Flere oppgaver

Her finner du flere relevante oppgaver – for alle de ulike temaene. Dersom du blir fort ferdig, ønsker å øve deg mer på programmering eller bare ønsker inspirasjon til oppgaver du selv kan gi til dine elever igjen kan du gjerne prøve deg på disse.

Oppgavene her er ganske forskjellige i omfang og vanskelighetsnivå. Noen er rettet mot grunnleggende programmering, som er gode hvis man vil repetere ulike konsepter. Andre er mye mer utfordrende, men vi håper at de også er mer interessante. Det kan lurt å være bevisst på hva man vil fokusere på – mengdetrening eller utfordringer. Plukk gjerne oppgaver som er relevant for de fagene du selv underviser i.

God koding!

Innhold

1 Variabler og regning

1.1 Grunnleggende programmering

Oppgave 1 *Tolke feilmedlinger*

I denne oppgaven blir du presentert for noen av de vanligste feilmeldingene man kan få når man programmerer. Her trenger du ikke programmere noe, men les meldingen nøye, og beskriv hva du tror kan føre til en slik feilmelding. Finn også ut hvilken linje feilen er på.

a)

```
Traceback (most recent call last):
  File "test.py", line 6, in <module>
    print(f'Jeg er {alder} år gammel!')
NameError: name 'alder' is not defined
```

b)

```
Traceback (most recent call last):
  File "test.py", line 4, in <module>
    sum = tall1 + tall2
TypeError: unsupported operand type(s) for +: 'int
   ' and 'str'
```

c)

```
  File "test.py", line 3
    navn = 'Jonas
                ^
SyntaxError: EOL while scanning string literal
```

Løsning oppgave 1 *Tolke feilmedlinger*

a) Her prøver vi å bruke variabelen `alder`, men den er ikke definert. Feilen er på linje 6.

b) Vi prøver å plusse sammen en variabel med type `int` og en med type

`str`. Feilen er på linje 4.

c) Her slutter teksten uten at den har fått en avsluttende fnutt på høyre side. Feilen er på linje 3.

1.2 Matematikk

Oppgave 2 *Vinkelsum*

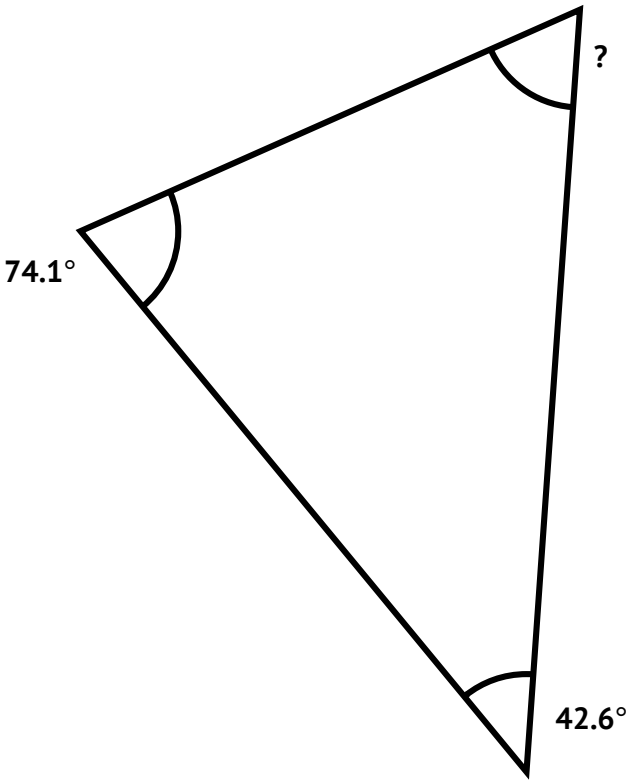
Vinkelsummen til en trekant er alltid 180°. I denne oppgaven skal du lage et program som regner ut den siste vinkelen, gitt at du kjenner til den første vinkelen.

a) Lag en variabel `første_vinkel` og sett den til å være 90°

b) Lag en variabel `andre_vinkel` og sett den til å være 60°

c) Regn ut hva den tredje vinkelen i trekanten må være og lagre svaret i en variabel `tredje_vinkel`

d) Bruk programmet ditt til å finne størrelsen på den ukjente vinkelen i figuren under:



Løsning oppgave 2 *Vinkelsum*

```
første_vinkel = 90
andre_vinkel = 60

tredje_vinkel = 180 - første_vinkel - andre_vinkel
print(f'Den siste vinkelen er {tredje_vinkel} grader')
```

Oppgave 3 *Radianer*

Får å regne om fra grader til vinkler kan vi bruke følgende formel:

$$\theta = \frac{\pi\phi}{180},$$

Hvor θ er vinkelen i radianer og ϕ er vinkelen i radianer

a) Lag en variabel `vinkel_grader` og gi den verdien 45.

b) Regn ut vinkelen i radianer og lagre det i en variabel `vinkel_radianer`. Skriv ut hvor mange radianer som tilsvarer 45 °

c) Modifiser programmet ditt så det spør brukeren om en vinkel i grader skriver ut hva det er i radianer.

d) Finn et utrykk for å gjøre om fra radianer til grader og lag et program som ber om en vinkel i radianer og skriver det ut i grader.

Løsning oppgave 3 *Radianer*

a)

```
1 vinkel_grader = 45
```

b)

```
1 from math import pi
2
3 vinkel_grader = 45
4 vinkel_radianer = pi * vinkel_grader / 180
5 print(f'{vinkel_grader} grader tilsvarer {
     vinkel_radianer:.2f} radianer')
```

c)

```
1 from math import pi
2
3 vinkel_grader = float(input('Skriv inn en vinkel i
      grader: '))
4 vinkel_radianer = pi * vinkel_grader / 180
5 print(f'{vinkel_grader} grader tilsvarer {
     vinkel_radianer:.2f} radianer')
```

d)

```
1 from math import pi
```

```
2
3  vinkel_radianer = float(input('Skriv inn en vinkel
       i radianer: '))
4  vinkel_grader = vinkel_radianer*180/pi
5  print(f'{vinkel_radianer} radianer tilsvarer {
       vinkel_grader.2f} grader')
```

Oppgave 4 *Regne mellom SI-enheter*

En millimeter er 0.01 centimeter. En mikrometer er 0.001 millimeter. En centimeter er 10 000 mikrometer.

