

UNIVERZITET „MEDITERAN“ PODGORICA  
FAKULTET ZA INFORMACIONE TEHNOLOGIJE, PODGORICA



STRUKTURE PODATAKA I ALGORITMI

Društvena mreža - Fimovi

Studenti:

Ajša Dacić

Br. indeksa: 01-23

Amer Hot

Br. indeksa: 17-23

Profesor:

dr. Amar Kapić

Podgorica, januar 2025.

## Sadržaj:

<b>1. Uvod.....</b>	<b>3</b>
<b>2. Opis problema .....</b>	<b>4</b>
<b>3. Opis Aplikacije .....</b>	<b>5</b>
3.1 Implementacija grafa.....	5
3.2 Opis metoda za dodavanje, modifikaciju i brisanje podataka iz grafa.....	7
3.3 Opis dodatnih metoda.....	9
<b>4. Zaključak .....</b>	<b>12</b>

# 1. Uvod

Ovaj dokument opisuje implementaciju sistema za socijalnu mrežu koja omogućava korisnicima preporuke filmova na osnovu njihovih lajkovanih filmova i povezanosti sa drugim korisnicima. Sistem je osmišljen kao graf čiji čvorovi predstavljaju korisnike, dok ivice predstavljaju njihove međusobne odnose i zajedničke filmove.

Za potrebe implementacije, kreirane su klase koje pokrivaju različite aspekte sistema, uključujući korisnike, prijateljstva, kao i samu strukturu mreže. Svaka klasa je detaljno opisana, zajedno sa metodama koje omogućavaju dodavanje, modifikaciju i prikaz podataka. Dodatno, implementirane su napredne funkcionalnosti poput predlaganja novih prijateljstava na osnovu zajedničkih filmova.

Dokumentacija je strukturirana tako da korak po korak objašnjava ključne dijelove sistema i način njihove implementacije, uz poseban fokus na logiku.

## 2. Opis problema

Zadatak koji smo dobili da riješimo glasi:

“Graf koji predstavlja 'prijatelje' na socijalnoj mreži za preporuku filmova.

Za svaku osobu zna se korisničko ime i koje je filmove lajkovala (samo nazivi filmova). Za svaki vezu zna se da li su dvije osobe prijatelji ili ne i koliko su istih filmova lajkovale.

a) Dodavanje, modifikovanje i brisanje podataka (sve mogućnosti analizirati i implementirati)

b) Metoda koja kao ulazni argument prima korisničko ime i predlaže osobi novo prijateljstvo

(osobu sa kojom ima najviše zajedničkih filmova, a nijesu prijatelji).

