

DEL I

Grunnleggende programmering

KAPITTEL 2

Variabler

2.1 Lagre tekst i en variabel

(2_tekst.py)

```
1 hobby = "piano"  
2 print("Min hobby er", hobby)
```

Variabler lar oss lagre tall eller tekst til senere bruk.

- a) Les koden ovenfor, og gjett på resultatet.

Nedenfor ser du tre typiske feil, sammen med den korrekten koden.

```
hobby = "piano"  
print("Min hobby er", hobby)
```

```
hobby = "piano"  
print("Min hobby er," hobby)
```

```
hobby = "piano"  
print("Min hobby er", Hobby)
```

```
hobby = piano  
print("Min hobby er", hobby)
```

- b) Forsök å finne feilen i hvert enkelt tilfelle.
- c) Skriv av koden, lagre som «2_tekst.py» og kjør programmet.
- d) Utvid koden som vist nedenfor. Gjett på resultatet, før du kjører programmet.

```
1 hobby = "piano"  
2 hobby = "fotball"  
3 hobby = "tennis"  
4 print("Min hobby er", hobby)
```

- e) Legg til to kodelinjer med `print("Min hobby er", hobby)` på riktig plass i koden, slik at resultatet blir som vist nedenfor.

```
Min hobby er piano  
Min hobby er fotball  
Min hobby er tennis
```

2.2 Lagre tall i variabler

(2_tall.py)

```
1 timelonn = 140
2 print("Timelønna er", timelonn)
```

Usman jobber som ekstrahjelp på et bilverksted. Han får 140 kr i timelønn. Kodelinen `timelonn = 140` gjør at programmet lagrer verdien 140 i variabelen `timelonn`.

- Les koden ovenfor, og gjett på resultatet.
- Lag en ny fil ved å trykke , eller velg **File > New** fra menyen.
- Skriv av koden, lagre som «2_tall.py» og kjør programmet.
- Endre på kodelinje 1 slik at resultatet av programmet blir:

```
Timelønna er 210
```

En uke jobbet Usman 12 timer.

- Legg til de to nye kodelinjene for å vise fram antall timer.

```
1 timelonn = 210
2 print("Timelønna er", timelonn)
3 timer = 12
4 print("Antall timer:", timer)
```

- Legg til kodelinja `ukelonn = 210*12`.

I stedet for å skrive `210*12`, kan vi bruke variabelnavnene `timelonn * timer`.

- Endre kodelinja `ukelonn = 210*12` til `ukelonn = timelonn * timer`.
- Legg til en kodelinje for å skrive ut følgende:

```
Ukelønna er 2520
```

Vi antar at Usman jobber 30 uker i året, og 12 timer per uke.

- Legg til følgende kodelinjer i riktig rekkefølge:

```
print("Årslønna er", aarslønn, "kr.")
      uker = 30
aarslønn = uker * ukelonn
```

- Utforsk ved å endre på `timer = 12` for å finne ut antall timer Usman må jobbe ukentlig for at årslønna blir 100 000 kr.

2.3 Bruke variabler i utregning

(2_fart.py)

```
1 km = 3.8
2 print(f"Du løp {km} km.")
```

Kateryna er en ivrig løper. Hun vil registrere lage et program hvor hun kan registrere distanse i kilometer og tid i timer, og få beregnet gjennomsnittsfarten i km/t og m/s.

- Les koden ovenfor, og gjett på resultatet.
- Skriv av koden, lagre som «2_fart.py» og kjør programmet.

I koden `print(f"Du løp {km} km.")`, betyr `{km}` at vi setter inn verdien av variabelen `km` på det stedet i strengen (teksten).

- Undersøk hva som skjer dersom du fjerner f-en slik: `print("Du løp {km} km.")`.
- Angre endringen og endre tilbake til kodelinje 2 slik det står øverst på siden.
- Legg til følgende kodelinjer i riktig rekkefølge for å regne ut gjennomsnittsfarten i km/t.

```
tid_min = 24                                     print(f"Farten var {fart_kmt} km/t.")
tid_timer = tid_min / 60                         fart_kmt = km / tid_timer
```

- Legg til kode for å regne ut farten i m/s og lagre verdien i variabelen `fart_mps`. Bruk at 1 m/s er 3,6 km/h.
- Skriv ut farten i m/s på samme måte som vi skrev ut farten i km/t.



Figur 2.1: Kateryna på joggetur.

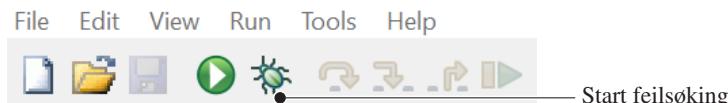
2.4 Feilsøking i Thonny

(2_feilsok.py)

```
1 x = 10 - 2*3**2
2 y = 4**0.5
3 summ = x + y
4 print(summ)
```

For at du skal forstå hvordan Python leser og evaluerer koden ovenfor, skal du nå benytte deg av Thonnyrs feilsøkerfunksjon.

