IN1000—innleveringsoppgave 2

Sist endret: 240123

Introduksjon

Denne gangen er det fem oppgaver som skal løses, der hver deloppgave teller 3 poeng. Les gjennom hver oppgave før du begynner å programmere, og forsøk gjerne å løse oppgavene på papir først! Pass på at oppgavene du leverer ligger i riktig navngitte filer, som vist under oppgavetittelen. For hvert program du skriver skal du legge ved en kommentar i toppen av fila som forklarer hva programmet gjør. Videre forventes det at du kommenterer koden underveis så det blir tydelig hva du har tenkt.

Læringsmål

Du skal nå ha fått en mer presis forståelse av hvordan koden fungerer, og skal kunne vise at du forstår kodeflyten i et program og hvordan uttrykk evalueres. Du skal ha lært om forskjellige datatyper og om gjenbruk av kode ved hjelp av prosedyrer.

Oppgave 1: Utskriftsprosedyre

Filnavn: *utskriftsfunksjon.py*

1. Lag et program som kommuniserer med brukeren slik at det tar inn et navn og et bosted fra terminalen. Eksempel på hvordan en kjøring av programmet skal se ut:

   > python3 utskriftsfunksjon.py
   Skriv inn navn: Espen Askeladd
   Hvor kommer du fra? Oslo
   Hei, Espen Askeladd! Du er fra Oslo.

2. Flytt koden som leser inn informasjon og skriver ut en hilsen til en egen prosedyre. Kall denne prosedyren 3 ganger slik at du får lest inn og skrevet ut informasjon om 3 personer. Til denne oppgaven trenger du kun å levere inn deloppgave 2. Eksempel på kjøring av programmet:

   > python3 utskriftsfunksjon.py
   Skriv inn navn: Espen Askeladd
   Hvor kommer du fra? Oslo
   Hei, Espen Askeladd! Du er fra Oslo.

   Skriv inn navn: Donald Duck
   Hvor kommer du fra? Andeby
   Hei, Donald Duck! Du er fra Andeby.

   Skriv inn navn: Minni Mus
   Hvor kommer du fra? Museby
   Hei, Minni Mus! Du er fra Museby.

Synes du denne oppgaven var vanskelig? se Trix-oppgave 2.01, 2.09, 2.12

Synes du denne oppgaven var lett? Se Trix-oppgave 2.06, 2.17, 2.22

Oppgave 2: Konvertering

Filnavn: *fahrenheit_celsius.py*

Du skal skrive et verktøy for å konvertere en gitt temperatur fra fahrenheit til celsius.

1. Skriv et program med en variabel som er temperatur i fahrenheit, bestem selv hva temperaturen skal være og hva variabelen skal hete.

2. Skriv ut temperaturen i fahrenheit.

3. Definer en ny variabel som bruker variabelen i punkt 1 til å finne temperaturen i celsius. For å regne om fra fahrenheit til celsius kan du bruke denne formelen:

   ((temperatur i fahrenheit) − 32) ∗ 5/9.

4. Skriv ut temperaturen i celsius.

5. Modifiser programmet slik at variabelen for fahrenheit blir gitt via brukerinput. **Sørg for at variabelen får en tallverdi!** Sjekk så programmet ditt ved å kjøre det og taste inn tallet 100. Får du ca 37,78 grader celsius? Hvis ikke, sjekk at du har fulgt formelen gitt over.

Synes du denne oppgaven var vanskelig? Se Trix-oppgave 2.02, 2.03, 2.08
Synes du denne oppgaven var enkel? Se Trix-oppgave 2.06, 2.16, 2.17

Oppgave 3: Kodeflyt

Filnavn: *kodeflyt.pdf*

Dette er en tekstoppgave. Du kan gjøre den med penn og papir og ta et bilde eller scanne inn arket, eller du kan gjøre den på datamaskin. Du skal levere oppgaven som en PDF-fil sammen med de andre kodefilene i Devilry. Det holder ikke å gi filnavnet endelsen .pdf for å gjøre den til en pdf.

Du skal forklare programflyten til et program, det vil si i hvilken rekkefølge de forskjellige linjene utføres når vi kjører programmet. Du skal gjøre dette ved å skrive tall foran linjene i programmet. Tallene beskriver i hvilken rekkefølge programsetningene utføres, slik at 1 er den første setningen som utføres, 2 er den neste, og så videre. Her er et eksempel (legg merke til tallrekkefølgen oppgitt til venstre!):

```
1  heltall = 13
2  if heltall < 10:
       print("Mindre!")
3  else:
4      print("Ikke mindre!")
```

Som vi ser definerer vi først en variabel *heltall* med verdien 13. Så sjekker vi om *heltall* er mindre enn 10. Ettersom dette ikke stemmer utføres ikke neste linje, men vi går istedet videre til *else* og utfører setningen vi finner der.

Hvis en linje kjøres flere ganger (for eksempel ved flere kall på samme prosedyre) kan du skrive flere tall ved siden av setningen.

Beskriv programflyten når følgende program kjøres. Som du ser tar programmet input fra bruker, og **du skal her anta at brukeren under kjøring av programmet taster inn tallet 8**.

```
def print_prosa():
    print("Melding til alle gaardeiere:")
    print("Antall dyr paa gaarden: ")

antall_dyr = 4
print_prosa()
print(antall_dyr)
antall_nye_dyr = int(input("Hvor mange nye dyr kommer til gaarden: "))
antall_dyr = antall_dyr + antall_nye_dyr
print_prosa()
print(antall_dyr)

if antall_dyr > 12:
    print("Det er mer enn ett dusin dyr paa gaarden!")
elif antall_dyr == 12:
    print("Det er ett dusin dyr paa gaarden!")
else:
    print("Det er mindre enn ett dusin dyr paa gaarden!")
```

Synes du denne oppgaven var vanskelig? Se Trix-oppgave 2.11, 2.14
Synes du denne oppgaven var lett? Se Trix-oppgave 2.18, 2.20, 2.21, 2.22

Oppgave 4: Kodeforståelse og feilsøking

Filnavn: *kodeforstaaelse.py*

Dette er en teorioppgave. Besvar teorispørsmålene som kommentarer øverst i *kodeforstaaelse.py*.

Les følgende kode grundig og forsøk å svare på deloppgavene under. Test deretter besvarelsen din ved å lime inn koden i *kodeforstaaelse.py* og kjøre programmet.

```
heltallSomTekst = input( "Tast inn et heltall! " )
heltall = int( heltallSomTekst )
if heltall < 10:
    print ( heltall + "Hei!" )
```

1. Vil dette programmet kjøre? Begrunn svaret.

2. Hvilke problemer vil vi kunne møte på når vi kjører denne koden?

Synes du denne oppgaven var vanskelig? Se Trix-oppgave 2.03, 2.05, 2.13
Synes du denne oppgaven var lett? Se Trix-oppgave 2.15, 2.23

Oppgave 5: Egen oppgave

1. Skriv din egen oppgavetekst til en oppgave som har med beslutninger å gjøre (if/else).

   Et forslag er å skrive et quiz-program som stiller spørsmål til bruker og gir tilbakemelding på om svarene er korrekte.

2. Løs oppgaven! Du skal levere både oppgaveteksten og besvarelsen (skriv oppgaveteksten som kommentar(er) over løsningen din).

Krav til koden

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

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

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