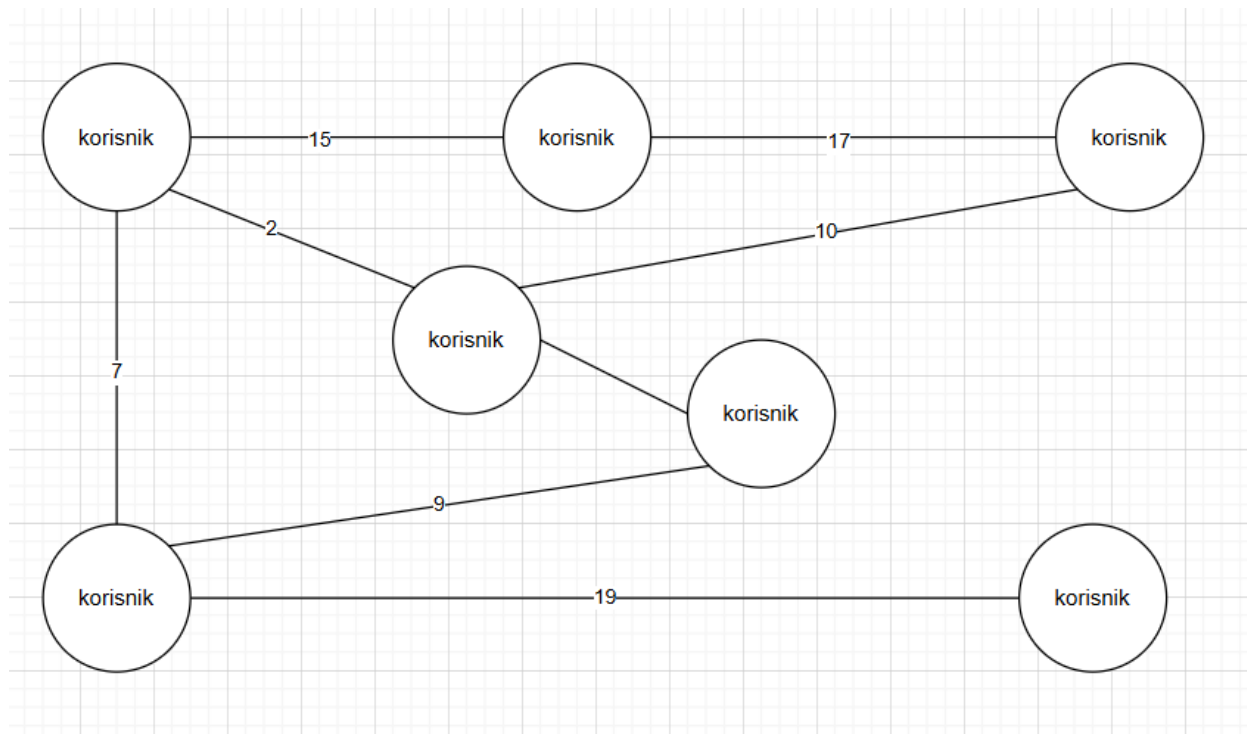
c) Metoda koja prikazuje sve parove prijatelja koji imaju broj zajedničkih filmova manji od prosječnog.”

Odlučili smo da ga riješimo korak po korak što je detaljno opisano u nastavku.

### 3. Opis Aplikacije

#### 3.1 Implementacija grafa

Na slici 1 nalazi se ideja Graf kojeg smo implementirali



Slika 1-prva ideja grafa

Graf predstavlja klasa LetterBoxd koja ima listu korisnika.

Svaki korisnik je čvor grafa (klasa Person), a povezan je Friendship objektom za ostale čvorove.

Friendship objekat predstavlja ivice grafa koje su dvosmjerne i imaju težinu.

Broj filmova koje su obje osobe lajkovale predstavlja težinu ivice.

Prema zadatku:

1. Za svaku osobu zna se korisničko ime i koje je filmove lajkovala (samo nazivi filmova).
  - Kreirana je klasa Osoba koja ima attribute: username (string), listu lajkovanih filmova (HashSet stringova - naziva filmova) i listu prijatelja (LinkedListu objekata klase Person).
2. Za svaku vezu zna se da li su dvije osobe prijatelji ili ne i koliko su istih filmova lajkovale.
  - Kreirana je klasa Friendship koja ima attribute: person1 i person2 (objekti klase Person) i broj zajedničkih lajkovanih filmova (int).

3. Graf koji predstavlja 'prijatelje' na društvenoj mreži za preporuku filmova.
- Kreirana je klasa LetterBoxd po uzoru na postojeći sajt koji ima ideju sličnu našem zadatku. U ovoj klasi atributi su: lista dostupnih filmova na aplikaciji (ArrayList) i lista svih korisnika (LinkedList). Ova klasa predstavlja osnovu grafa. Čvorovi našeg grafa su korisnici aplikacije, dok su veze ili ivice grafa predstavljene objektima Friendship koji povezuju korisnike jedne s drugima.

