

Oppgaver

I denne seksjonen finner du oppgaver som hører til dag 1 av Kodeskolens kræsjkurs i programmering.

Tema for første dag er variabler, skilpaddeprogrammering, løkker og betingelser. Dersom du står fast er det bare å spørre. God koding!

Variabler og regning

Oppgave 1 *Finn tre feil!*

Her følger det tre programmer som ikke fungerer. Skriv programmet inn på din egen maskin og kjør det, les feilmeldingen, og prøv å tolke den. Når du skjønner hva som er galt, rett opp feilen og kjør programmet.

a)

```
1 print(Hva heter du?)
```

b)

```
1 navn = 'Sahid'
2 print('hei' navn)
```

c)

```
1 x = '3'
2 z = x + 2
3 print('x + y =', z)
```

Oppgave 2 *Konvertering av temperatur*

I Norge oppgir vi temperaturer i målestokken *Celsius*, men i USA bruker de ofte målestokken *Fahrenheit*. Hvis du finner en kakeoppskrift fra USA kan det for eksempel stå at du skal bake kaken ved 350 grader. Da mener de altså 350°F. Vi vil nå lage et verktøy som kan konvertere denne temperaturen for oss, sånn at vi vet hva vi skal bake kaken ved i Celsius..

For å regne over fra Fahrenheit til Celsius bruker vi formelen:

$$C = \frac{5}{9}(F - 32).$$

Der F er antall grader i Fahrenheit, og C blir antall grader i celsius.

- a) Start med å opprette en variabel, `fahrenheit`, som du setter lik 350.
- b) Lag et program regner ut hvor mange grader Celsius 350 °F tilsvarer. Virker det rimelig å skulle bake en kake ved denne temperaturen?

Programmet du har lagd tar en temperatur i Fahrenheit, og gjør om til Celsius. Men hva om vi ønsker å gå motsatt vei? Om vi ønsker å lage et nytt program som gjør motsatt, så må vi først ha en formel for F .

- c) Klarer du å ta uttrykket

$$C = \frac{5}{9}(F - 32).$$

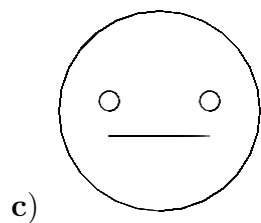
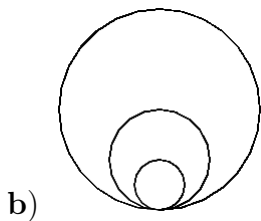
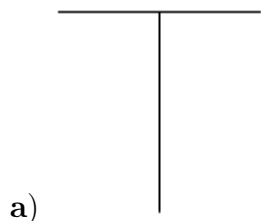
og løse for F ?

- d) Lag et nytt program som har en variabel, `fahrenheit`, som du setter lik 220. Regn så ut hvor mange grader Fahrenheit dette er, og skriv det ut til brukeren.
- e) Modifiser programmet ditt til å finne frysepunktet og kokepunktet til vann i Fahrenheit målestokken.

Skilpaddeprogrammering

Oppgave 3 *Vår første turtletegning*

Bruk `turtle` til å gjenskape figurene under. (HINT: du kan finne en liste av alle mulige `turtle`-kommandoer i dokumentasjonen: <https://docs.python.org/3/library/turtle.html>)



(HINT: For å løfte pennen mellom tegninger trenger du penup og pendown funksjonene)

Oppgave 4 *Vi slår opp i dokumentasjonen*

Under er en kodesnutt som tegner en tegning:

```

1  from turtle import *
2
3  left(60)
4  forward(50)
5  right(40)
6  forward(30)
7  home()
8  left(120)
9  forward(50)
10 left(40)
11 forward(30)
12 home()
13
14 done()
```

15 `bye()` # Nødvendig hvis vi bruker Spyder

- a) Kjør koden. Hva tror du `home()` gjør?
- b) Se i dokumentasjonen (<https://docs.python.org/3/library/turtle.html>). Hvordan beskrives `home`-funksjonen? Hadde du rett?