Lag en variabel med din høyde i cm, og lag en ny variabel som gjør denne høyden til mm. Lag enda en ny variabel som gjør høyden i mm til µm. Til slutt, lag en variabel som på ny definerer din høyde i cm, men regnet fra µm. Print denne siste variabelen, har du regnet rett og fått riktig høyde i cm?

Løsning oppgave 4 *Regne mellom SI-enheter*

```
1  din_høyde_cm = 165
2  din_høyde_mm = din_høyde_cm * 10
3  din_høyde_um = din_høyde_mm * 10**3
4  din_høyde_nycm = din_høyde_um / 10**4
5
6  print(din_høyde_nycm)
```

Oppgave 5 *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) Lag et program som spør brukeren om en temperatur i antall grader Fahrenheit, og skriver ut den tilsvarende temperaturen i antall grader Celsius.

Husk å gjøre om svaret til et tall, med enten `int(input())` eller `float(input())`.

b) Bruk programmet ditt til å finne ut hvor mange grader Celsius 350°F svarer til. 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 spør brukeren om en temperatur i antall grader celsius, og så skriver ut den tilsvarende temperaturen.

e) Bruk programmet ditt til å finne frysepunktet og kokepunktet til vann i Fahrenheit målestokken.

Løsning oppgave 5 *Konvertering av temperatur*

a)

```
    fahrenheit = float(input('Fahrenheit: '))
    celcius = (5/9)*(fahrenheit-32)

    print(f'{fahrenheit} grader fahrenheit
       tilsvarer {celcius:.0f} celcius')
```

b)

```
    Fahrenheit: 350
    350.0 grader fahrenheit tilsvarer 177 celcius
```

177 ° C virker som en rimelig kakebaketemperatur

c)

$$F = \frac{9}{5}C + 32.$$

d)

```
    celcius = float(input('Celcius: '))
    fahrenheit = (9/5)*celcius + 32

    print(f'{celcius} grader celcius tilsvarer {
       fahrenheit:.0f} fahrenheit')
```

e)

```
    Celcius: 0
    0.0 grader celcius tilsvarer 32 fahrenheit
```

```
    Celcius: 100
    100.0 grader celcius tilsvarer 212 fahrenheit
```

Oppgave 6 *Jordkloden*

I denne oppgaven skal vi øve på å bruke Python som kalkulator, ved å regne litt på jordkloden. Husk at formelen for volumet av en kule er

$$V = \frac{4}{3}\pi R^3.$$

a) Jordkloden er tilnærmet en perfekt kule, og har en radius på 6371 km. Lag et kort program som først definerer en variabel `radius`, og deretter regner ut en variabel `volum`. Skriv til slutt ut svaret til brukeren med `print()`-funksjonen. La svaret være i km^3.

b) Endre programmet ditt så svaret istedet skrives ut i antall liter.

c) Den totale massen til jordkloden er omtrent $M = 5.972 \cdot 10^{24}$ kg. Regn ut hvor mange kg hver liter av jordkloden veier i gjennomsnitt. Virker svaret ditt rimelig?

Løsning oppgave 6 *Jordkloden*

a)

```
radiuskm = 6371
volumkm = (4/3)*3.14*(radiuskm**3)
print("Volumet til jorden er", volumkm, "
   kubikkkilometer.")
```

b)

```
radiusdm = 6371 * 10**4
volumdm = (4/3)*3.14*(radiusdm**3)
```

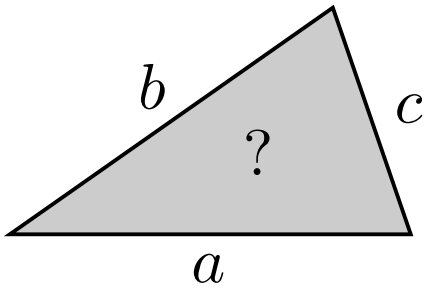
```
print("Volumet til jorden er", volumdm, "liter.")
```

c)

```
massejord = 5.972 * 10**24
vektperliter = massejord/volumdm
print("I gjennomsnitt veier hver liter av jorda",
   vektperliter, "kg.")
```

Oppgave 7 *Herons formel*

Herons formel er et matematisk uttrykk vi kan bruke for å finne arealet av en trekant, hvis vi kjenner lengdene på de tre sidene av trekanten. Herons formel fungerer for alle trekanter, de trenger ikke for eksempel å være rettvinklet.



For å bruke Herons formel må vi først regne ut et tall s, som er halve omkretsen til trekanten,

$$s = \frac{a+b+c}{2}.$$

Når vi har funnet s kan vi regne ut arealet til hele trekanten med formelen:

$$A = \sqrt{s(s-a)(s-b)(s-c)}.$$

a) Regn ut arealet av en trekant med sidelengder på 3, 4, og 5 for hånd, ved hjelp av Herons formel.

Vi skal nå skrive et program, som spør brukeren om lengdene på de tre sidene, og så regner ut arealet for oss.

b) Bruk `float(input(...))` for å spørre brukeren om lengden på de tre sidene av en trekant. Lagre svarene fra brukeren i tre variabler.

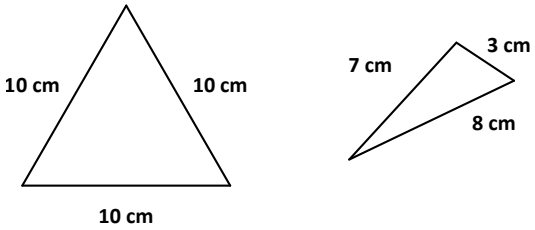
c) Regn så ut variabelen s, som var halve omkretsen til trekanten.

d) Regn så ut arealet A og skriv resultatet tilbake så brukeren kan lese det.