## 3.2 Opis metoda za dodavanje, modifikaciju i brisanje podataka iz grafa

Opis prema zadatku:

3-a) Dodavanje, modifikovanje i brisanje podataka (sve mogućnosti analizirati i implementirati):

Implementirane metode po klasama:

Person

- likeMovie - void metoda kojoj je ulazni argument naziv filma. Ona nad listom lajkovanih filmova poziva metodu add kojoj se proslijeđuje naziv filma.
- addFriend - void metoda s ulaznim argumentom osobe koja se dodaje. Kroz for-each petlju provjerava se da li prijateljstvo između osobe nad kojom je metoda pozvana i osobe koja se dodaje postoji s bilo koje strane. Ukoliko ne postoji, u listu prijatelja se dodaje novi objekat Friendship ove dvije osobe.
- removeFriend - void metoda kojoj se proslijeđuje Person. Poziva se removeIf u kojem se provjerava da li prijateljstvo postoji. Na kraju metode, iz liste prijatelja proslijeđene osobe uklanja se osoba nad kojom se metoda poziva.

LetterBoxd

- findUser - Metoda koja vraća objekat klase Person, a kao ulazni argument traži string username. Ova metoda ima for-each petlju koja prolazi kroz listu svih korisnika i upoređuje uneseni username. Ukoliko se username nađe, vraća ga kao string, a ako ne, vraća null.
- addUser - void metoda koja kao ulazni argument prima string username i kroz if statement ispituje da li metoda findUser vraća null. Ako je to slučaj, poziva se metoda add i kreira novi objekat. U suprotnom, ispisuje odgovarajuću poruku.
- addMovie - void metoda, ulazni argument je string koji predstavlja ime filma. Pošto se filmovi nalaze u ArrayList, moguće je pozvati metodu contains u if uslovu tako da, ako ta metoda vraća false, poziva se metoda add nad listom dostupnih filmova. Ukoliko je true, metoda vraća odgovarajuću poruku.
- displayMovies - void metoda koja preko for petlje ispisuje nazive dostupnih filmova.
- likeMovie - void metoda koja kao ulazni argument prima korisnika koji lajkuje film i indeks filma koji lajkuje. Ova metoda poziva metodu findUser. Ukoliko korisnik nije pronađen, vraća null i izbacuje odgovarajuću poruku. Provjerom da li uneseni indeks filma postoji, potvrđujemo da se film nalazi u listi i on se na kraju dodaje u listu lajkovanih filmova te osobe pozivanjem metode likeMovie iz klase Person.
- calculateCommonMovies - Metoda koja vraća int - broj zajedničkih filmova i kao ulazne argumente unose se imena dvije osobe koje želimo uporediti. Kreiramo skup stringova u čiji konstruktor unosimo lajkovane filmove prve osobe, zatim pozivamo metodu retainAll nad kreiranim skupom i proslijeđujemo listu lajkovanih filmova druge osobe. Ova metoda nam vraća skup koji predstavlja presjek ova dva skupa. Na kraju, u return unosimo set.size() kako bi metoda vraćala samo broj zajedničkih filmova kao int.

- `addFriendship` - void metoda koja kao ulazni argument prima dva stringa - korisnička imena dva korisnika. Prvo se pomoću metode `findUser` provjerava da li oba korisnika postoje, zatim se nad obje osobe poziva metoda `addFriend` u koju prosljeđujemo osobu jednu drugoj. Na kraju, ukoliko osobe ne postoje, program ispisuje odgovarajuću poruku.
- `removeFriendship` - Ova metoda radi na isti način kao i prethodna, osim što se umjesto `addFriend` nad osobama poziva `removeFriend`.

## Friendship

- `getCommonMovies` - Ova metoda vraća `int` i radi na isti način kao i metoda `calculateCommonMovies`, osim što direktno iz objekta `Friendship` uzima osobe koje se u njemu nalaze.



### 3.3 Opis dodatnih metoda

Opis prema zadatku:

3-b) Metoda koja kao ulazni argument prima korisničko ime i predlaže osobi novo prijateljstvo (osobu s kojom ima najviše zajedničkih filmova, a nisu prijatelji).

