Structuri de date (Seria CB)
Tema 2 - *Netflix*

Responsabili temă	Cosmin-Dumitru Oprea, Maria-Anca Băluțoiu, Ștefan Vodiță
Data publicării	**16.04.2021**
Termen predare	**09.05.2021 (ora 23:59)** Se acceptă teme trimise cu penalizare de 10 puncte/zi (din maxim 150 de puncte) până la data de **15.05.2021** (**ora 23:59)**
Versiune document	2

1. Introducere

Andrei este cinefil, dar cum cinematografele au fost închise, el și-a făcut abonament la Netflix și acum se uită la seriale. Vom gestiona serialele după cum urmează:
Există 4 categorii de seriale:

1. Tendințe
2. Documentare
3. Tutoriale pentru limbajul C
4. Top10 în România

Fiecare serial are un **ID** al categoriei din care face parte, un **nume unic**, un **rating**, și un **număr de sezoane**. Un serial este o **coadă** de unul sau mai multe sezoane, iar un sezon este o **coadă** de unul sau mai multe episoade.
Fiecare sezon are un **număr de episoade**, iar fiecare episod are o **durată** în minute.

2. Implementare

Pentru gestionarea serialelor vom folosi următoarele structuri de date:

2.1 Liste pentru gestionarea categoriilor de seriale

Cele 4 categorii de seriale vor fi gestionate folosind **liste generice simplu înlănțuite**.
Serialele din interiorul fiecărei liste vor fi ordonate descrescător după rating, iar în caz de rating egal, crescător după nume.
Serialele din Top10 sunt speciale, așa că ele vor avea pe lângă **nume**, **rating**, **ID**, **număr de sezoane** și un **număr de ordine** reprezentând poziția lor în Top10. În lista Top10 vor fi păstrate întotdeauna maxim 10 intrări.

2.2 Coada watch_later

În această coadă vor fi mutate seriale pe care Andrei dorește să le urmărească în viitor. Adăugarea unui serial în coada **watch_later** presupune eliminarea serialului din lista categoriei lui și inserarea în coada **watch_later**.

2.3 Stiva currently_watching

Această stivă gestionează serialele pe care Andrei a început să le vizioneze. Serialele din stivă vor fi păstrate ordonat, în vârful stivei aflându-se serialul la care Andrei s-a uitat cel mai recent.

2.4 Stiva history

Serialele vizionate integral vor fi mutate în stiva **history.** Serialele vor fi păstrate ordonat in stivă, ultimul serial vizionat integral va fi în vârful stivei.

3. Cerință

Va trebui să gestionați serialele lui Andrei pe baza unui set de comenzi citite dintr-un fișier.

3.1. Descrierea operațiilor și a datelor de intrare

A. Adăugare serial în baza de date

***Sintaxă*:** `add <ID> <nume> <rating> <#sezoane> <#epSez1> <durataEp1> <durataEp2>...<#epSez2> <durtaEp1> <durataEp2>`

Mod de funcționare: Adaugă un serial nou în categoria `1<=ID<=3`. Se va afișa un mesajul:

```
Serialul <nume> a fost adaugat la pozitia %d.\n
```

Exemplu: `add 3 numeA 8.9 2 3 10 8 9 2 7 6` */* se cere adăugarea unui serial în categoria 3, rating 8.9, serialul are 2 sezoane, primul sezon are 3 episoade de câte 10, 8, 9 minute, al doilea sezon are 2 episoade de câte 7, 6 minute*/*

```
Serialul numeA a fost adaugat la pozitia 1.\n
```

B. Adăugare sezon integral

***Sintaxă*:** `add_sez <nume> <#ep> <durataEp1> <durataEp2>`

Mod de funcționare: Adaugă un sezon nou pentru serialul cu numele `nume`. Se pot adăuga sezoane pentru orice serial care nu se află deja în `history`. Se va afișa un mesaj de forma:

```
Serialul <nume> are un sezon nou.\n
```

Exemplu: `add_sez numeA 3 10 8 9` */* se cere adăugarea unui sezon de 3 episoade pentru serialul numeA, episoadele au 10, 8, 9 minute*/*

```
Serialul numeA are un sezon nou.\n
```

C. Adăugare serial în Top10

***Sintaxă*:** `add_top <poz> <nume> <rating> <#sezoane> <#epSez1> <durataEp1> <durataEp2>...<#epSez2> <durataEp1> <durataEp2>`