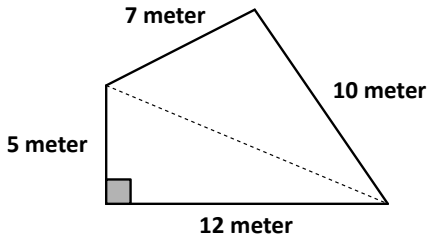
Om du tror du har klart å skrive hele programmet er det på tide å teste det:

e) Bruk programmet ditt til å sjekke trekanten med sider på 3, 4, og 5. Får du samme svar som du gjorde for hånd?

f) Bruk programmet til å finne arealet av de to trekantene under.



g) En firkantet park er formet som figuren under. Klarer du å finne arealet av parken?



Løsning oppgave 7 *Herons formel*

a)

$$s = \frac{3+4+5}{2} = \frac{12}{2} = 6$$

$$A = \sqrt{6(6-3)(6-4)(6-5)} = \sqrt{6 \cdot 3 \cdot 2 \cdot 1} = \sqrt{36} = \underline{\underline{6}}$$

b) c) d) e) f)

```
#importer kvadratrot
from math import sqrt

#bestemme a, b og c
a = float(input("Hva er lengden av side 1? "))
b = float(input("... og lengden av side 2? "))
c = float(input("... og lengden av side 3? "))

#regne ut s
s = (a + b + c)/2

#regne ut A
A = sqrt(s*(s-a)*(s-b)*(s-c))

print(f"Arealet er {A}")
```

g) Bruk pythagoras-programmet ditt til å regne ut den ukjente siden c, eller bruk `c = sqrt(5**2+ 12**2)`.

```
#bestemme a, b og c
a = 5
b = 12
c = sqrt(a**2 + b**2)
d = 7
e = 10

#regne ut s for den øverste trekanten
s = (c + d + e)/2

#regne ut A for den øverste trekanten
A1 = sqrt(s*(s-c)*(s-d)*(s-a))
#regne ut A for den nederst trekanten
A2 = (a*b)/2
```

```
15
16 A_tot = A1 + A2
17 print(f"Det totale arealet er {A_tot:.2f}")
```

```
Det totale arealet er 78.99
```

1.3 Kjemi

Oppgave 8 *Konsentrasjonskalkulator*

Når man gjør eksperimenter i kjemien må man ofte lage løsninger med en bestemt molar konsentrasjon.

Si at vi ønsker å lage en løsning av et stoff X, som veier M g/mol. Vi ønsker å lage en løsning av volum V mL av en gitt konsentrasjon c mol/L.

a) Lag et program som regner ut hvor mange gram av stoff X man må veie ut og løse opp, oppgitt med en nøyaktighet på 0.01 gram.

Hint: For å skrive ut en variabel med 3 desimaler bruk print-formatering med `{m:.2f}`.

b) Test programmet ved å finne mengden bordsalt (NaCl) du trenger for å lage 100 mL løsning med 2.5 mol/L.

c) Ved 25°C klarer man å løse opptil 35,7 gram NaCl per 100 mL vann, etter dette er løsningen metter. Øk konsentrasjonen c i programmet ditt til du er omtrent på denne grensa, hva slags konsentrasjon svarer dette til?

Løsning oppgave 8 *Konsentrasjonskalkulator*

a) Programmet kan se ut som følger. Det viktigste i denne oppgaven er å holde styr på alle enhetene, da er kommentarer et godt virkemiddel.

```
stoff = "NaCl"
M = ... # Molar masse av stoffet i g/Mol
V = ... # Volum i mL
c = ... # Ønsket konsentrasjon i mol/L

m = M*V*c # Masse av stoffet i mg
m /= 1000 # Regner om fra mg til g

# Skriv ut løsningen
print(f"{m:.3f} gram {stoff}")
```

```
11 print(f"Løst i {V} mL vann")
12 print(f"Gir en {c} mol/L løsning")
```

b) Når vi setter inn verdiene oppgitt i oppgaven får vi følgende output

```
14.61 gram NaCl
Løst i 100 mL vann
Gir en 2.5 mol/L løsning
```

Dette kan man sjekke høres rimelig ut ved å dobbeltsjekke beregningen for hånd, eller for eksempel å bruke en online konsentrasjonskalkulator.

c) Utifra svaret fra forrige oppgave ser vi at 14,6 gram gir en 2,5 mol/L løsning. Vi trenger altså litt mer en dobbelt så masse for å treffe metningspunktet. Prøver vi oss frem litt finner vi 6,1 mol/L som et godt estimat

```
35.648 gram NaCl
Løst i 100 mL vann
Gir en 6.1 mol/L løsning
```

Denne oppgaven er ment for å vise hvordan et program som allerede er skrevet kan brukes for å utforske andre problemstillinger en de man originalt var ute etter å løse. Dette spørsmålet kan også enkelt besvares med penn og papir, eller ved å skrive ny kode, men det hadde nok tatt lengre tid.

Oppgave 9 *Regne ut pH*

pH-verdien til en væske eller løsning er et mål på hvor sur væsken er, det vil si konsentrasjonen av hydrogenioner $[H^+]$. Formlene for å regne med pH er gitt ved

$$\text{pH} = -\log_{10}([H^+]), \qquad [H^+] = 10^{-\text{pH}}.$$

a) Bruk et Python program til å regne ut konsentrasjonen av hydrogenioner for en væske med pH 3.6.

b) Bruk et Python program til å regne ut pH verdien til en væske med konsentrasjon av hydrogenioner lik $4.7 \cdot 10^{-5}$ mol/L. **Hint:** for å regne ut 10-logaritmen kan bruke funksjonen `log10` som finnes i `math`, `pylab` og `numpy`

c) Cola har en pH på ca 2.5, nøytralt vann har en pH på 7. Hvor mange ganger fler hydrogenioner er det i cola sammenlignet med vann?