- Kreirana je `suggestNewFriendship` - void metoda koja prima string koji predstavlja korisničko ime korisnika kojem želimo predložiti prijatelja. Ponovo, pomoću metode `findUser`, provjerava se postoji li korisnik. Inicijalizuje se predloženi prijatelj kao objekat klase `Person` na `null` i `int` `maxCommonMovies` na `0`. Kroz `for-each` petlju prolazimo po listi korisnika, gdje novi objekat `Person` `other` upoređuje sve korisnike. Pomoću `if` uslova osiguravamo da se osoba ne poredi sama sa sobom i da se ne poredi s osobama koje su joj već prijatelji. Dalje, pozivanjem metode `calculateCommonMovies` za unesenu osobu i osobu s kojom se upoređuje dobijamo `int` koji označava broj njihovih zajedničkih filmova. U svakoj iteraciji, poređenjem s `maxCommonMovies`, određuje se je li to novi maksimum ili ne, i `suggestedFriend` se postavlja na `other` (osobu s kojom je maksimum filmova u tom trenutku). Poslije toga, van `for` petlje, provjerava se da li je predloženi prijatelj `null`. Ukoliko nije, program ispisuje odgovarajuću poruku, te predloženog prijatelja. U ostalim slučajevima ispisuje se poruka o grešci.

3-c) Metoda koja prikazuje sve parove prijatelja koji imaju broj zajedničkih filmova manji od prosječnog.

- Ovaj dio zadatka je riješen u klasi `LetterBox`, kao void metoda `displayFriendshipsBelowAverage`. Opis:
  - Prvo se inicijalizuju `int`-ovi `totalCommonMovies`, koji predstavlja ukupan broj zajedničkih filmova za sva prijateljstva, i `totalFriendships`, koji predstavlja ukupan broj prijateljstava, na `0`.
  - Zatim se inicijalizuje `Set` lista `allFriendships`, koja predstavlja sva prijateljstva bez duplikata (jer su prijateljstva obostrana), i `Set` lista `processedPairs`, koja predstavlja parove koji su obrađeni (ovom listom se ograničava dupliranje prijateljstava u `allFriendships`).
  - Kroz ugniježdenu `for-each` petlju se za svaku osobu provjeravaju njeni prijatelji. Ta prijateljstva se unose u listu svih prijateljstava, a parovi se unose u listu obrađenih parova (ovi parovi se identifikuju preko stringova koji se kreiraju pri svakoj iteraciji). Na ovaj način obezbjeđuje se da parovi prijatelja budu jedinstveni, odnosno jedan par se neće ponoviti kada `for` petlja bude obrađivala drugu osobu tog prijateljstva. To je bitno za računanje prosjeka, jer bi u suprotnom broj prijateljstava bio duplo veći. Na kraju, posljednja

provjera osigurava da, ukoliko ne postoje prijateljstva, aplikacija izbaci odgovarajuću poruku.

- Računa se prosjek zajedničkih filmova i for petljom, koja prolazi kroz sva prijateljstva, ispisuju se parovi čiji je broj zajedničkih filmova manji od prosjeka.

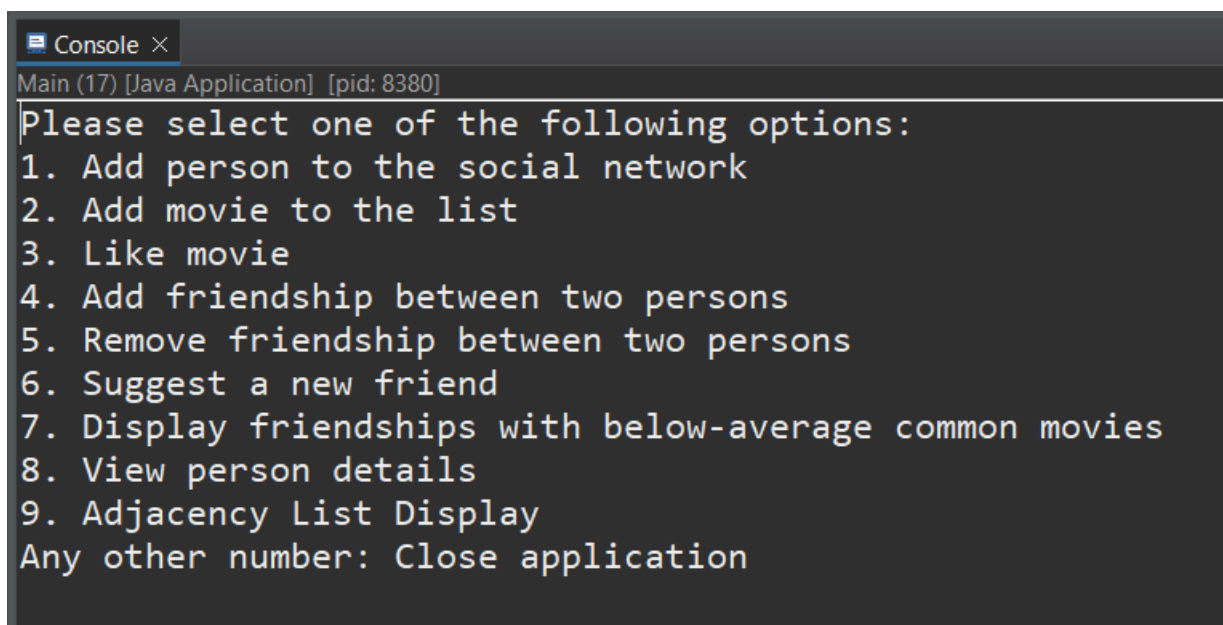
Dodatne metode i klase:

