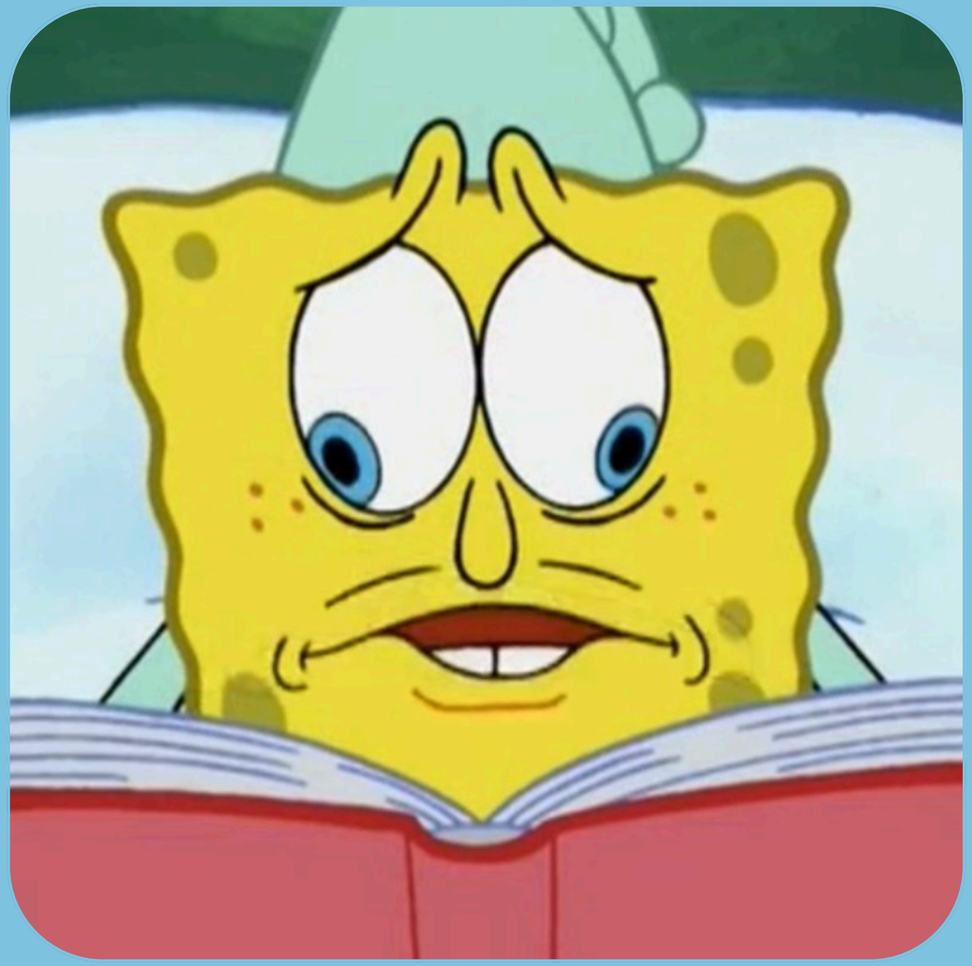
Uke 12 - Problemløsning

IN1000 gruppe 9
jonasbny@ifi.uio.no

Agenda



Eksamen høst 2024
Oppgave 4

Oppgave 4

Oppgave 4. Strømmetjeneste (50 poeng)

I denne oppgaven skal du lage en første versjon av en enkel *strømmetjeneste*. Strømmetjenesten tilbyr *abonnentene* en rekke ulike *serier*. Hver serie har én eller flere episoder.

For å gjøre det lettere for abonnenter å finne frem til serier de liker, bruker tjenesten tag-er. En *tag* sier noe om innholdet i en episode. Eksempler på tag-er er «komedie», «dokumentar» og «drama».

Hver episode kan ha flere tag-er. Til sammen beskriver tag-ene på alle episoder i serien hva slags innhold serien har, og hvor mye det er av ulike typer innhold.

Når det opprettes en ny abonnent, må abonnenten oppgi sine *preferanser.* For hver tag som finnes spør tjenesten om abonnenten liker serier med denne typen innhold, er nøytrale til innholdet, eller misliker innholdet.

Tjenesten kan dermed foreslå serier som passer for en abonnent ved å beregne en *match* mellom abonnentens preferanser og tag-ene på de ulike seriene.

Husk at du kan bruke klasser og metoder fra en tidligere deloppgave selv om du ikke har besvart den. Bruk navn på klasser og metoder som gitt i oppgaveteksten (navn i **bold**).