Mod de funcționare: Adaugă un serial nou în categoria Top10. În Top10 vor fi păstrate întotdeauna maxim 10 seriale. Serialele care au ieșit din Top10 nu vor mai reveni în top. Se garantează că pozițiile din top sunt continue în momentul inserării. Se va afișa un mesaj de forma:

```
Categoria top10: [(<nume1>, <rating1>), (<nume2>, <rating2>)].\n
```

Exemplu: `add_top 2 numeB 8.0 1 3 10 11 12` */* se cere adăugarea unui serial în categoria top10, pe poziția 2, cu rating 8.0, serialul are 1 sezon cu 3 episoade de câte 10, 11, 12 minute (în top există deja un alt serial, care este primul) */*

```
Categoria top10: [(primul_in_top, 7.5), (numeB, 8.0)].\n
```

D. Adăugare serial în watch_later

***Sintaxă*:** `later <nume>`

Mod de funcționare: Mută un serial din cele 4 categorii în coada watch_later. Se va afișa un mesaj de forma:

```
Serialul <nume> se afla in coada de asteptare pe pozitia %d.\n
```

Exemplu: `later numeX` */* se cere mutarea serialului numeX în coada watch_later */*

```
Serialul numeX se afla in coada de asteptare pe pozitia 3.\n
```

E. Vizionare serial

***Sintaxă*:** `watch <nume> <durata>`

Mod de funcționare: Mută un serial în stiva **currently_watching** dacă nu există deja, apoi vizionează `durata` minute din serialul `nume`. Dacă întreg serialul are mai puțin de `durata` minute de vizionat, acesta va fi vizionat integral. Se va cere vizionarea unui serial care nu se află deja în `history`. Mutarea unui serial din Top10 presupune actualizarea (decrementarea) numărului de ordine pentru serialele aflate mai jos în top. Dacă serialul a fost vizionat integral, se va afișa mesajul următor:

```
Serialul <nume> a fost vizionat integral.\n
```

Exemplu: `watch numeX 100` */* se cere vizionarea serialului numeX timp de maxim 100 de minute */*

F. Afișare seriale

***Sintaxă*:** `show <X> ; X∈{ 1, 2, 3, top10, later, watching, history }`

Mod de funcționare: Afișează lista cu serialele din categoria X. Se va afișa un mesaj de forma:

```
Categoria <X>: [(<numeA>, <rA>), (<numeB>, <rB>)].\n
```

Exemplu: `show later` */* se cere afișarea serialelor din watch later */*

```
Categoria later: [(A, 9.7), (B, 8.5)].\n
```

***Exemplu*:** `show 1`

```
Categoria 1: [(A, 9.0), (B, 9.0)].
```

5. Restricții și precizări:

- programul va fi rulat astfel: **./tema2 in_file out_file**
- comenzile se citesc din fișierul ***in_file***, iar rezultatele se scriu în fișierul ***out_file***
- se garantează că datele de intrare vor fi corecte
- se garantează că nu va fi adaugat un sezon nou după ce Andrei a vizionat serialul integral
- stivele și cozile vor fi implementate folosind **liste generice simplu înlănțuite**
- numele serialelor vor fi alocate dinamic
- ratingurile vor avea maxim o zecimala, iar afisarea se va face intotdeauna cu o zecimala
- **nu** aveți voie să folosiți **variabile globale**
- `1 <= număr de seriale <= 2048`
- `1 <= număr de sezoane pe serial <= 10`
- `1 <= număr de episoade pe sezon <= 20`
- `1 <= durată episod <= 120`
- `1 <= lungime nume serial <= 32`

- `1.0 <= rating <= 10.0`
- este permisă folosirea funcțiilor de prelucrare a elementelor unei stive/cozi
- 30% din testele de intrare vor respecta următoarele condiții:
    - serialele au un singur sezon
    - vor exista doar comenzile A și F.

