id1 movie_id2 … movie_idk`
    - Ordonarea se face descrescator dupa timestamp aparitie
- `string get_top_k_actor_pairs(int k)`
    - Actorii care au jucat impreuna in cele mai multe filme (perechi de cate doi)
    - Intoarce un triplet de forma `(actor_id1, actor_id2, nr_filme)` delimitate prin spatiu, sortate dupa `nr_filme`. In caz de egalitate, se va sorta dupa `actor_id1` si in caz de egalitate din nou, dupa `actor_id2`. Totodata, `actor_id1 < actor_id2`.
- `string get_top_k_partners_for_actor(int k, string actor_id)`
    - Top K actori care au colaborat cel mai mult cu un actor dat (dupa numarul de filme).
    - Intoarce o lista de actori ordonati descrescator dupa numarul de filme (si crescator dupa `actor_id` in caz de egalitate) separati prin spatiu
- `string get_top_k_most_popular_movies(int k)`
    - Cele mai populare k filme. (popular = cate rating-uri a primit)
    - Intoarce o lista de id-uri ale filmelor ordonate crescator dupa numar (si, in caz de egalitate dupa id film) delimitate prin spatiu
    - Daca filmul nu a primit niciun rating, atunci numarul de rating-uri primite este 0, nu `none`. Doar rating-ul este `none`.
- `string get_avg_rating_in_range (int timestampStart, int timestampEnd)`
    - Media finala se face pe baza rating-urilor filmelor deja aflate (nerotunjite) ce corespund intervalului `[timestampStart, timestampEnd]`.
    - Returneaza un singur string corespunzator unui numar real cu doua zecimale
    - Folositi `double` pentru reprezentarea rating-ului pentru a fi siguri ca obtineti acelasi rezultat la acest task

Observatii generale pentru toate interogarile:

- daca raspunsul nu exista, atunci se va intoarce string-ul `none`
- pentru afisarea numerelor sub forma de string se va folosi rotunjire, deci 9.33511 se va intoarce intr-un `std::string` ca 9.34

Date de intrare

Scheletul se ocupa de citirea tuturor datalor si paseaza parametrii corespunzatori fiecarei functii.

Fisiere de intrare pot fi si inspectate manual, fiecare linie reprezentand o singura operatie. Primul string al fiecarei linii reprezinta tipul operatiei.

Date de iesire

Toate output-urile interogarilor sunt deja preluate in schelet si scrise intr-un fisier. Tot ce trebuie sa faceti e ca la fiecare operatie sa generati un string pe baza rezultatui.

De exemplu, functia `get_top_k_most_popular_movies(int k)` va returna o lista de id-uri concatenate prin spatiu: "`id1 id2 id3 .. idk`"

Teste

Toate testele vor fi publice! Punctajul de pe vmchecker va fi cel luat in calcul (pentru ca timpii de rulare sunt calibrati pentru server, nu pentru masinile voastre locale) dar corectitunea poate fi verificata si local.

Punctaj

- 80 puncte obținute pe testele de pe vmchecker. Condiții pentru obținerea punctajului total:
  - fără memleak-uri
  - fără erori de valgrind
- 20 puncte: README
- **Bonus 20% din punctajul obținut** pentru coding style
  - spre exemplu: cu 60p din 100p și coding style perfect, obțineți 60*1.2 = 72p

Nu copiați! Toate soluțiile vor fi verificate folosind o unealtă de detectare a plagiatului. În cazul detectării unui astfel de caz, atât plagiatorul cât și autorul original (nu contează cine care e) vor primi punctaj 0 pe **toate temele**!

De aceea, vă sfătuim să nu vă lăsați rezolvări ale temelor pe calculatoare partajate (la laborator etc), pe mail/liste de discuții/grupuri etc.

Exemplu input si output rulare

Testul de mai jos contine 3 filme si cativa actori la fiecare film. Totodata, exemplifica ce output s-ar obtine pentru fiecare apel de functie din cele detaliate mai sus.