Løsning oppgave 9 *Regne ut pH*

a) Merk at variabelnavn i Python bare kan inneholder bokstaver, understrek (_) og tall, så vi kan ikke kalle konsentrasjonen $[H^+]$ i koden vår, vi velger derfor isteden å kalle den `konsentrasjon`. Vi velger også å skrive ut med stilen `:.2g`, bokstaven gbetyr at konsentrasjonen skrives ut som et desimaltall om det er passende, avhengig av størrelsesorden.

```python
pH = 3.6
konsentrasjon = 10**-pH
print(f"pH = {pH:.1f} ---> [H+] = {konsentrasjon:.2g} mol/L")
```

```
pH = 3.6 --->  [H+] = 0.00025 mol/L
```

b) For å regne ut konsentrasjonen trenger vi briggisk logaritme, altså logaritme av base 10. I matematikk skrives denne ofte bare som log, men i programmering er log den naturlige logaritmen. Da bruker vi istedet funksjonen `log10`, for å tydeliggjøre basen.

```python
from pylab import log10

konsentrasjon = 4.7e-5 # mol/L
pH = -log10(konsentrasjon)
print(f"[H+] = {konsentrasjon:.2g} mol/L ---> pH = {pH:.1f}")
```

```
[H+] = 4.7e-05 mol/L ---> pH = 4.3
```

c) Vi kan nå enten kjøre programmet vårt fra første deloppgave to ganger:

```
pH = 2.5 --->  [H+] = 0.0032
pH = 7.0 --->  [H+] = 1e-07
```

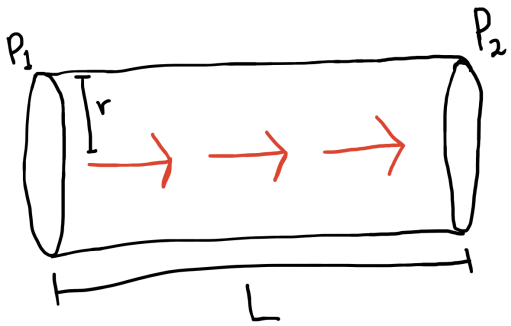
Og så dele det ene svaret på det andre. Eller vi kan skrive et nytt lite program som gjør hele beregningen:

```python
pH_cola = 2.5
pH_vann = 7.0

konsentrasjon_cola = 10**-pH_cola
konsentrasjon_vann = 10**-pH_vann

forhold = konsentrasjon_cola/konsentrasjon_vann

print(f"Cola har omtrent {forhold:.0f} ganger fler hydrogenioner enn vann.")
```

```
Cola har omtrent 31623 ganger fler hydrogenioner
   enn vann.
```

1.4 Fysikk/biologi

Oppgave 10 *Poiseuilles lov og blodstrøm*



Volumstrømmen i blodårer kan uttrykkes ved hjelp av Poiseulles lov. Poiseulles lov angir volumstrømmen i et sylinderformet rør, volum per tidsenhet, og kan uttrykkes slik:

$$q_V = \frac{\pi r^4 (p_1 - p_2)}{8\eta L} \tag{1}$$

Komponentene i telleren er $\pi = 3.14$, radius r samt trykket i starten og slutten av sylinderen, angitt ved p_1 og p_2. I nevneren finner vi konstanten 8, viskositeten η (som for blod er 3-4 centipoise, vi kan gi den et gjennomsnitt på 3,5) og lengden av sylinderen L.

a) Lag et program som tar inn alle parameterverdiene fra bruker ved hjelp av `input`, og lagrer disse i hver sin variabel. Husk å konvertere til flyttall ved hjelp av `float`.

b) Utvid programmet ditt til å regne ut volumstrømmen ved hjelp av formelen over, og skriv ut en informativ melding om hva væskeflyten blir til bruker.

c) Finn et sett av egenvalgte parametere. Prøv å doble en og en verdi. Hvilken av parametrene du endrer fører til størst endring i volumstrøm?

d) Blodtrykket (gitt ved p_1 og p_2) varierer typisk mellom 80 og 120 mmHg. Blodårene i kroppen har ulik størrelse. Aorta, hovedpulsåren, har en radius på ca. 10 mm. Denne forgrener seg til større arterier, og videre til mindre arterier (arterioler), som til slutt har en radius som varierer mellom 5 og 100 μm.

Gitt $p_1 = 100$, $p_2 = 95$ mmHg og lengde $L = 1$ mm, regn ut volumstrømmen for en arterie med radius

- 10 mm
- 1 mm
- 5 µm

Løsning oppgave 10 *Poiseuilles lov og blodstrøm*

```
1  from math import pi
2
3  r = float(input("Hva er radiusen i sylinderen?"))
4  p1 = float(input("Hva er trykket i begynnelsen av
       sylinderen?"))
5  p2 = float(input("Hva er trykket på slutten av
       sylinderen?"))
6  eta = float(input("Hva er viskositeten til væsken?"))
7  L = float(input("Hvor lang er sylinderen?"))
8
9  volumstrøm = (pi*r**4*(p1-p2))/(8*eta*L)
10
11 print(f"Volumstrømmen i sylinderen, gitt radius {r},
       trykk ved start og slutt {p1} og {p2}, viskositet {
       eta} og lengde {L}, er {volumstrøm}.")
```

2 Betingelser

2.1 Grunnleggende programmering

Oppgave 11 and *og* or

I denne oppgaven skal vi se på hvordan and- og or-operatorene fungerer og hvordan de brukes i betingelser.

a) Lag en variabel som inneholder tallet 5. Bruk en betingelse til å teste om tallet er mindre enn 10, og print isåfall ut en melding.

b) Bruk nøkkelordet and til å teste om tallet er mindre enn 10 og større enn 2.

c) Bruk nøkkelordet or til å teste om tallet er mindre enn 6 eller større enn 10.