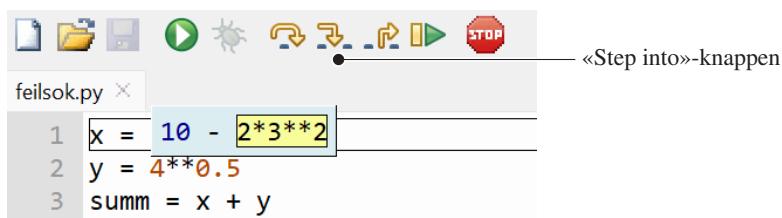
- Les koden ovenfor, og gjett på resultatet.
- Skriv av koden, lagre som «2_feilsok.py» og kjør programmet.



Figur 2.2: Feilsøke-knappen i Thonny.

For å få større knapper i Thonny, velg **Tools** > **Options...** > **General** > **UI scaling factor** i menyen. Sett «UI scaling factor» til for eksempel 2,5, og start Thonny på nytt.

- Trykk på insekt-knappen for å starte feilsøkingen. Se figur 2.2. Fra menyen kan du gjøre slik: **Run** > **Debug current script (nicer)**.



Figur 2.3: Feilsøking i Thonny.

- Trykk på «step into»-knappen gjentatte ganger til feilsøkingen er ferdig. Se figur 2.3. Legg merke til hvordan koden blir evaluert.
- Endre den første kodelinja til `x = (10 - 2)*3**2`. Gjett på verdien til `x`, før du kjører feilsøking på nytt. Det vil si trykk , og deretter gjentatte ganger.

Du kan trykke når som helst for å stoppe feilsøkingen.

2.5 Endre en variabel

(2_endring.py)

```

1 tall = 5
2 tall = tall + 2
3 tall = tall * 3
4 print(tall)

```

- Les koden ovenfor, og gjett på resultatet.
- Skriv av koden, lagre som «2_endring.py» og kjør programmet.
- Kjør feilsøking på koden. Det vil si trykk  , og deretter  gjentatte ganger.
Legg merke til hvordan variabelen endrer verdi.
- Legg til kodelinja `tall = 10` rett ovenfor `print(tall)` . Gjett på resultatet, før du kjører programmet.

Vi har nå sett at `tall = 10` gir variabelen `tall` verdien 10, mens `tall = tall + 2` øker verdien med 2.

```

1 tall = 12
2 tall = tall - 3
3 tall = tall / 2
4 print(tall)

```

Gi tall verdien 12
 Reduser tall med 3
 Gjør tall halvparten så stor
 Skriv ut verdien til tall

- Les koden ovenfor, og gjett på resultatet.
- Erstatt tidligere kode med koden ovenfor, og bruk feilsøking. Trykk  og deretter  gjentatte ganger.

Vi utvider nå algoritmen med ytterligere to steg.

Gi tall verdien 12
 Reduser tall med 3
 Gjør tall halvparten så stor
 Øk tall med 5.5
 Gjør tall 10 ganger så liten
 Skriv ut verdien til tall

- Endre programmet slik at det følger algoritmen ovenfor.

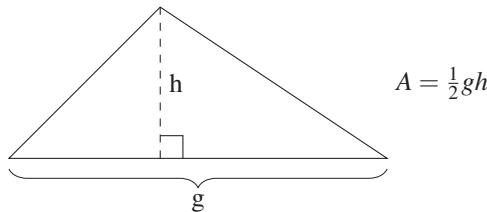
2.6 Endre flere variabler

(2_flere.py)

```
1 g = 6
2 h = 2
3 A = 1/2*g*h
4 print(A)
```

Arealet av en trekant er gitt ved $A = \frac{1}{2} \cdot g \cdot h$. Se figur 2.4. Ved programmering kan vi utforske sammenhenger mellom g , h og A .

- Les koden ovenfor, og gjett på resultatet.
- Skriv av koden, lagre som «2_flere.py» og kjør programmet.



Figur 2.4: Arealet av en trekant.

Anta først at $h = 2 \cdot g$. Det betyr at høyden alltid er det dobbelte av grunnlinja.

- Endre `h = 2` til `h = 2*g`
- Endre grunnlinja til 3 ved å endre `g = 6` til `g = 3`.
- Endre grunnlinja til 4, 5 og 10. Kjør programmet på nytt for hver gang og legg merke til arealet. Forklar mønsteret i arealene.
- Endre programmet ved å sette sammen følgende kodelinjer i riktig rekkefølge:

```
g = h/2 - 3           A = 1/2*g*h
h = 8                 print(A)
```

- Forsök andre verdier av `h` helt til du finner verdien som gjør at arealet blir 18.
- Skriv ut en setning på formen `Høyde 8 gir areal 4.0`. Bruk `print(f"\u2022")`. Se side 5 dersom du har glemt hvordan.

2.7 Heltallsdivisjon

(2_boller.py)

- 25 g gjær
- 60 g margarin
- 2,5 dl melk



Figur 2.5: Nystekte boller.

Bengt skal bake boller. Til 1 porsjon trenger han blant annet ingrediensene og mengdene vist ovenfor. Vi skal lage et program som bruker heltallsdivisjon til å bestemme antall porsjoner Bengt kan bake. Han finner frem gjær og måler at han har 185 g gjær.