input file:

```
// Inputul nu o sa contina comentarii ; le includem aici pentru a evidentia query-urile mai usor
// add_movie(string movie_name, string movie_id, int timestamp, vector<string> categories,
//           string director_name, vector<string> actors_ids)
// observati ca argumentele sunt in ordine in fisier, la fel ca semnatura functiei.
// nm0000199,nm0000563,nm0714310 e un vector de string-uri reprezentand id-urile actorilor
// la fel e si la celelalte functii, argumentele in fisier sunt in ordinea in care sunt si in semnaturile functiilor
// corespunzatoare.
add_movie;Scent of a woman;tt0105323;725068800;drama;Martin Brest;nm0000199,nm0000563,nm0714310
add_user;ch0114428;Patrick Jane
add_user;ch0111140;Teresa Lisbon
add_user;ch0000033;Chewbacca
add_actor;nm0000199;Al Pacino
add_actor;nm0000563;Robin Tunney
add_actor;nm0048932;Simon Baker
add_actor;nm0714310;James Rebhorn
get_rating;tt0105323                          // 1
add_rating;ch0114428;tt0105323;9
add_rating;ch0111140;tt0105323;9
add_rating;ch0000033;tt0105323;2
update_rating;ch0000033;tt0105323;10
get_rating;tt0105323                          // 2
remove_rating;ch0000033;tt0105323
remove_rating;ch0111140;tt0105323
get_rating;tt0105323                          // 3
get_longest_career_actor                      // 4
// crime,drama e tot un vector de string-uri, reprezentand categoriile; la filmul anterior
// vectorul avea dimensiunea 1.
add_movie;The Godfather;tt0068646;83462400;crime,drama;Francis Ford Coppola;nm0000199,nm0000008
add_actor;nm0000008;Marlon Brando
get_longest_career_actor                      // 5
get_most_influential_director                 // 6
get_best_year_for_category;drama              // 7
```

```
add_rating;ch0111140;tt0068646;10
get_best_year_for_category;drama                // 8
get_best_year_for_category;comedy               // 9
get_2nd_degree_colleagues;nm0000199             // 10
add_movie;The Godfather: Part II;tt0071562;169344000;crime,drama;Francis Ford Coppola;nm0000199,nm0000380
add_actor;nm0000380;Robert Duvall
get_2nd_degree_colleagues;nm0000199             // 11
get_2nd_degree_colleagues;nm0000008             // 12
get_most_influential_director                   // 13
get_top_k_most_recent_movies;2                  // 14
get_top_k_actor_pairs;2                         // 15
get_top_k_partners_for_actor;2;nm0000380        // 16
get_top_k_partners_for_actor;2;nm0000199        // 17
get_top_k_most_popular_movies;2                 // 18
add_rating;ch0000033;tt0105323;3
get_top_k_most_popular_movies;4                 // 19
get_avg_rating_in_range;0;725068800             // 20
get_avg_rating_in_range;169344000;169344000     // 21
add_rating;ch0111140;tt0105323;10
add_rating;ch0111140;tt0071562;9
get_avg_rating_in_range;126403200;725068800     // 22
get_avg_rating_in_range;0;1000                  // 23
get_avg_rating_in_range;725068800;725068800     // 24
get_avg_rating_in_range;169344000;169344000     // 25
```

output file:

```
// Output-ul nu o sa contina comentarii; le includem aici pentru a evidentia query-urile mai usor
none                                            // 1
9.33                                            // 2
9.00                                            // 3
// Al Pacino este primul actor ordonat dupa id, asa ca il intoarcem pe el, fiindca toti actorii au activitate 0 momentan
nm0000199                                       // 4
// Acum Al Pacino are activitate mai mare decat 0; e singurul de altfel.
nm0000199                                       // 5
Martin Brest                                    // 6
// godfather n-are rating
1992                                            // 7
// godfather are rating mai mare acum pt aceasta categorie
1972                                            // 8
none                                            // 9
none                                            // 10
none                                            // 11
// Care sunt legaturile de grad 2 ale lui Marlon Brando?
// A jucat in Godfather 1 cu Al Pacino, astfel ca toti colegii cu care a jucat
// Al Pacino dar nu si Marlon Branco sunt de gradul 2.
nm0000380 nm0000563 nm0714310                   // 12
// E la egalitate cu Martin Brest ca numar de actori (trei) cu care a colaborat;
// dar e primul ordonat lexicografic dupa nume
Francis Ford Coppola                            // 13
tt0105323 tt0071562                             // 14
(nm0000008 nm0000199 1) (nm0000199 nm0000380 1)     // 15
nm0000199                                       // 16
nm0000008 nm0000380                             // 17
tt0068646 tt0105323                             // 18
tt0105323 tt0068646 tt0071562                   // 19
8.00                                            // 20
// The Godfather: Part II n-are niciun rating inca
none                                            // 21
// Scent of a woman are rating 7.3333333 (double precision)
// Godfather 2 are rating 9
8.17                                            // 22
none                                            // 23
7.33                                            // 24
9.00                                            // 25
```

FAQ

Q: Se poate folosi STL?

A: Se pot folosi urmatoarele structuri de date din STL: string, vector, map, list.

Q: Ce fac daca numarul de returnat e mai mic decat K intr-un query de tip Top K?

A: Returnezi cate sunt (mai putine decat K).

Q: Ambii membri ai echipei obtin acelasi punctaj?

A: Nu neaparat; cei doi membri ai echipei pot sa obtina punctaje diferite si trebuie sa ne convinga prin readme de contributia fiecaruia.

Q: E suficienta o singura submisie per pereche?

A: Da. Trebuie sa specificati in readme ambii membri ai echipei.

Q: Putem folosi algoritmi de sortare din STL?

A: Da, puteti folosi `std::sort` din `<algorithm>`.

Q: Putem folosi malloc și Free?

A: Nu. Folosiți new și delete, specifice c++.

Q: Are ceva dacă alocam memorie static?

A: Nu, atâta timp cât știți dimensiunea de la bun început și puteți face asta în condiții ok (nu se șterge zona de memorie). Daca însă vreți sa alocați static o dimensiune foarte mare (10mil sau ceva mare hardcodat) doar pentru a va trece testele, ne rezervam dreptul de a va scădea din punctaj.