Løsning oppgave 11 and *og* or

a)

```
tall = 5
if tall < 10:
    print(f"Tallet {tall} er mindre enn 10.")
```

```
tall = 5
if tall < 10 and tall > 2:
    print(f"Tallet {tall} er mindre enn 10 og stø
        rre enn 2.")
```

b)

```
tall = 5
if tall > 10 or tall < 6:
    print(f"Tallet {tall} er mindre enn 6 eller st
        ørre enn 10.")
```

Oppgave 12 and *og* or *2*

Skriv om de følgene programlinjene blir `True` eller `False`. Du trenger ikke skriv noe kode i denne oppgaven, men er du usikker på hva svaret er kan du selv prøve å se hva som skjer ved å taste det inn i Python.

```
tall = 5 #setter variabelen tall lik 5

tall == 5
tall == 7
tall > 8
tall > 2
tall >= 5
tall > 5
not True
True != False

True or False
False and True
True and True

tall == 5 or tall == 6
tall > 5 and tall == 1
tall < 10 and tall > 2
tall < 10 or tall > 6

(tall > 4 or tall == 2) and tall > 6
(tall == 5 or tall > 4) or True
(tall == 5 and tall > 4) or (tall < 4  or tall > 1)

not (tall == 5)
tall != 4
not (tall != 10)

(not (tall < 5) and (tall > 1)) or not tall != 5
```

Løsning oppgave 12 and *og* or *2*

```
tall = 5 #setter variabelen tall lik 5

tall == 5 #True
tall == 7 #False
tall > 8 #False
tall > 2 #True
tall >= 5 #True
tall > 5 #False
not True #False
True != False #True

True or False #True
False and True #False
True and True #True

tall == 5 or tall == 6 #True
tall > 5 and tall == 1 #False
tall < 10 and tall > 2 #True
tall < 10 or tall > 6 #True

(tall > 4 or tall == 2) and tall > 6 #False
(tall == 5 or tall > 4) or True #True
(tall == 5 and tall > 4) or (tall < 4  or tall > 1) #
   True

not (tall == 5) #False
tall != 4 #True
not (tall != 10) #False

(not (tall < 5) and (tall > 1)) or not tall != 5 #True
```

2.2 Matematikk

Oppgave 13 *Absoluttverdi*

Et reelt tall består av et fortegn og en tallverdi, kalt *absoluttverdi*. Når vi finner absoluttverdien til et tall «fjerner vi fortegnet». Det betyr at absoluttverdien til et tall alltid er positiv. Absoluttverdien til et tall a skrives $| a |$ og er

definert som:

$$| a |= \begin{cases} a, & \text{hvis } a \geq 0 \\ -a, & \text{hvis } < 0 \end{cases} \tag{2}$$

a) Lag et progam som ber brukeren om et tall ved hjelp av `input` og `float`, og skriver ut absoluttverdien av tallet

b) Lag et program som ber brukeren om to tall og skriver ut hvilket av tallene som har høyest absoluttverdi

Løsning oppgave 13 *Absoluttverdi*

a)

```
tall = float(input('Skriv inn et tall'))

if tall < 0:
    absoluttverdi = -tall
else:
    absoluttverdi = tall

print(absoluttverdi)
```

b)

```
tall1 = float(input('Skriv inn et tall'))
tall2 = float(input('Skriv inn enda et tall'))

if tall1 < 0:
    absoluttverdi1 = -tall1
else:
    absoluttverdi1 = tall1

if tall2 < 0:
    absoluttverdi2 = -tall2
```

```
11  else:
12      absoluttverdi2 = tall2
13
14  if absoluttverdi1 > absoluttverdi2:
15      print(absoluttverdi1)
16  else:
17      print(absoluttverdi2)
```

Oppgave 14 *Sjekk alder*

En film på kino har aldersgrense 15 år. Vi skal lage et program som interagerer med brukeren, og finner ut om de kan se filmen eller ikke.

a) Lag et program som printer ut en hilsen til brukeren, og spør hvor gamle de er. Lagre alderen i variabelen `alder`.

b) Lag en test som sjekker om brukeren er 15 år eller eldre. Skriv ut passende svar avhengig av om de er gamle nok til å se filmen eller ikke.

c) Utvid programmet ditt til å regne ut hvor mange år det er til brukeren kan se filmen.

d) La programmet ditt gjøre et unntak for de som er 12 år eller eldre dersom de har med seg en voksen.

Løsning oppgave 14 *Sjekk alder*

a)

```
1  alder = int(input("Hei! Hvor gammel er du? "))
```

b) Vi bruker en `if`-test:

```
1  if alder >= 15:
2    print("Vellommen inn på kino!")
3  else:
4    print("Du er dessverre ikke gammel nok")
```

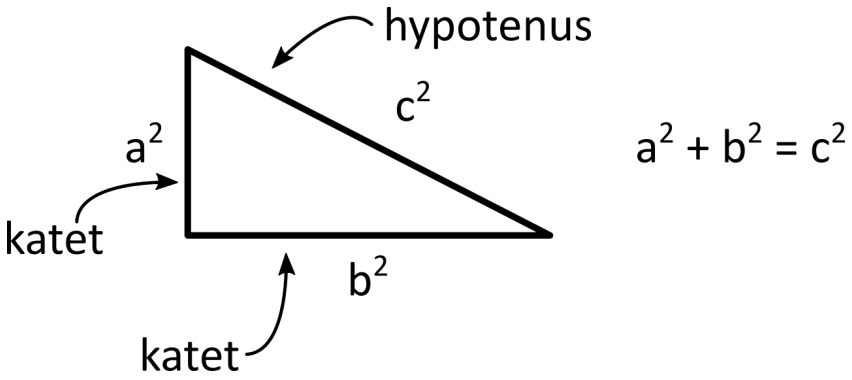
c) Regner ut antall år til brukeren er 15:

```
if alder >= 15:
  print("Vellommen inn på kino!")
else:
  år_igjen = 15 - alder
  print(f"Du er dessverre ikke gammel nok, kom
     tilbake om {år_igjen} år")
```

d) Legger en ny `if`-test inne i den første testen. Hele programmet blir:

```
alder = int(input("Hei! Hvor gammel er du? "))

if alder >= 15:
  print("Vellommen inn på kino!")

elif alder >= 12:
  med_voksen = input("Har du med deg en voksen (ja
     /nei)? ")
  if med_voksen == 'ja':
    print("Velkommen inn på kino!")
  else:
    år_igjen = 15 - alder
    print(f"Du er dessverre ikke gammel nok. Kom
       tilbake om {år_igjen} år eller ha med deg
       en voksen.")

else:
  år_igjen = 15 - alder
  print(f"Du er dessverre ikke gammel nok, kom
     tilbake om {år_igjen} år")
```

Oppgave 15 *Pytagoras*



a) Skriv et program som spør brukeren om lengden på de to katetene i en rettvinklet trekant, og så skriver ut lengden på hypotenusen

b) Utvid programmet ditt, slik at det først spør brukeren om de vil finne en katet eller en hypotenus. Avhengig av hva brukeren svarer må du deretter spørre om lengden på de to kjente sidene og regne deg frem til den ukjente siden.