- a) Les koden nedenfor, og gjett på resultatet.

```

1 gjar_1porsjon = 25
2 gjar_hjemme = 185
3
4 gjar_porsjoner = gjar_hjemme / gjar_1porsjon
5 print(f"Gjær gir {gjar_porsjoner} porsjoner.")

```

- b) Skriv av koden, lagre som «2_boller.py» og kjør programmet.

Vi er bare interessert i antall hele porsjoner han kan lage.

- c) Endre `/` til `//` på kodelinje 4. Kjør programmet.
- d) Endre verdien til `gjar_hjemme` fra 185 til 174, 175 og 176. Kjør programmet mellom hver gang og gjett på resultatet.
- e) Endre verdien til `gjar_hjemme` tilbake til 185.

Bengt finner 500 g margarin i kjøleskapet.

- f) Legg til kodelinjene på korrekt plass slik at du får beregnet antall hele porsjoner fra margarinen:

```

marg_1porsjon = 60      marg_porsjoner = marg_hjemme // marg_1porsjon
marg_hjemme = 500       print(f"Margarin gir {marg_porsjoner} porsjoner.")

```

Bengt finner 15,5 dl melk.

- g) Bruk blant annet kodelinjene `molk_1porsjon = 2.5` og `molk_hjemme = 15.5`, og legg til kode for å bestemme antall hele porsjoner boller man får for 15,5 dl melk.
- h) Avgjør til sist hvor mange hele porsjoner med boller Bengt kan lage.

2.8 Heltallsdivisjon og rest

(2_rest.py)

```
1 boller = 11
2 venner = 5
3 print(boller / venner)
4 print(boller // venner)
5 print(boller % venner)
```

Noen venner skal dele boller imellom seg slik at alle får like mye. Det er snakk om knallgode kanelboller med pistasj-smak. Det er lørdag.

- Les koden ovenfor, og gjett på resultatet.
- Skriv av koden, lagre som «2_rest.py» og kjør programmet.

For å forstå resultatet av operatorene `//` og `%` bedre, kan vi dele opp brøken slik:

$$\frac{11}{5} = \frac{10}{5} + \frac{1}{5} = 2 + \frac{1}{5}$$

Her er 2 resultatet av heltallsdivisjonen `//` og 1 er resultatet av rest-divisjonen `%`.

- Endre antallet boller i koden til 10. Gjett på resultatet, før du kjører programmet.
- Endre antallet venner til 4. Gjett på resultatet, før du kjører programmet.
- Slett tidligere `print`-kommandoer, og endre koden slik:

```
1 boller = 4
2 venner = 9
3 per_venn = boller // venner
4 rest = boller % venner
5 print(f"{per_venn} boller per venn")
6 print(f"De {rest} siste bollene kan deles opp...")
```

Anta at vi deler hver bolle i 8 biter, av de bollene som er til overs.

- Legg til kodelinja `ekstrabiter = 8*rest`, og lag deretter variablene `ekstrabiter_per_venn` og `rest_ekstrabiter`.
- Fullfør programmet ved å beregne og skrive ut antall ekstra bollebiter per venn, og antall ekstra bollebiter som ble til overs.

2.9 Oppgaver

Oppgave 2.1A

(2_opg1A.py)

```
elv : "Glomma"
Print(elv, "er Norges lengste elv.")
```

Skriv av koden ovenfor, og kjør programmet. Du får feilmelding. Rett feilene i koden, slik at programmet kjører feilfritt.

Oppgave 2.2A

(2_opg2A.py)

```
print(C, "grader Celsius")
C = (F-32) * 5/9
F = 68
print(F, "grader Fahrenheit er")
```

Sett sammen kodelinjene ovenfor slik at programmet konverterer 68 °F til °C.

Oppgave 2.3A

(2_opg3A.py)

```
1 aar = 80
2 dager = aar*365
3 timer = ...
4 print(...)
```

Skriv av og fullfør programmet slik at det skriver ut antall dager og timer i 80 år.

Oppgave 2.4A

(2_opg4A.py)

```
pris = 400
ny_pris = pris - rabatt_kr
rabatt_prosent = 0.70
rabatt_kr = pris*rabatt_prosent
```

En vare koster 400 kr. Den selges nå med 70 % rabatt. Sett sammen kodelinjene i riktig rekkefølge slik at programmet beregner den nye prisen til varen. Avslutt med å skrive ut den nye prisen.

Oppgave 2.5A

(2_opg5A.py)

```
verdi
okning_prosent
ny_verdi
okning_kr
```

Et maleri koster 550 kr. Verdien øker plutselig med 30 %. Bruk de fire variablene ovenfor til å lage et program som beregner og skriver ut den nye verdien til maleriet.

Oppgave 2.6A

(2_opg6A.py)

```
1  fart = 40
2  strekning_m = 75000
3  strekning_km = ___
4  timer = ___
5  print("Turen tar", ___, "timer.")
```

En bil kjører i 40 km per time. Den skal kjøre en strekning på 75 000 m. Skriv ferdig programmet ovenfor slik at det beregner tiden turen tar i timer.

