Sist endret: 26.03.2025

IN1000 Obligatorisk oppgave 7

Filmklubb

Frist for innlevering: 8. april kl. 23:59

Introduksjon

I denne oppgaven skal du starte utvikling av et program som holder rede på filmer for en filmklubb. I den versjonen du skal levere inn er det to klasser med hvert sitt testprogram. De ordene som er uthevet med **fet skrift** i oppgavene er enten Python-nøkkelord eller navn som skal brukes i det ferdige programmet. Filnavn oppgis i *kursiv*.

Sammen med dette dokumentet finner du en datafil (*film.txt)* og skjeletter til testprogrammer for hver klasse (*test_film.py* og *test_filmklubb.py*). Du skal fylle ut det som mangler i testprogrammene etter hvert som du skriver metodene i oppgave 2 og oppgave 3. De ferdige testprogrammene leveres sammen med klassene i de to oppgavene, og skal kjøre uten kjørefeil som gjør at programmet avslutter med feilmelding. Oppdager du logiske feil som du ikke klarer å rette, skal du kommentere disse øverst i testprogrammet.

I denne oppgaven skal du selv vurdere hvilke deler som trenger å kommenteres. Det bør være med en kort beskrivelse før hver metode (hva trenger en programmerer som skal bruke klassene å vite?), og en forklaring på spesielle løsninger eller valg som er gjort andre steder i programmene.

Hvilke filer som skal leveres for obligatorisk oppgave 7, er oppgitt for hver enkelt oppgave og oppsummert nederst i dette dokumentet.

Oppgave 1: Teorispørsmål

Leveres:

1. 1a) leveres i bildefil (*teori1a.png, teori1a.jpg eller teori1a.pdf*). Digital tegning eller håndtegning fotografert med mobil.
2. 1b) og 1c) besvares i samme fil, *teori.txt.*

Prøv deg gjerne på teorioppgavene før du har løst de andre oppgavene (men etter å ha *lest hele* denne teksten!) – det vil gjøre programmeringen enklere. Når du har programmert oppgave 2 og 3, kan du gå tilbake og eventuelt forbedre svarene dine.

Tegning av datastrukturer gir oss bedre forståelse av hva som skjer under utføring av et program, og er nyttig for å huske i detalj hvilke variabler, objekter og referanser vi opererer på. Dette er spesielt nyttig for å bygge opp en mental modell i starten av faget – men også når man skal arbeide med komplekse datastrukturer senere.

a) Lag en tegning av datastrukturen etter at følgende kode er utført:

```
from film import Film
from filmklubb import Filmklubb

klubb = Filmklubb()
klubb.les_filmer_fra_fil("filmer.txt")
funnet = klubb.finn_film_tittel("Hidden ")
klubb.legg_til_skuespillere(funnet)  # Bruker legger til 2 skuespillere
```

Du trenger ikke tegne mer enn to **Film**-objekter. Den ene filmen skal ha to skuespillere. Bruk notasjonen fra forelesningen i uke 8, med variabler som bokser, objekter som sirkler eller bokser med runde hjørner, og referanser som piler fra en variabel til et objekt. Du kan tegne strenger som innhold i enkle variabler, mens en liste eller en ordbok tegnes som et objekt.

b) Klassen **Filmklubb** kunne lest inn filen med filmer i konstruktøren, det vil si ved opprettelsen av et nytt **Filmklubb**-objekt. Ser du noen fordel ved å ikke lese inn filen med filmer i konstruktøren?

c) Parameteren **år** til konstruktøren i klassen **Film** er en **int** (heltall). Hvilken annen type ville det vært naturlig å velge, og hvilken forskjell ville det gjort?

Oppgave 2: Klassen **Film**

Leveres:

- Filen *film.py* med klassen **Film**.
- Ferdigskrevet testprogram for klassen i filen *test_film.py* (instruksjoner for dette ligger i den vedlagte filen).