Løsning oppgave 15 *Pytagoras*

a)

```
1  from math import sqrt
2
3  katet1 = float(input("Oppgi lengde paa katet 1:\n"
       ))
4  katet2 = float(input("Oppgi lengde paa katet 2:\n"
       ))
5  hypotenus = sqrt(katet1**2 + katet2**2)
6
7  print("Hypotenusen i en trekant med kateter" + \
8          "gitt ved {} og {} er {:.2f}.".\
9          format(katet1, katet2, hypotenus))
```

b)

```
1  from math import sqrt
2
3  oppgave = input("Vil du regne ut hypotenusen eller
```

```
    " + \
        "en ukjent katet? Skriv 'H' for hypotenus,
            eller " + \
        "'K' for katet:\n")

if oppgave=="K":
    katet1 = float(input("Oppgi lengde paa katet 1
        :\n"))
    katet2 = float(input("Oppgi lengde paa katet 2
        :\n"))

    hypotenus = sqrt(katet1**2 + katet2**2)

    print("Hypotenusen i en trekant med kateter "
        + \
        "gitt ved {} og {} er {:.2f}.".\
        format(katet1, katet2, hypotenus))
elif oppgave=="H":
    side1 = float(input("Oppgi lengde paa en av "
        + \
            "sidene:\n"))
    side2 = float(input("Oppgi lengde paa den " +
        \
            "andre siden:\n"))

    # hypotenusen er alltid lengre enn begge
        katetene

    if side1 < side2:
        katet1, hypotenus = side1, side2
    else:
        katet1, hypotenus = side2, side1

    katet2 = sqrt(hypotenus**2 - katet1**2)

    print("Den ukjente kateten i en trekant med
        hypotenus " + \
            "og kjent katet gitt ved {} og {} er
              {:.2f}.".\
        format(hypotenus, katet1, katet2))
```

```
else:
    print("Feil: Jeg skjønner ikke hva du vil jeg skal" + \
    "regne ut. Prøv igjen?")
```

Oppgave 16 *Spisse og butte trekanter*

En trekant kalles *spiss* dersom alle de tre vinklene til trekanten er spisse (altså $< 90°$), tilsvarende kalles trekanten *butt* dersom én av de tre vinkelene er butt (altså $> 90°$). En trekant vil alltid enten være spiss, rettvinklet, eller butt.

Dersom vi kaller de tre sidelengdene a, b og c og lar c være den lengste av disse kan vi vise at

$$a^2 + b^2 \begin{cases} > c^2 & \text{for spiss trekant,} \\ = c^2 & \text{for rettvinklet trekant,} \\ < c^2 & \text{for butt trekant.} \end{cases} \tag{3}$$

For rettvinklede trekanter er dette bare Pytagoras' læresetning. Men vi ser altså at likheten i Pytagoras' endres til ulikheter for spisse og butte trekanten.

Du skal nå lage et program der brukeren oppgir de tre sidelengdene i trekanten a, b og c og programmet skal si om trekanten er spiss, butt eller rettvinklet.

a) Forklar skjelettkoden under og fyll inn de bitene som mangler

```
print("Skriv inn de tre sidelengdene a, b og c der c skal være den lengste.")
a = float(input("Sidelengde a: "))
b = float(input("Sidelengde b: "))
c = float(input("Sidelengde c: "))

if c < a or c < b:
    raise ValueError("c må være den lengste sidelengden.")

if ...:
    print(...)

```

```
elif ...:
    print(...)

elif ...:
    print(...)
```

b) Vi gir nå sidelengdene til fem ulike trekanter. Hvilke to er spisse, hvilke to er butte og hvilken er rettvinklet?

- 4, 6, 8
- 3, 5, 5
- 5, 12, 13
- 6, 6, 8
- 9, 13, 17

Oppgave 17 *Hvilken skala?*

Et hus kan ha lengde 1 314 cm – eller, muligens på en mer hensiktsmessig skala, omtrent 13 meter. En celle kan være 0.000054 m lang – altså 54 μm. En fotballbane kan ha et areal på 6443399361 mm^2 – eller ca 6405 m^2.

I denne oppgaven skal vi, litt trinn for trinn, bygge opp et program for å konvertere *lengdemåleneheter* til en hensiktsmessig skala. Deretter vil vi gjerne utfordre dere på hvordan dere vil tenke for å utvikle et lignende program for å konvertere areal eller volum på lignende måte.

Vi anser at en målenhet er «hensiktsmessig» å bruke dersom målet kan oppgis med et tall større enn 0 foran kommaet, og at vi bruker størst mulig skala der dette er mulig. For å sjekke om man får et positivt tall foran kommaet (dvs. punktumet i Python) kan man konvertere tallet til et heltall ved hjelp av `int` og så sjekke om dette blir større enn 0.

a) Lag et Python-program som spør en bruker om hvor langt noe er. Lagre dette i en variabel. Husk å konvertere lengden til et desimaltall (ved hjelp av `float()`.

b) Spør deretter brukeren om hvilken målenhet dette er i. Her bør dere opplyse om hvilke mulige enheter det kan være – start med «m, cm eller mm».

c) Bruk betingelser, basert på målenheten brukeren gir, til å konvertere lengden brukeren har oppgitt til meter. Skriv ut svaret midlertidig og prøv noen ganger med regning for å sjekke at du har programmert dette riktig.

d) Regn ut hva lengden også blir i de andre enhetene (ut fra lengden i meter), og lagre disse i nye variable.