Oppgave 4a

4a) 6 poeng

Skriv klassen **Abonnent** med konstruktør. Konstruktøren har parametere for abonnentnavn (streng) og preferanser (ordbok beskrevet senere i oppgaven). Instansvariabler:

- abonnentnavn (streng)
- preferanser (ordboken som er parameter til konstruktøren)
- påbegynte serier (ordbok der serienavn er nøkkel, siste episodenummer abonnenten har sett er verdi)

Skriv også metodene:

- **hent_preferanser**. Returnerer ordboken med abonnentens preferanser.
- **sjekk_om_sett**. Metoden har en parameter serienavn (streng). Den returnerer True om abonnenten har sett en eller flere episoder av serien, ellers False.
- **se_en_episode** Metoden har en parameter serienavn, og legger til en ny serie eller oppdaterer episodenummer for serien hvis den allerede er påbegynt.

Oppgave 4b

4b) 4 poeng

Skriv klassen **Serie** med konstruktør. Konstruktøren har parameter serienavn. Den kaller på hjelpemetoden **_les_fra_fil** som leser episoder og tag-er fra fil (se neste deloppgave).
Instansvariabler:

- serienavn
- episoder i serien (ordbok med episodenummer som nøkkel, liste med tags som verdi)
- tag-er i serien (ordbok der tag er nøkkel, antall episoder med tag-en er verdi)

Oppgave 4c

4c) 10 poeng

Utvid klassen Serie med metodene:

- **_les_fra_fil**. Metoden har ingen parametere (annet enn self) eller returverdi. Den leser en fil med samme navn som serien etterfulgt av ".txt". Hver linje i denne filen inneholder tag-er for én episode, atskilt av mellomrom. På linje 1 ligger tag-er for episode 1, på linje 2 for episode 2, osv. Metoden legger inn hver episode med tag-er og oppdaterer hvilke tag-er som finnes for serien.
- **hent_serietags**. Metoden returnerer alle tag-er tilknyttet serien som en liste.

Oppgave 4d

4d) 10 poeng

Skriv klassen **Tjeneste** med konstruktør. Konstruktøren har en parameter for en liste av serienavn. Instansvariabler:

- seriene tjenesten tilbyr (ordbok der serienavn er nøkkel)
- tag-er i bruk i tjenesten (liste av strenger)
- abonnenter (ordbok der abonnentnavn er nøkkel, referanse til abonnent er verdi)

Konstruktøren oppretter et **Serie**-objekt for hvert av serienavnene og lagrer disse i ordboken med serier. Tag-ene i serien legges til i tjenesten dersom de ikke allerede er registrert.

Oppgave 4e

4e) 10 poeng

Skriv også følgende metode i klassen Tjeneste:

- **ny_abonnent**. Metoden har ingen parametere eller returverdi. Den ber om og leser abonnentens navn, og preferanser for hver enkelt tag i tjenesten fra terminal. Preferansene lagres i en ordbok der tag er nøkkel, verdi er -1 (*misliker*) eller 1 (*liker*). Om bruker gir annen preferanse for en tag enn -1, 0 eller 1 skal metoden be om ny verdi inntil bruker gir lovlig input. Tag-er abonnenten er nøytral til (0) tas ikke med i preferanser-ordboken. Ny abonnent med navn og preferanser opprettes og legges til tjenesten.

Oppgave 4f

4f) 5 poeng

Du skal nå utvide klassene **Serie** og **Tjeneste** med metoder som gjør det mulig å foreslå serier for en abonnent basert på abonnentens preferanser.

Utvid klassen **Serie** med metoden:

- **beregn_match**. Metoden har en parameter for en abonnents preferanser (samme som i Abonnent-klassen) og returnerer et heltall som angir hvor godt denne serien matcher disse. Hvis serien ikke har noen av tag-ene i preferansene, er match = 0. Hvis det er flest tag-er abonnenten *liker* skal match være et positivt tall, hvis det er flest tag-er abonnenten *misliker*, skal match være et negativt tall. Ta hensyn til hvor mange episoder tag-ene forekommer i.

Oppgave 4g

4g) 5 poeng

Utvid klassen **Tjeneste** med følgende metode:

- **foreslå_serier**. Metoden har en parameter som angir abonnentnavn, og beregner match mellom hver serie og abonnentens preferanser ved hjelp av metoden beregn_match. Metoden foreslår kun serier abonnenten ikke har sett før, og som har match > 0. Dersom ingen serier tilfredsstiller disse kravene skriver metoden ut beskjed om dette, ellers skriver den ut navnene på foreslåtte serier.

Oppgave 5b

Oppgave 5b

Skriv en funksjon ***adskilt*** som har en parameter ***tall_lister*** som tar inn en nøstet liste - en (ikke-tom) liste av (ikke-tomme) lister av tall. Funksjonen skal returnere True dersom det i den nøstede listen finnes to lister hvor det er slik at hvert av tallene i den ene lista er ekte større enn alle tallene i den andre lista.

Kallet *adskilt([[1,10], [6,8], [2,3]])* skal altså returnere True fordi hvert tall i den andre lista er større enn alle tall i den tredje.

Kallet *adskilt([[1,10], [6,8], [2,3,7]])* skal returnere False.