Du skal skrive en klasse **Film** i filen *film.py.* Tabellen nedenfor angir klassens grensesnitt. Klassen har instansvariabler som representerer filmens tittel (**string**), året den er produsert (**int**) og skuespillere. Skuespillerne lagres i en ordbok (**dict)** der nøkkel er en skuespillers fulle navn (**string**) og verdi er rollen denne skuespilleren spiller i filmen (**string)**. Skuespillere kan legges til et **Film**-objekt etter at objektet er opprettet.

Du trenger ikke teste input fra terminalen eller fil for andre situasjoner enn det som er beskrevet i oppgaven, men koden din må håndtere ulike mulige returverdier fra metodekall. Ved sammenligning av titler skal du skille på små og store bokstaver.

For hver metode du skriver i klassen, skriver du test-kode for metoden i *test_film.py* som beskrevet under kommentaren med metodenavnet og kjører *test_film* på nytt. Da får du testet hver metode når den er skrevet og kan rette eventuelle feil før du går videre. **Viktig:** Ikke gå videre til oppgave 3 før du har testet at hver metode i klassen **Film** kjører uten feil!

Metode	**Parameter (type)**	**Returverdi** **Kommentar**

__init__	tittel (string) år (int)		- Konstruktør som lagrer filmens tittel og produksjonsår. **tittel** er en streng som består av en eller flere bokstavsekvenser atskilt av blanke tegn. Eksempler: "The Matrix" og «Hidden Figures». **år** er et heltall med 4 siffer.
hent_tittel	-	string	Returnerer filmens tittel.
ny_skuespiller	navn (string) rolle(string)		- Legger til i Film-objektet at en skuespiller spiller en bestemt rolle i filmen. Om en skuespiller forsøkes lagt til flere ganger i samme film, skal metoden skrive ut en feilmelding og returnere.

hent_skuespiller_na vn
 	
skriv_ut_film
	-
	
	-
		list av string	Returnerer en liste over skuespillerne i filmen. Hvert element er en string som inneholder navn på en av skuespillerne i filmen.
			
			- Skriver ut all informasjon om filmen i terminalen, f.eks. slik:
			
			`Hidden Figures(2016). Medvirkende:`
			`Taraji P. Henson som Katherine Johnson`
			`Octavia Spencer som Dorothy Vaughan`

sjekk_periode	år_1 (int) år_2 (int)	boolean	Returnerer **True** hvis filmen er produsert etter **år_1** og før **år_2** (perioden mellom **år_1** og **år_2**), ellers **False.**
sjekk_tittel	tittel_start (string)	boolean	Returnerer **True** hvis strengen **tittel_start** er nøyaktig lik begynnelsen på filmens tittel (den trenger ikke inneholde hele tittelen), ellers **False**. Hvis **tittel_start** er en tom streng, skal metoden returnere **True** for alle filmer. Hvis lengden av **tittel_start** er større enn lengden av filmens tittel, skal **False** returneres.
__str__ (gjennomgås i uke 9)	-	string	Returnerer en string med innholdet i **Film**-objektet. Formatet kan gjerne være det samme som vist for metoden **skriv_ut_film**.

__eq__ **Frivillig**	annen (Film)	boolean	Returnerer True hvis tittel og år er det samme i objektet den kalles på og i objektet som refereres av parameteren

Innlevering 6 med lenker til Trix-oppgaver gir god trening i det du trenger i oppgave 2. Se også Trix-oppgave 08.04 og 08.05.

Oppgave 3: Klassen **Filmklubb**

Leveres:

- Filen *filmklubb.py* med klassen **Filmklubb**
- Ferdigskrevet test-program for klassen i filen *test_filmklubb.py* (instruksjoner for dette ligger i den vedlagte filen).

I filen *filmklubb.py* skal du skrive konstruktør og metodene oppgitt i tabellen nedenfor. Klassen **Filmklubb** skal bruke metodene i **Film**-klassen, og må importere denne. Du skal ikke aksessere instansvariablene i **Film** direkte. Test hver metode etter hvert, med kode som beskrevet i *test_filmklubb.py*. Klassen **Filmklubb** skal tilby følgende metoder i grensesnittet:

Metode	**Parameter type)**	**Returverdi**	**Kommentar**
__init__	-	-	Definerer instansvariabelen **_filmer** som en foreløpig tom liste.
les_filmer_fra_fil	filnavn (string)	-	Metoden åpner en fil med navn angitt av parameteren **filnavn**. Filen inneholder en linje per film, på formatet
			tittel;produksjonsår
			Siden **tittel** kan inneholde blanke tegn, brukes semikolon som skilletegn. Produksjonsår er et firesifret årstall. Merk at det er lurt å fjerne tegn for linjeskift på slutten av hver linje før du deler den opp. Metoden skal opprette et nytt **Film**-objekt for hver linje, og legge disse inn i listen **_filmer**. Merk at filen ikke inneholder data om skuespillere.

skriv_ut_alle_filmer	-	-	Skriver ut all informasjon om alle filmer i terminalen, på formen vist under **skriv_ut_film** i klassen **Film**.

registrer_film	-	-	Ber om og leser inn tittel (string) og produksjonsår (4-sifret int) for en ny film fra terminalen, oppretter et **Film**-objekt for filmen og legger til dette i **_filmer**. Skuespillere registreres ikke i denne metoden. *Hvis __eq_ er implementert i Film*: Hvis film med samme tittel og år er registrert fra før skal metoden avslutte med en feilmelding.
finn_film_tittel	tittel (string)	Film eller None	Leter gjennom listen av filmer til den finner én som har tittelen angitt i parameteren, eller en tittel som begynner med nøyaktig de samme tegnene som i parameteren. Returnerer den første filmen den finner, eller **None** hvis ingen filmer starter med den oppgitte strengen.
legg_til_skuespillere	film (string)	-	Ber om og leser inn navn (string) og rolle (string) for en og en skuespiller fra terminalen, og legger disse til Film-objektet i parameteren film ved å kalle på metoden ny_skuespiller så lenge bruker oppgir nye skuespillernavn.
finn_filmer_periode	år_1 (int) år_2 (int)	list	Leter etter filmer som er produsert etter år_1 og før år_2. Returnerer en liste med Film-objekter som oppfyller kravene (listen kan være tom).

Syntes du oppgave 3 var vanskelig? Prøv Trix-oppgavene 09.03, 9.08 og 09.09.

Følgende filer skal leveres for obligatorisk innlevering 7:

- *teori1a.jpg (.png, .pdf)*
- *teori.txt*
- *film.py*
- *test_film.py*
- *filmklubb.py*
- *test_filmklubb.py*

- *filmer.txt*

Hvordan levere oppgaven

1. Kommenter på følgende spørsmål i kommentarfeltet i Devilry. Spørsmålene **skal** besvares.

a. Hvordan synes du innleveringen var? Hva var enkelt og hva var vanskelig?

b. Hvor lang tid (ca) brukte du på innleveringen?

Var det noen oppgaver du ikke fikk til? Hvis ja:

i. Hvilke(n) oppgave er det som ikke fungerer i innleveringen?

ii. Hvorfor tror du at oppgaven ikke fungerer?

iii. Hva ville du gjort for å få oppgaven til å fungere hvis du hadde mer tid?

iv. Hva vil du ha tilbakemelding på?

2. Logg inn på Devilry.

3. Lever alle filene som er nevnt under "Følgende filer skal leveres for obligatorisk innlevering 7", og husk å svare på spørsmålene i kommentarfeltet.

4. Husk å trykke på *Add delivery or question*. Sjekk deretter at innleveringen din er komplett. Du kan levere så mange ganger du vil frem til fristen. Det er ikke mulig å slette innleveringer i Devilry. *Alle* filer må være med hver gang du leverer, og filene må ha samme navn som oppgitt i oppgaveteksten. Retteren din vil gi tilbakemelding på kun den siste innleveringen, og derfor er det viktig at alle filer er med i siste innlevering.

5. De fleste vil ha behov for å løse mange flere oppgaver enn de du finner i denne innleveringen. Det får du anledning til i gruppetimene og i Trix. Du finner Trix-oppgaver for uke 8 her og for uke 9 her. Læreboken inneholder også mange oppgaver (noen går litt utenfor vårt pensum). Du finner dessuten mye på nett - og ikke minst kan du finne på egne programmer og utvidelser til oppgaver.