Oppgave 2.7A

(2_opg7A.py)

```
vf_okning = 1 + prosent/100           vf_nedgang = 1 - prosent/100
prosent = 35
```

Ta utgangspunkt i kodelinjene ovenfor, og lag et program som skriver ut vekstfaktoren til 35 % økning, og til 35 % nedgang. Resultatet av programmet skal være på formen:

```
35 prosent økning gir vekstfaktor ___
35 prosent nedgang gir vekstfaktor ___
```

Oppgave 2.8A

(2_opg8A.py)

x = x + 4

x = 2

Gi x verdien 2

Øk x med 4

Reduser x med 3

Gjør x 3 ganger så stor

x = 3*x

x = x - 3

Ordne kodelinjene ovenfor i riktig rekkefølge, slik at algoritmen følges. Avslutt med å skrive ut verdien til x .

Oppgave 2.9A

(2_opg9A.py)

En kjole koster ordinært 499 kr. Nå selges den med 149 kr avslag. Lag et program som beregner den nye prisen, og skriver den ut. Du må bruke variablene pris , avslag og ny_pris i løsningen.

Oppgave 2.10A

(2_opg10A.py)

I en klasse har 15 av elevene vært på tur i høstferien. Sverre har regnet ut at det betyr at 60 % av elevene i klassen har vært på tur. Lag et program som regner ut antall elever i klassen. Bruk blant annet kodelinjene elever_tur = 15 og prosent = 0.60 .

2. Variabler

Oppgave 2.11B

(2_opg11B.py)

```
tall = 5  
tall2 = 4 + tall  
tall = 8  
print(tall2)
```

```
tall2 = 4 + tall  
tall = 5  
tall = 8  
print(tall2)
```

```
tall = 5  
tall = 8  
tall2 = 4 + tall  
print(tall2)
```

Les kodebitene ovenfor. En av kodebitene gir en feilmelding ved kjøring, hvilken? Hva blir resultatet av de to andre kodebitene? Test svarene dine ved å skrive av koden. Lagre til slutt den kodebiten som gir resultatet **9**.

Oppgave 2.12B

(Eksamens IP, vår 2022)

(2_opg12B.py)

```
1 prosent_for = 2.0  
2 prosent_etter = 2.2  
3 prosentpoeng = ___  
4 prosent = ___
```

Renten på et lån steg fra 2,0 % til 2,2 %. Lag et program som først beregner antall prosentpoeng renten steg med, og deretter regner ut antall prosent renten steg med. Du kan ta utgangspunkt i koden ovenfor.



Figur 2.6: Illustrasjon: å drukne i gjeld.

Oppgave 2.13B

(2_opg13B.py)

```
Sett a lik 4  
Øk a med 2*a  
Gjør a 5 ganger så stor  
Gjør a 3 ganger så liten  
Reduser a med 4*a  
Skriv ut verdien til a
```

Lag et program som følger algoritmen ovenfor. Den første kodelinja er **a = 4**.

Oppgave 2.14B

(Eksamensoppgave 1P, høst 2022)

(2_opg14B.py)

```
1 gallon_per_fat = 42
2 liter_per_gallon = 3.785
3 fat = 100*10**6 # 100 millioner fat per dag
4 gallon = ____
5 liter = _____
```

Fat er en enhet for volummåling av råolje. Ett fat tilsvarer 42 US gallon. Det er 3,785 liter i 1 US gallon. I 2022 er det anslått at etterspørselen av råolje vil være 100 millioner fat per dag. Skriv ferdig programmet ovenfor slik at du bestemmer hvor mange liter dette tilsvarer per dag. Gi svaret på standardform.



For å skrive svaret på standardform med to desimaler, kan du skrive
`print(f"Antall liter: {liter:.2e}")`.

Oppgave 2.15B 1T

(2_opg15B.py)

```
1 import math
2 print(math.pi)
```

3.141592653589793

Koden viser hvordan vi kan importere π fra biblioteket `math`. Skriv av koden ovenfor, og bruk deretter `math.pi` til å beregne volumet av ei kjegle med radius 5 og høyde 2.

Formelen for volumet av ei kjegle er $V = \frac{\pi r^2 h}{3}$.

Oppgave 2.16B 1T

(2_opg16B.py)

La a og b være hele positive tall, og $a > b$. Velg for eksempel $a = 4$. Skriv et program som regner ut $(a - b)^2 + 2 \cdot b \cdot (a - b) + b^2$. Forklar resultatet.

Oppgave 2.17B

(2_opg17B.py)

Lag et program som gjør om 15 333 sekunder til hele timer, minutter og sekunder. Du skal løse oppgaven ved bruk av operatorene `//` og `%`.

Oppgave 2.18B

(2_opg18B.py)

Du skal planlegge mottak av 137 migranter til et asylmottak. Hvert rom i lokalet har plass til opptil 6 migranter. For å sikre at alle får et rom, trenger du å vite hvor mange rom du må forberede.

Skriv et program som beregner antall fulle rom, og antall migranter som blir værende i det siste rommet. Løsningen skal bruke operatorene `//` og `%`.