Lister og løkker

Oppgave 5 *Fruktliste*

I denne oppgaven skal vi øve å opprette lister og hente ut elementer med indeksering

- a) Opprett en liste, `frukt`, som inneholder elementene `'eple'`, `'hanningmelon'`, `'kiwi'` og `'appelsin'`
- b) Hvilken indeks har `'hanningmelon'`? Bruk indeksering til å hente ut dette elementet fra lista og skriv det ut til terminalen
- c) Hvilket element får du dersom du indekserer med `[-1]`? Skriv ut elementet med indeks `-1` og se om du hadde rett.

Oppgave 6 *For-løkker*

Syntaksen for å løkke over en liste med en `for`-løkke i Python er slik:

```
1 for [løkkevariabel] in [liste]:  
2     [det som skal gjentas, f.eks print(løkkevariabel)]
```

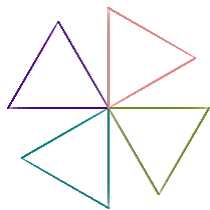
- a) Opprett en liste, `navneliste`, som skal inneholde fem navn. (Du kan velge selv hvilke navn)
- b) Bruk en `for`-løkke til å gå igjennom lista med navn og skrive ut en hilsen til hvert navn

Oppgave 7 *Trekantblomst*

Les koden under

```
1 from turtle import *
2
3 forward(100)
4 right(120)
5 forward(100)
6 right(120)
7 forward(100)
8 right(120)
9
10 done()
11 bye() # Nødvendig hvis vi bruker Spyder
```

- a) Hva tror du den tegner? Kjør koden og se om du hadde rett.
- b) Gjør om koden slik at den bruker en **for**-løkke til å tegne den samme figuren uten å skrive `forward` eller `right` mer enn en gang hver.
- c) Opprett en liste `farge_liste` som inneholder fargene `'olivedrab'`, `'teal'`, `'indigo'` og `'lightcoral'`
- d) Bruk en **for**-løkke og `color`-kommandoen sammen med `farge_liste` og koden fra oppgave b) til å gjenskape blomsten under.



Betingelser

Oppgave 8 *Vinkeltyper*

Vi kan dele vinkler inn i tre forskjellige typer:

1. En vinkel som er mindre enn 90° kalles en *spiss* vinkel.
 2. En vinkel som er større enn 90° kalles en *stump* eller *butt* vinkel.
 3. En vinkel som er akkurat 90° kalles en *rett* vinkel
-
- a) Opprett en variabel `vinkel` som du gir en verdi mellom 0 og 360. Bruk så `if`, `elif` og `else` til printe ut om vinkelen er *spiss*, *stump* eller *rett*
 - b) Modifiser koden slik at du bruker en for-løkke, med en tellevariabel, `vinkel`, som varierer mellom 0 og 360 grader. La det gå 10 grader mellom hver vinkel og print ut ut om hver av disse 36 vinklene er spiss, stump eller rett. (hint, du kan bruke `range(0, 360, 10)` for å generere vinkler mellom 0° og 360° med 10° mellom hver vinkel)

Oppgave 9 Mønstre med løkker og betingelser

I denne oppgaven skal vi se på hvordan vi kan kombinere løkker og betingelser for å lage fine mønstre.

- a) Bruk `pensize`-kommandoen for å sette pennestørrelsen lik 50.
- b) Lag en for-løkke som du itererer over 100 ganger. For hver iterasjon skal du bevege skilpadden to steg fremover
- c) Bruk en betingelse for å sjekke om tellevariabelen er et partall eller et oddetall. Dersom den er et partall skal du sette pennefargen til svart ('`black`') og dersom det er et oddetall skal du sette pennefargen til gull ('`gold`'). (Hint: du kan bruke `tall % 2 == 0` for å sjekke om `tall` er delelig på 2)
- d) Nå skal vi også endre pennestørrelsen inne i løkka for å lage et interessant mønster. På starten av løkka skal du bruke `if` og `elif` for å sette pennestørrelsen til 20 på iterasjon 25 og iterasjon 75, og 50 på iterasjon 0 og iterasjon 50.
- e) Nå har vi kode for å lage en interessant strek. Det neste vi skal gjøre er å bruke denne koden for å lage en kul trekant. For å gjøre det må du plassere løkka som lager en rett strek inni en ny løkke som du itererer over tre ganger. Etter at hver strek har blitt tegnet bruker du

right-funksjonen for å snu skilpadda 120° . Nedenfor ser du hvordan sluttfiguren din kan se ut.

