INF1001 - Obligatorisk innlevering 5

Frist: 3.10.16 12:00

Introduksjon

Du får her fem oppgaver du skal løse, hver oppgave teller ett poeng, unntatt oppgave 4 som teller to poeng. Oppgaven skal leveres på devilry, hvor hver oppgave skal leveres som en egen python-fil. I en oppgave med flere deloppgaver skal du kun levere en fil. Merk at det kun er ett forsøk på poenggivende obligatoriske innleveringer. Hvert program skal starte med en kommentar som forteller hva programmet skal gjøre. Kommenter koden videre underveis mens du jobber med den og lever kode med kommentarer. Les krav til innlevering lengre ned i dokumentet før du starter arbeidet med oppgavene.

Oppgave 1: Bil

Filnavn: *bil.py*
Du skal skrive en klasse `Bil` som skal modellere kjøringen av biler. En `Bil` har et merke, et registreringsnummer og en kilometerstand som viser hvor langt bilen har kjørt.

1. Skriv klassen `Bil` med en konstruktør (init-metode) som initierer de instansvariablene klassen trenger.
2. Skriv en metode `kjor(self, km)` som øker kilometerstanden.
3. Skriv en metode `hent_kilometerstand(self)` som returnerer bilens totale kilometerstand.
4. Skriv en metode `skriv_ut(self)` som skriver ut bilens merke, registreringsnummer og kilometerstand.
5. I denne deloppgaven skal du teste programmet ditt. Opprett et objekt av klassen `Bil` med et merke, et registreringsnummer og en kilometerstand. Øk kilometerstanden med 10 km, og sjekk at kilometerstanden ble oppdatert. Skriv til slutt ut alle variabler i bilen, med metoden `skriv_ut`.

Synes du denne oppgaven var vanskelig? Se øvingsoppgave 6.01 og 6.02.
Synes du denne oppgaven var lett? Se utfordringsoppgave 6.05.

Oppgave 2: Flere biler

Filnavn: *biler.py*
I denne oppgaven skal du ta utgangspunkt i klassen `Bil` fra forrige oppgave. Sammen med obligteksten ble det lagt ut en fil, biler.csv på følgende format:

```
Mercedes,NN123456,10000
Volvo,AB99999,3000
...
```

Denne fila inneholder informasjon om biler, en bil for hver linje. Hver linje består av tre variabler: et bilmerke, et registreringsnummer og en kilometerstand, separert av komma.

1. Opprett en tom liste med navn `biler`. Les inn fila, og for hver linje, opprett et objekt av klassen `Bil` med gitt merke, registreringsnummer og kilometerstand. Legg bilen inn i lista.
2. Skriv ut informasjon om alle bilene i lista. Hint: Her kan det være lurt å bruke en for-løkke og metoden `skriv_ut` i klassen `Bil`.
3. Alle bilene har deltatt i det samme landeveisløpet på 500 km. Du må derfor oppdatere kilometerstanden i alle bilene. Skriv ut informasjon om alle bilene for å sjekke om kilometerstanden ble oppdatert riktig.

Synes du denne oppgaven var vanskelig? Se øvingsoppgave 6.03.
Synes du denne oppgaven var lett? Se utfordringsoppgave 6.06.

Oppgave 3: Teorioppgave

Filnavn: *teori.txt*
Gi korte svar på spørsmålene under:

1. Hva er innkapsling? Hvorfor er det nyttig?
2. Hva er grensesnittet til en klasse? Hvordan skiller det seg fra implementasjonen av en klasse?
3. Hva er en instansmetode, og hvordan skiller dette seg fra prosedyrene/funksjonene vi har møtt hittil?

Oppgave 4: Isbod (gir 2 poeng)

Filnavn: *isbod.py*
Det er sommer, det er varmt, og folket vil ha is! Du skal lage et enkelt system som skal holde styr på ansatte for sjefen i en isbod. Å være ansatt i en isbod er meget sesongbetinget, og hvor mange ansatte som trengs varierer fra uke til uke. Vi antar her at de som har vært ansatt kortest (lavest ansiennitet), vil bli avskjediget først, dersom noen må gå.

1. Skriv en klasse `Isbod`. Denne skal inneholde to instansvariabler: En liste med strenger som skal inneholde navnene til de ansatte, og en heltallsvariabel med maksimalt antall ansatte. Lista skal aldri inneholde mer enn maksimalt antall ansatte.

2. Skriv en metode `ansett(self, navn)` i klassen som registrerer en ny ansatt. Navnet på den nyansatte skal lagres på siste plass i lista. Husk at du må sjekke om det blir flere ansatte enn maksimalt antall ansatte - da kan du ikke ansette flere!

3. Skriv en metode `gi_sistemann_sparken(self)` i klassen som gir den som sist ble ansatt sparken. Programmet skal da gi en utskrift med navnet på den som ble ansatt, fjerne vedkommende fra lista.

4. Skriv en metode `skriv_alle_ansatte(self)` i klassen som skriver ut alle ansatte i rekkefølge, der den første har lengst ansiennitet, og den siste har kortest. Til denne oppgaven skal du bruke en for-løkke.

5. I denne deloppgaven skal du teste programmet dit. Lag et `Isbod`-objekt. Ansett tre personer. Skriv ut alle ansatte, og sjekk at korrekte navn kommer ut. Spark så den siste, og skriv ut på nytt. Sjekk at utskriften nok en gang blir korrekt.

Du har nå implementert en konstruksjon som kalles en "stack". Dette er en datastruktur som brukes mye i "det virkelige programmeringsliv", og som du nok kommer til å jobbe mye med som programmerer!

Synes du denne oppgaven var lett? Se utfordringsoppgave 6.10.

Oppgave 5: Egen oppgave 5

Filnavn: *min_oppgave5.py*

1. Skriv oppgavetekst til en oppgave som handler om klasser og objekter. Skriv oppgaveteksten som kommentarer i Python-filen.

2. Løs oppgaven!

Du skal levere både oppgaveteksten og besvarelsen.

Krav til innleveringen

1. Oppgaven må kunne kjøres på IFI sine maskiner.

2. Kun .py-filene, teori.txt, README.txt og den umodifiserte filen biler.csv skal leveres inn.

3. Koden skal inneholde gode kommentarer som forklarer hva programmet gjør.

4. Programmet skal inneholde gode utskriftssetninger som gjør det enkelt for bruker å forstå.

Fremgangsmåte for innlevering i INF1001

1. Lage en fil som heter README.txt. Følgende spørsmål skal være besvart i filen:

   - Hvordan synes du innleveringen var? Hva var enkelt og hva var vanskelig?
   - Hvor lang tid (ca) brukte du på innleveringen?
   - Var det noen oppgaver du ikke fikk til? Hvis ja:
     - Hvilke(n) oppave er det som ikke fungerer i innleveringen?
     - Hvorfor tror du at oppgaven ikke fungerer?
     - Hva ville du gjort for å få oppgaven til å fungere hvis du hadde mer tid?

2. Logg inn på Devilry.

3. Lever alle .py-filene samt README.txt i *samme innlevering*.

4. Husk å trykke lever og sjekk deretter at innleveringen din er komplett.

Den obligatoriske innleveringen er minimum av hva du bør ha programmert i løpet av en uke. Du finner flere oppgaver for denne uken her.