6. Exemple:

Intrare	Ieșire

`add 1 JAP 8.1 1 3 82 76 79`
`add 1 CAR 6.3 1 2 14 46`
`add 3 ITP 9.9 1 1 61`
`show 1`
`add 2 HAR 9.4 1 2 30 56`
`add 3 QUE 2.6 1 1 2`
`add 1 DIR 8.1 1 2 54 21`
`show 2`
`add 2 ARE 8.1 1 2 27 105`
`add 3 WAL 5.4 1 3 88 75 91`
`add 2 PUR 5.6 1 1 10`
`show 2`
`show 3`
	`Serialul JAP a fost adaugat la pozitia 1.`
	`Serialul CAR a fost adaugat la pozitia 2.`
	`Serialul ITP a fost adaugat la pozitia 1.`
	`Categoria 1: [(JAP, 8.1), (CAR, 6.3)].`
	`Serialul HAR a fost adaugat la pozitia 1.`
	`Serialul QUE a fost adaugat la pozitia 2.`
	`Serialul DIR a fost adaugat la pozitia 1.`
	`Categoria 2: [(HAR, 9.4)].`
	`Serialul ARE a fost adaugat la pozitia 2.`
	`Serialul WAL a fost adaugat la pozitia 2.`
	`Serialul PUR a fost adaugat la pozitia 3.`
	`Categoria 2: [(HAR, 9.4), (ARE, 8.1), (PUR, 5.6)].`
	`Categoria 3: [(ITP, 9.9), (WAL, 5.4), (QUE, 2.6)].`

Explicație:

1. Se adaugă serialul JAP în categoria Tendințe
2. Se adaugă serialul CAR în categoria Tendințe, după JAP
   CAR.rating = 6.3; JAP.rating = 8.1; CAR.rating < JAP.rating
3. Se adaugă serialul ITP în categoria Tutoriale
4. Se afișează categoria Tendințe și observam ordonarea JAP, CAR
5. Se adaugă serialul HAR în categoria Documentare
6. Se adaugă serialul QUE în categoria Tutoriale, după ITP
   QUE.rating = 2.6; ITP.rating = 9.9; QUE.rating < ITP.rating
7. Se adaugă serialul DIR în categoria Tendințe
8. Se afișează categoria Documentare, cu un singur serial, HAR
9. Se adaugă serialul ARE în categoria Documentare, după HAR
   ARE.rating = 8.1; HAR.rating = 9.4; ARE.rating < HAR.rating
10. Se adaugă serialul WAL în categoria Tutoriale, după ITP
    WAL.rating = 5.4; ITP.rating = 9.9; WAL.rating < ITP.rating
11. Se adaugă serialul PUR în categoria Documentare, după ARE
    PUR.rating = 5.6; ARE.rating = 8.1;PUR.rating < ARE.rating

Intrare	Ieșire

`add 1 AAA 10.0 2 2 10 10 2 10 10`
`add_sez AAA 2 10 10`
`show 1`
`later AAA`
`show later`
`watch AAA 40`
`show later`
`show watching`
`watch AAA 20`
`show watching`
	`Serialul AAA a fost adaugat la pozitia 1.`
	`Serialul AAA are un sezon nou.`
	`Categoria 1: [(AAA, 10.0)].`
	`Serialul AAA a fost adaugat in coada de asteptare pe pozitia 1.`
	`Categoria later: [(AAA, 10.0)].`
	`Categoria later: [].`
	`Categoria watching: [(AAA, 10.0)].`
	`Serialul AAA a fost vizionat integral.`
	`Categoria watching: [].`

`show history`	`Categoria history: [(AAA, 10.0)].`

Explicație:

1. Este creat serialul AAA, cu 40 de minute timp total de vizionare
2. Este adăugat încă un sezon, crește timpul total de vizionare la 60 de minute
3. Afisam categoria 1, din care face parte serialul
4. AAA este mutat in watch_later
5. Afisam watch_later
6. Se urmăresc primele 2 sezoane din serial
7. AAA nu mai apare în watch_later
8. Dar apare acum în currently_watching
9. Se urmărește ultimul sezon din serial
10. AAA nu mai apare în currently watching
11. Dar apare in history

Intrare	Ieşire

`add_top 1 AAA 8.5 1 1 60`
`add_top 2 BBB 9.0 1 1 60`
`add_top 3 CCC 8.2 1 1 60`
`show top10`
`later AAA`
`show later`
`show top10`
`add_top 1 DDD 9.3 1 1 60`
`show top10`
	`Categoria top10: [(AAA, 8.5)].`
	`Categoria top10: [(AAA, 8.5), (BBB, 9.0)].`
	`Categoria top10: [(AAA, 8.5), (BBB, 9.0), (CCC, 8.2)].`
	`Categoria top10: [(AAA, 8.5), (BBB, 9.0), (CCC, 8.2)].`
	`Serialul AAA a fost adaugat in coada de asteptare pe pozitia 1.`
	`Categoria later: [(AAA, 8.5)].`
	`Categoria top10: [(BBB, 9.0), (CCC, 8.2)].`
	`Categoria top10: [(DDD, 9.3), (BBB, 9.0), (CCC, 8.2)].`
	`Categoria top10: [(DDD, 8.5), (BBB, 9.0), (CCC, 8.2)].`