2. Variabler

Oppgave 2.19B

(2_opg19B.py)

```
1 klokka_naa = 14
2 timer = 51
3 ankomst = klokka_naa + timer
4 hele_dager = ...
```

Klokka er 1400. Din venn kommer på besøk om 51 timer. Skriv et program som finner antall dager til vennen kommer, og klokkeslettet han ankommer på. For eksempel **Vennen kommer kl 15 om 4 dager.** Begynn koden som vist ovenfor.

Oppgave 2.20B

(2_opg20B.py)

Vigdis skal lage vafler. For å lage en porsjon trenger hun følgende ingredienser:

- 100 g mel
- 2 egg
- 3 dl melk

Vigdis finner ut at hun har 750 g mel, 12 egg, og 15 dl melk hjemme. Hun vil vite hvor mange hele porsjoner vafler hun kan lage med ingrediensene hun har.

Skriv ut hvor mange hele porsjoner Vigdis kan lage med hver av ingrediensene. For å finne ut hvor mange porsjoner vafler Vigdis kan lage totalt, må hun ta hensyn til ingrediensen hun har minst av i forhold til hva oppskriften krever.



For å finne det minste tallet av flere tall i Python, kan du bruke `min()`-funksjonen.
For eksempel:

```
minste_antall_porsjoner = min(porsjoner_mel, porsjoner_egg, porsjoner_melk) .
```

Oppgave 2.21S

(2_opg21S.py)



```
1 print("3+"*7)
2 svar = 8*3
3 regnestykke = "8*3 = " + "3+"*7 + "3 =""
4 print(regnestykke, svar)
```

Utforsk de nye elementene på kodelinje 1 og 3 ovenfor. Endre deretter koden slik at du skriver $99 \cdot 4$ som gjentatt addisjon, det vil si $4 + 4 + 4 + \dots$ korrekt antall ganger. Du skal benytte deg av teknikkene som demonstreres på kodelinje 1 og 3. Undersøk deretter desimalmønsteret i $\frac{3}{7}$, og skriv brøken som et desimaltall med 100 desimaler.

KAPITTEL 3

Inndata

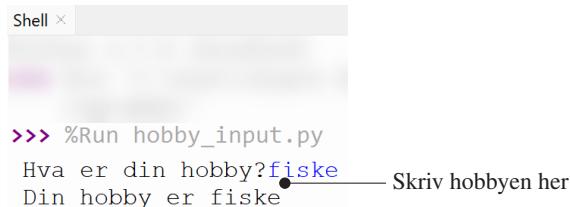
3.1 Lese inn tekst

(3_tekst.py)

```
1 hobby = input("Hva er din hobby?")
2 print("Din hobby er", hobby)
```

Kommandoen `input` lar brukeren skrive inn et ord eller et tall, når programmet kjører.

- Les koden ovenfor, og gjett på resultatet.
- Skriv av koden, lagre som «3_tekst.py» og kjør programmet.
- Når programmet kjører, må du skrive inn en hobby i *Shell*-delen av Thonny.
Trykk deretter `Enter`. Se figur 3.1.



Figur 3.1: Kjøring av program med inndata.

- Legg til koden nedenfor for å spørre om brukerens bosted.

```
bosted = input("Hva kalles ditt bosted? ")
```

- Legg til den riktige av følgende kodelinjer, for å skrive ut resultatet:

<code>print("Ditt bosted er," bosted)</code>	<code>print("Ditt bosted er" bosted)</code>
<code>print("Ditt bosted er", bosted)</code>	<code>print("Ditt bosted er, bosted")</code>

- Legg til et tredje spørsmål på samme måte. Lagre svaret i en variabel og skriv ut en setning. Spør for eksempel om favorittfarge.

3.2 Lese inn tall

(3_tall.py)

```
1 tallA = input("Oppgi et tall: ")
2 tallB = input("Oppgi et tall til: ")
3 summen = tallA + tallB
```

Vi skal lage et program som leser inn to tall under kjøring, og deretter regner ut summen.

- Les koden ovenfor, og gjett på resultatet.
- Skriv av koden, lagre som «3_tall.py» og kjør programmet.
- Oppgi heltallene 3 og 5 til det kjørende programmet. Resultatet av programmet skal se ut som vist nedenfor.

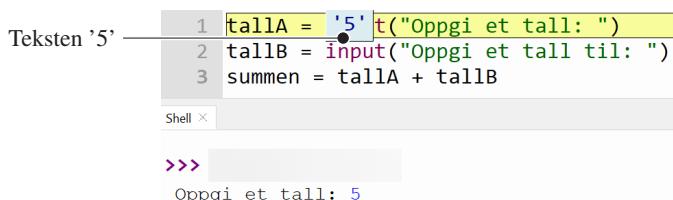
```
>>> %Run 3_tall.py
Oppgi et tall: 3
Oppgi et tall til: 5
```

- Bruk feilsøking til å undersøke verdien til `summen`. Trykk  etterfulgt av gjentatte trykk på . Feilsøkingen tar en pause når det er tid for å oppgi et tall.



Har du glemt hvordan feilsøke? Se side 6.