Metode toString su prilagođene svakoj klasi kako bi ispis bio tačan.

Metoda viewPersonDetails poziva toString metodu nad osobom koja se proslijedi kroz skener.

U klasi Menu se nalaze metode koje korisniku u konzoli ispisuju moguće opcije dok aplikacija radi, traže unos broja za određenu opciju i pomoću switch-a pozivaju odgovarajuću metodu.

U nastavku se nalaze slike menija i uotput-a u konzoli.

A screenshot of a Java application console window. The title bar says "Console x". Below the title bar, it says "Main (17) [Java Application] [pid: 8380]". The console output shows a menu of options: "Please select one of the following options:", followed by a numbered list from 1 to 9: "1. Add person to the social network", "2. Add movie to the list", "3. Like movie", "4. Add friendship between two persons", "5. Remove friendship between two persons", "6. Suggest a new friend", "7. Display friendships with below-average common movies", "8. View person details", "9. Adjacency List Display". At the end of the list, it says "Any other number: Close application".

```
Console x
Main (17) [Java Application] [pid: 8380]
Please select one of the following options:
1. Add person to the social network
2. Add movie to the list
3. Like movie
4. Add friendship between two persons
5. Remove friendship between two persons
6. Suggest a new friend
7. Display friendships with below-average common movies
8. View person details
9. Adjacency List Display
Any other number: Close application
```

Slika glavnog menija

```

3
Enter username:
amer
Available movies:
1. Munich - The Edge of War
2. Bridge of Spies
3. The Spy Who Came in from the Cold
4. The Imitation Game
5. Argo
6. Fury
7. Tinker Tailor Soldier Spy
Select a movie by number:
1
amer liked the movie: Munich - The Edge of War

```

Slika-Lajkovanje Filma

```

6. Fury
7. Tinker Tailor Soldier Spy
Select a movie by number:
1
ajsa liked the movie: Munich - The Edge of War
Please select one of the following options:
1. Add person to the social network
2. Add movie to the list
3. Like movie
4. Add friendship between two persons
5. Remove friendship between two persons
6. Suggest a new friend
7. Display friendships with below-average common movies
8. View person details
9. Adjacency List Display
Any other number: Close application

9
Displaying all friendships:
=====
amer - ajsa[1]
ajsa - amer[1]
bakir -
adrian -
=====

```

Slika-prikaz grafa

## **4. Zaključak**

Ovaj rad predstavlja rješenje problema kreiranja društvene mreže za preporuku filmova, koja omogućava korisnicima da lajkuju filmove, povezuju se s drugim korisnicima kroz prijateljstva i dobijaju preporuke na osnovu zajedničkih interesa. Kroz implementaciju klasa Person, Friendship i LetterBox, obuhvaćeni su svi ključni zahtjevi zadatka.