Explicație:

1. Se adaugă serialul AAA în categoria Top10 pe prima poziție
2. Se adaugă serialul BBB în categoria Top10 pe a doua poziție
3. Se adaugă serialul CCC în categoria Top10 pe a treia poziție
4. Se afișează categoria Top10 cu cele 3 seriale
5. Serialul AAA este mutat în coada **watch_later**
6. Se afișează **watch_later**, care contine serialul AAA
7. Se afișează categoria Top10, cu cele 2 seriale rămase, în ordine: BBB, CCC
8. Se adaugă serialul DDD în categoria Top10 pe prima poziție
9. Se afișează categoria Top10, cu cele 3 seriale rămase, în ordine: DDD, BBB, CCC

Intrare	Ieşire

`add 1 IR 4 2 3 3 1 19 1 10`
`show 1`
`add_top 1 EB 2.8 2 2 16 15 1 18`
`later IR`
	`Serialul IR a fost adaugat la pozitia 1.`
	`Categoria 1: [(IR, 4.0)].`
	`Categoria top10: [(EB, 2.8)].`
	`Serialul IR se afla in coada de asteptare pe pozitia 1.`

show top10 show later add_sez EB 3 3 23 23 watch IR 38 show later show watching show history watch EB 54 show 1 show top10 show watching show history	Categoria top10: [(EB, 2.8)]. Categoria later: [(IR, 4.0)]. Serialul EB are un sezon nou. Serialul IR a fost vizionat integral. Categoria later: []. Categoria watching: []. Categoria history: [(IR, 4.0)]. Categoria 1: []. Categoria top10: []. Categoria watching: [(EB, 2.8)]. Categoria history: [(IR, 4.0)].

7. Notare:

- **135 de puncte** obținute pe testele de pe vmchecker;
- **10 puncte**: coding style;
- **5 puncte**: README - va conține detaliile de implementare a temei, precum și **punctajul obținut la teste** (la rularea pe calculatorul propriu);
- **bonus: 20 de puncte** pentru soluțiile care nu au memory leak-uri (bonusul se acordă doar dacă testul a trecut cu succes);
- temele care nu compilează, nu rulează sau obțin punctaj 0 pe vmchecker, vor primi punctaj 0;
- se depunctează pentru:
    - warninguri la compilare (trebuie compilat cu -Wall);
    - linii mai lungi de 80 de caractere;
    - folosirea incorectă de pointeri;
    - lipsa verificărilor codurilor de eroare întoarse de funcții;
    - fiecare tip de structură de date folosit trebuie declarat in propriul fișier. Ex: stiva.h, coada.h, lista.h. Altfel se va aplica o penalizare de **10p**.
    - alte situații nespecificate aici, dar considerate inadecvate;

8. Reguli de trimitere a temelor:

- temele vor trebui încărcate atât pe vmchecker (în secțiunea **Structuri de Date Seria CB**) cât și pe *curs.upb.ro*, în secțiunea aferentă temei 2.
- arhiva cu rezolvarea temei trebuie să fie **.zip** și să conțină:
- fișiere sursă (fiecare fișier sursă va trebui sa înceapă cu un comentariu de forma:

  / NUME Prenume - grupa */*

- fișier **README**, denumit obligatoriu astfel, care să conțină detalii despre implementarea temei
- fișier **Makefile**, denumit obligatoriu astfel, cu două reguli:
    - **build**, care va compila sursele și va obține executabilul cu numele **tema2**
    - **clean**, care va șterge executabilele și alte fișiere obiect generate
- arhiva trebuie să conțină doar fișierele sursă (inclusiv Makefile și README); **nu** se acceptă fișiere executabile sau obiect
- dacă arhiva nu respectă specificațiile de mai sus nu va fi acceptată la upload și tema nu va fi luată în considerare.

Referințe:

[1] https://developer.gnome.org/programming-guidelines/stable/c-coding-style.html.en

[2] https://curs.upb.ro/pluginfile.php/536981/mod_resource/content/2/Regulament.pdf