Når vi bruker `input`, tolker Python det vi taster inn som en tekst. Summen av tekstene '3' og '5' gir teksten '35', mens summen av tallene 3 og 5 gir tallet 8. Se figur 3.2.



Teksten '5' —

```
1 tallA = '5't("Oppgi et tall: ")
2 tallB = input("Oppgi et tall til: ")
3 summen = tallA + tallB
```

Shell

```
>>>
Oppgi et tall: 5
```

Figur 3.2: Feilsøking. Inndata som tekst.

- Skriv inn to andre heltall og gjett på resultatet.
- Gjenta feilsøkingen med de nye tallene.

3. Inndata



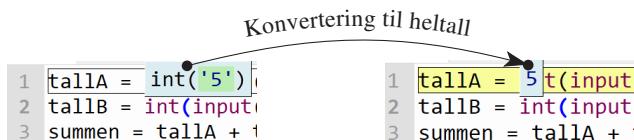
Figur 3.3: Heltall. Fantasi.

Løsningen på problemet, er å konvertere til heltall.

- g) Endre de to første kodelinjene slik:

```
1 tallA = int(input("Oppgi et tall: "))
2 tallB = int(input("Oppgi et tall til: "))
```

- h) Kjør feilsøking på nytt, og skriv inn to heltall. Observer hva som skjer.



Figur 3.4: Feilsøking. Konvertering med int.

Figur 3.4 viser hvordan `int`-kommandoen konverterer fra teksten `'5'` til tallet `5`.

- i) Legg til kodelinja `print(f"tallA + tallB = {summen}")` for å lage en pen utskrift.
- j) Lag ferdig programmet ved å legge til et tredje tall, `tallC`, som leses fra brukeren på tilsvarende måte. Resultatet av programmet skal vises som en utskrift på formen:

```
3 + 9 + 1 = 13
```

3.3 Lese inn desimaltall

(3_desimaltall.py)

```
1 kg = int(input("Oppgi antall kilogram: "))
2 g = kg * 1000
3 print(f"{kg} kg = {g} g")
```

Vi skal lage et program som klarer å lese inn desimaltall.

- Les koden ovenfor, og gjett på resultatet.
- Skriv av koden, lagre som «3_desimaltall.py» og kjør programmet. Oppgi et heltall når programmet kjører, for eksempel 3 eller 9.
- Kjør programmet igjen. Denne gangen oppgir du et desimaltall når programmet kjører, for eksempel 3.2.

Programmet gir en feilmelding av typen **ValueError**. Det kommer av at kommandoen `int` feiler med å konvertere teksten `"3.2"` til et heltall.¹

- Erstatt `int` med `float` og kjør deretter programmet. Oppgi et desimaltall denne gangen også.

`float` konverterer tekst til desimaltall².

- Legg til kodelinja `hg = g / 100` for å regne ut antall hektogram. Legg deretter til kode slik at resultatet blir skrevet ut slik:

```
Oppgi antall kilogram: 0.680
0.680 kg = 680.0 g
680.0 g = 6.8 hg
```

Anta nå at du plukker poteter. På slutten av dagen oppgir du antall kg poteter du har plukket, og tiden du har plukket i timer.

- Fullfør programmet slik at du får vite plukkehastigheten i kg per time, og minutter per hektogram. Bruk blant annet disse kodelinjene:

```
timer = float(input("Oppgi antall timer: "))           kg_per_time = kg/timer
minutter_per_hg = minutter / hg                         minutter = timer * 60
```

¹Riktignok fungerer `int(3.2)`; dette resulterer i tallet 3.

²Desimaltall kalles ofte flyttall innenfor programmering.

3.4 Formatert utskrift

(3_formatert.py)

```

1 verb1 = "liker"
2 verb2 = input("Oppgi et verb: ")
3 print("Jeg", verb1, "å reise. Jeg", verb2, "fort.")

```

Vi lager et program der du spesielt ser nytten av å bruke `print(f"....")` i stedet for `print("....", varA, "...", varB, "...")`.

- Les koden ovenfor, og gjett på resultatet.

Kodelinje 3 er en klønnete måte å skrive ut en tekst kombinert med variabler på. Det er mulig å bruke *formatert tekst* for å få dette til på en mer hensiktsmessig måte.

- Endre kodelinje 3 slik:

```
print(f"Jeg {verb1} å reise. Jeg {verb2} fort.")
```

- La `verb1` og `verb2` bytte plass på kodelinje 3. Gjett på resultatet, før du kjører programmet på nytt.
- Endre kodelinje 1 slik at verdien til `verb1` leses inn fra brukeren.
- Legg til en kodelinje for å lese inn et tredje verb fra brukeren. Lagre verbet i `verb3`.
- Endre programmet slik at resultatet blir en tekst på formen:

```
Jeg verb1 en tur på stien
og verb2 skogens ro.
Da verb3 jeg fra lien
en gjøk som gol koko.
```

Fortell brukeren at verbene skal være i preteritum (fortid).



Figur 3.5: Gjøk på en gren. Ko-ko.