e) Bruk en betingelse for å først sjekke om det er hensiktsmessig å oppgi enheten i den største enheten (altså m). Hvis ja, skriv ut hva det blir i meter – hvis ikke, sjekk om det er hensiktsmessig å oppgi den i den neste (cm), osv. Hvis det hverken er hensiktsmessig å oppgi målenheten i m eller cm, skriv ut svaret i den siste mulige (mm).

f) Skriv ut svaret på en fin måte. Bestem dere for antall desimaler dere vil oppgi i svaret. Dere kan formatere tallet ved å skrive et kolon, et punktum, antall desimaler og *f* etter variabelen i utskriften, altså f.eks. `print(f"... {lengde_meter:.2f} ...")`.

g) Utvid programmet deres til å også akseptere km og μm som mulige enheter.

h) Finn på fem objekter med lengde, i forskjellige størrelsesordener, og prøv om programmet gir forventet resultat for disse.

i) Nå har dere skrevet et program som kan konvertere ulike lengder til en skala som (muligens) er mer hensiktsmessig for det man vil måle. Hvordan ville dere utviklet et tilsvarende program for konvertering av areal? Av volum? Hva ville vært likt som i dette programmet, og hva ville vært annerledes? Hva ville vært de matematiske utfordringene?

Diskuter spørsmålene over med en kollega eller i en gruppe. Det er også mulig å programmere dette for å utfordre seg selv – eller kanskje elevene?

Løsning oppgave 17 *Hvilken skala?*

a)

```
lengde = float(input("Hva er lengden til objektet? "))
```

b)

```
målenhet = input("Hva er målenheten? m, cm eller mm?")
```

c)

```
if målenhet=="m":
    lengde_meter = lengde
elif målenhet=="cm":
    lengde_meter = lengde/100
elif målenhet=="mm":
    lengde_meter = lengde/1000
else:
    print("Ukjent målenhet! Velg enten m, cm eller mm.")
```

d)

```
lengde_cm = lengde_meter*100
lengde_mm = lengde_cm*10
```

e)

```
if int(lengde_meter) > 0:
    print(f"Objektet er {lengde_meter:.2f} m langt.")
elif int(lengde_cm) > 0:
    print(f"Objektet er {lengde_cm:.2f} cm langt.")
else:
    print(f"Objektet er {lengde_mm:.2f} mm langt.")
```

f) Hele programmet ser nå slik ut:

```python
lengde = float(input("Hva er lengden til objektet? "))
målenhet = input("Hva er målenheten? km, m, cm, mm eller um?")

if målenhet=="km":
    lengde_meter = lengde*1000
elif målenhet=="m":
    lengde_meter = lengde
elif målenhet=="cm":
    lengde_meter = lengde/100
elif målenhet=="mm":
    lengde_meter = lengde/1000
elif målenhet=="um":
    lengde_meter = lengde/1000000
else:
    print("Ukjent målenhet! Velg enten km, m, cm, mm eller um.")

lengde_km = lengde_meter/1000
lengde_cm = lengde_meter*100
lengde_mm = lengde_cm*10
lengde_um = lengde_mm*1000

if int(lengde_km) > 0:
    print(f"Objektet er {lengde_km:.2f} km langt.")
elif int(lengde_meter) > 0:
    print(f"Objektet er {lengde_meter:.2f} m langt.")
elif int(lengde_cm) > 0:
    print(f"Objektet er {lengde_cm:.2f} cm langt.")
elif int(lengde_mm) > 0:
    print(f"Objektet er {lengde_mm:.2f} mm langt.")
else:
    print(f"Objektet er {lengde_um:.2f} um langt.")
```

g) Vi forsøkte med 0.000001 km, 1000 m, 0.05 cm, 1500 mm og 20000 um, og fikk til svar (med to desimaler) 1.00 um, 1.00 km, 500 um, 1.5 m og 2.00 cm.

Når man tester slik bør man skrive inn tall slik at man kommer inn i alle betingelsene (alle if-testene), slik at mulige feil i alle disse kan fanges opp. Man vil helst teste både for programmeringsfeil og logiske feil.

Oppgave 18 *Finn riktige typer fliser*

Vi skal legge fliser på kjøkkenet, og har en flate på 75×350 cm^2 som skal dekkes. Fra en butikk har vi funnet et utvalg av fliser som vi synes er fine, og nå vil vi sjekke om vi kan bruke noen av disse slik at vi får hele fliser hele veien – altså om noen av de har mål som passer akkurat både i høyden og bredden. For enkelthets skyld antar vi at flisene legges helt tett, det vil si at vi ikke trenger å beregne plass til fugemasse imellom.

Flisene vi har funnet har følgende mål:

1. 25×15 cm^2
2. 25×25 cm^2
3. 23×10 cm^2
4. 30×60 cm^2
5. 9×9 cm^2
6. 6×12 cm^2

For å sjekke om antall fliser vil passe én vei kan vi bruke *modulo*-operatoren, som vil gi oss *resten* i et delestykke. I Python bruker man `%` til dette. For eksempel vil

```
1 print(5 % 2)
```

gi oss

```
1
```

siden 5 går 2 ganger opp i 2, med rest *1*. Vi kan sjekke om antall fliser passer akkurat ved å sjekke at vi får rest *0*.

a) Lagre alle målene i variabler.

b) Bruk modulooperatoren for å sjekke om flis nr. 1 vil passe akkurat i *høyden*. Skriv programmet ditt slik at den både skriver ut en beskjed om flisen passer, og om den ikke gjør det.

c) Bruk modulooperatoren og den logiske operatoren `and` for å sjekke om flis nr. 1 også vil passe akkurat i bredden.

d) Gjør det samme for flis nummer 2 (her kan du kopiere koden din fra a og b, og så endre variabelnavnene).

e) Dersom antall fliser passer akkurat, bruk heltallsdivisjon (`//`) for å finne ut hvor mange fliser vi trenger totalt, og skriv ut resultatet. Legg inn dette både for flis 1 og flis 2.

f) Endelig skal vi sjekke alle flisene: Gjenta (kopier) koden over så mange ganger som nødvendig, og sjekk for hver flis om den kommer til å passe eller ikke.

g) Vi vurderer noen andre mål. Hva skjer om man endrer målene på flaten til 90×360 cm^2? Eller 81×360 cm^2? Vil noen av fliskombinasjonene passe da? Er det flere som kan passe?