3.5 Prosentkalkulator

(3_prosent.py)

```
1 pris_for = 499
2 print(f"En vare koster opprinnelig {pris_for} kr.")
3 pris_etter = float(input("Oppgi ny pris: "))
4 pris_diff = pris_etter - pris_for
5 print(f"Prisforskjellen er {pris_diff} kr.")
```

Vi skal lage et program som gjør noen prosentberegninger for oss.

- Les koden ovenfor, og gjett på resultatet.
- Skriv av koden, lagre som «3_prosent.py» og kjør programmet.
- Legg til følgende kodelinjer på riktig plass for å regne ut prisforskjellen i prosent:

```
prosent_diff = pris_diff / pris_for
print(f"Forskjellen er {prosent_diff} %.")
```

Nå kan en kjøring av programmet se for eksempel slik ut:

```
En vare koster opprinnelig 499 kr.
Oppgi ny pris: 650
Prisforskjellen er 151.0 kr.
Forskjellen i prosent er 0.3026052104208417 %.
```

- Endre `print(f"Forskjellen er {prosent_diff} %.")` slik:
`print(f"Forskjellen er {prosent_diff*100:.1f} %.")`. Vi ganger med 100, og avrunder til 1 desimal.
- Ta utgangspunkt i koden nedenfor og fullfør den slik at brukeren oppgir bare oppgir prosentvis endring, og programmet regner ut den nye prisen.

```
1 pris_for = 499
2 endring_pros = float(input("Oppgi endring i prosent: "))
3 endring_kr = pris_for * endring_pros/100
4 pris_etter = ...
5 ...
```

Hvis prisen skal øke med 40 %, skriver du 40 under kjøring. Ønsker du å finne prisen dersom rabatten er 70 %, skriver du -70.

3.6 Oppgaver

Oppgave 3.1A

(3_opg1A.py)

```
1 film = input(...)  
2 print(...)
```

Hva er din favorittfilm? Sphere
Din favorittfilm er Sphere.

Skriv av ferdig koden slik at programmet spør brukeren om favorittfilm. Kjøringen av programmet skal se ut som ovenfor til høyre dersom brukeren taster inn «Sphere».

Oppgave 3.2A

(3_opg2A.py)

```
egg = int(input("Oppgi antall egg: "))  
pris = "egg"*9  
print(f"Prisen for {egg} kr er {pris} egg")
```

En bonde selger egg for 9 kr stykk. Lag et program der brukeren kan taste inn antall egg, og få vite hvor mye han må betale. Ta utgangspunkt i koden ovenfor, og korrigér feilene.

Oppgave 3.3A

(3_opg3A.py)

En type maling skal bestå av 35 % rød og 65 % grønn maling. Lag et program som leser inn antall liter maling av denne typen, og deretter beregner antall liter rød og grønn maling. Bruk disse kodelinjene, og legg til den manglende koden:

```
liter = float(input("Antall liter: "))           liter_rod = liter*0.35  
print(f"{liter_rod} L rød maling og {liter_gronn} L grønn maling.")
```

Oppgave 3.4A

(3_opg4A.py)

Lag et program der brukeren kan skrive inn sin alder, og få vite hvilket år vedkommende dør. Programmet legger til grunn en forventet levealder på 83 år. Løs oppgaven ved å fullføre programmet nedenfor slik at kjøringen er på formen vist til høyre.

```
1 forventet_levealder = 83  
2 aar_naa = int(input("Året i år: "))  
3 alder = int(input("Din alder: "))  
4 aar_slutten = ...  
5 ...
```

Året i år: 2024
Din alder: 16
Du vil trolig dø i år 2091.

Oppgave 3.5A

(Eksamens 1P, høst 2022)

(3_opg5A.py)

I en bygård er det 40 leiligheter med til sammen 90 rom. Hver leilighet har enten to eller tre rom. Hvor mange leiligheter har to rom, og hvor mange har tre rom?

Lag et program som lar deg gjette på antall 2-roms leiligheter, og få vite antall 3-roms leiligheter og totalt antall rom. Du skal kunne prøve deg fram helt til det blir 90 rom. Skriv av koden nedenfor, og plasser deretter kodebitene til høyre på rett plass i der det mangler kode (...).

```
1 leil_totalt = ...
2 leil_2rom = int(input("Antall 2-roms: "))
3 leil_3rom = ...
4 rom = 2*leil_2rom + ...
5 print("Antall 3-roms:", leil_3rom)
6 print("Antall rom:", ...)
```

rom
40
 $3 \cdot \text{leil_3rom}$
 $\text{leil_totalt} - \text{leil_2rom}$

Oppgave 3.6A

(Eksamens 1T, vår 2023)

(3_opg6A.py)



Figur 3.6: Teltplass ved elv.

En gruppe speidere har slått opp telt ved en elv. De har et tau som er 80 m langt, og fire pinner. Tauet og pinnene skal de bruke til å sette opp et gjerde rundt teltet. Området de gjerder inn, skal ha form som et rektangel, og de vil ikke sette opp gjerde langs elven. Se skissen på figur 3.6.

Skriv et program som lar brukeren skrive inn lengden under kjøring, beregner bredden og til slutt bestemmer arealet av området. Begynn slik:

```
1 tau = 80
2 lengde = int(input("Oppgi lengden: "))
3 bredde = ...
4 areal = ...
5 # Skriv ut bredden og arealet
```

Oppgave 3.7A

(3_opg7A.py)

Jagerfly bruker ofte Mach som fartsenhet. Lydens hastighet er 1 Mach, som ved havnivå tilsvarer 1230 km/t. Lag et program der brukeren kan skrive inn farten i km/t og få oppgitt farten i antall Mach.



Figur 3.7: Jagerfly og pilot.

Oppgave 3.8B

(3_opg8B.py)

Lag et program der brukeren oppgir lengden og bredden av et rektangel, og får regnet ut omkretsen.

Oppgave 3.9B

(3_opg9B.py)

Lag et program som regner ut rabatt i kroner og ny pris til en vare. Brukeren skal oppgi ordinær pris, og rabatt i prosent. Kjøringen av programmet skal være på formen:

```
Oppgi pris: 499
Oppgi prosent rabatt: 70
Du får 349.30 kr i rabatt
Den nye prisen er 149.70 kr
```

Oppgave 3.10B

(3_opg10B.py)

```
1 innskudd = ...
2 rente = float(input("Oppgi årlig rente: "))
3 vekstfaktor = 1 + rente/100
```

Lag et program som regner ut hvor mye et innskudd på sparekonto vokser i løpet av 10 år. Brukeren skal kunne oppgi innskudd i kroner og årlig rente. Bruk gjerne koden ovenfor som utgangspunkt. Dersom innskuddet er 15 000 kr og den årlige renta er 2,3 %, så er verdien etter 10 år gitt ved

$$15000 \cdot 1,023^{10}$$

Oppgave 3.11B

(3_opg11B.py)

Formelen for svingetiden T til en pendel, når vi ser bort fra friksjon og luftmotstand, er gitt ved

$$T = 2\pi \cdot \sqrt{\frac{L}{g}}$$

L er snorlengden gitt i meter, og g er tyngdens akselerasjon. På jorden er $g = 9,81 \text{ m/s}^2$. Skriv et program der brukeren kan oppgi snorlengden, og få vite svingetiden.

Oppgave 3.12B

(3_opg12B.py)

Lag et program som regner ut den lengste siden i en rettvinklet trekant. Brukeren skal oppgi lengdene av de to korte sidene. Svaret skal avrundes til én desimal.

La a og b være lengden av de to korte sidene. Da kan du regne ut den lengste siden c ved formelen

$$c = \sqrt{a^2 + b^2}$$

Oppgave 3.13B

(3_opg13B.py)

Yatzy-klubben på Yannestad vgs skal avholde en Yatzy-kveld. Ved hvert spillebord må det være 5 terninger for at de skal kunne spille Yatzy. Lag et program der brukeren skriver inn antall terninger, og får vite hvor mange spillebord de kan sette opp og hvor mange terninger de har til overs.



Bruk heltallsoperatoren `//` og modulo-operatoren `%`.

Oppgave 3.14B 1T

(3_opg14B.py)

Tverrsummen av 531 er $5 + 3 + 1 = 9$. Lag et program der brukeren skriver inn et tresifret tall og få vite tverrsummen av tallet. Tresifrede tall er tall fra og med 100 til og med 999.

Programmet kan ha denne strukturen:

```
1 tall = int(input("..."))
2 siffer1 = tall ...
3 siffer2 = tall ...
4 ...
```



Benytt deg av heltallsoperatoren `//` for å hente ut sifrene.

Oppgave 3.15B

(3_opg15B.py)

Lag et program der brukeren kan skrive inn ett klokkeslett , for eksempel 16:42, og få vite antall timer til midnatt. For å lese inn et klokkeslett og dele det opp i timer og minutter, kan du gjøre slik:

```
timer, minutter = input("Oppgi klokkeslett: ").split(":")
```

Som en testverdi, kan du bruke 16:42. Programmet skal da gi at det er 7,3 timer igjen til midnatt.



Figur 3.8: Midnattsol i Finnmark. Maleri.

Oppgave 3.16S

(3_opg16S.py)



En gruppe venner planlegger å dra på en vennetur. Kostnaden for turen er beregnet til 24 800 kr, og dette inkluderer drivstoff, mat, og overnatting og så videre. Foreldre har gått med på å spleise på deler av turen, men de legger bare inn beløp i hele tusener. Vennene må spleise på det resterende beløpet.

Hvis det er 24 foreldre, betaler hver forelder 1000 kr og vennene må spleise på de resterende 800 kr. Dersom det er 5 foreldre, betaler hver forelder 4000 kr, og vennene må spleise på de resterende 4800 kr. Vennene har bestemt at de skal dele kostnadene likt mellom seg, men en av vennene, Arfan, kan bare være med på halve turen. Arfan skal derfor bare betale halvparten av det de andre vennene betaler.

Lag et program som leser inn antall foreldre og antall venner som skal bli med på turen (inkludert vennen som bare blir med halve turen), og beregner hvor mye hver venn og hver forelder må betale. Programmet skal også beregne hvor Arfan, som bare blir med halve turen, må betale.

DEL II

Programmering i matematikk S1

DEL III

Programmering i matematikk S2