Løsning oppgave 18 *Finn riktige typer fliser*

a)

```
1  høyde_totalt = 75          # cm
2  bredde_totalt = 350        # cm
3
4  høyde_flis1 = 25
5  bredde_flis1 = 15
6
7  høyde_flis2 = 25
8  bredde_flis2 = 25
```

```
9
10 høyde_flis3 = 23
11 bredde_flis3 = 10
12
13 høyde_flis4 = 30
14 bredde_flis4 = 60
15
16 høyde_flis5 = 9
17 bredde_flis5 = 9
18
19 høyde_flis6 = 6
20 bredde_flis6 = 12
```

b)

```
1 if (høyde_totalt % høyde_flis1 == 0):
2     print(f"Flis nr 1 vil passe akkurat i høyden {
          høyde_totalt} cm.")
3 else:
4     print(f"Flis nr 1, med høyde {høyde_flis1} cm,
          vil ikke passe inn i høyden {høyde_totalt}
          cm.")
```

```
1 if (høyde_totalt % høyde_flis1 == 0 and
     bredde_totalt % bredde_flis1 == 0):
2     print(f"Flis nr 1 vil passe akkurat i målene {
          høyde_totalt} x {bredde_totalt} cm.")
3 else:
4     print(f"Flis nr 1, med høyde {høyde_flis1} cm
          og bredde {bredde_flis1}, vil ikke passe
          inn i målene {høyde_totalt} x {
          bredde_totalt} cm.")
```

c)

```
1 if (høyde_totalt % høyde_flis2 == 0 and
     bredde_totalt % bredde_flis2 == 0):
2     print(f"Flis nr 2 vil passe akkurat i målene {
          høyde_totalt} x {bredde_totalt} cm.")
3 else:
```

```
    print(f"Flis nr 2, med høyde {høyde_flis2} cm
       og bredde {bredde_flis2}, vil ikke passe
       inn i målene {høyde_totalt} x {
       bredde_totalt} cm.")
```

d)

```

if (høyde_totalt % høyde_flis1 == 0 and
   bredde_totalt % bredde_flis1 == 0):
    print(f"Flis nr 1 vil passe akkurat i målene {
       høyde_totalt} x {bredde_totalt} cm.")

    antall_høyde = høyde_totalt // høyde_flis1
    antall_bredde = bredde_totalt // bredde_flis1
    antall = antall_høyde*antall_bredde

    print(f"Totalt trenger vi {antall} fliser for
       å dekke hele flaten.")

else:
    print(f"Flis nr 1, med høyde {høyde_flis1} cm
       og bredde {bredde_flis1}, vil ikke passe
       inn i målene {høyde_totalt} x {
       bredde_totalt} cm.")

print("")       # bare for å få litt mellomrom i
   utskriften

if (høyde_totalt % høyde_flis2 == 0 and
   bredde_totalt % bredde_flis2 == 0):
    print(f"Flis nr 2 vil passe akkurat i målene {
       høyde_totalt} x {bredde_totalt} cm.")

    antall_høyde = høyde_totalt // høyde_flis2
    antall_bredde = bredde_totalt // bredde_flis2
    antall = antall_høyde*antall_bredde

    print(f"Totalt trenger vi {antall} fliser for
       å dekke hele flaten.")
```

```
24
25  else:
26      print(f"Flis nr 2, med høyde {høyde_flis2} cm
           og bredde {bredde_flis2}, vil ikke passe
           inn i målene {høyde_totalt} x {
           bredde_totalt} cm.")
27
28  print("")
```

e)

```
1       if (høyde_totalt % høyde_flis3 == 0 and
           bredde_totalt % bredde_flis3 == 0):
2       print(f"Flis nr 3 vil passe akkurat i målene {
           høyde_totalt} x {bredde_totalt} cm.")
3
4       antall_høyde = høyde_totalt // høyde_flis3
5       antall_bredde = bredde_totalt // bredde_flis3
6       antall = antall_høyde*antall_bredde
7
8       print(f"Totalt trenger vi {antall} fliser for
           å dekke hele flaten.")
9
10  else:
11      print(f"Flis nr 3, med høyde {høyde_flis3} cm
           og bredde {bredde_flis3}, vil ikke passe
           inn i målene {høyde_totalt} x {
           bredde_totalt} cm.")
12
13  print("")
14
15
16
17  if (høyde_totalt % høyde_flis4 == 0 and
       bredde_totalt % bredde_flis4 == 0):
18      print(f"Flis nr 4 vil passe akkurat i målene {
           høyde_totalt} x {bredde_totalt} cm.")
19
20      antall_høyde = høyde_totalt // høyde_flis4
21      antall_bredde = bredde_totalt // bredde_flis4
```

```
22      antall = antall_høyde*antall_bredde
23
24      print(f"Totalt trenger vi {antall} fliser for
            å dekke hele flaten.")
25
26  else:
27      print(f"Flis nr 4, med høyde {høyde_flis4} cm
            og bredde {bredde_flis4}, vil ikke passe
            inn i målene {høyde_totalt} x {
            bredde_totalt} cm.")
28
29  print("")
30
31
32  if (høyde_totalt % høyde_flis5 == 0 and
        bredde_totalt % bredde_flis5 == 0):
33      print(f"Flis nr 5 vil passe akkurat i målene {
            høyde_totalt} x {bredde_totalt} cm.")
34
35      antall_høyde = høyde_totalt // høyde_flis5
36      antall_bredde = bredde_totalt // bredde_flis5
37      antall = antall_høyde*antall_bredde
38
39      print(f"Totalt trenger vi {antall} fliser for
            å dekke hele flaten.")
40
41  else:
42      print(f"Flis nr 5, med høyde {høyde_flis5} cm
            og bredde {bredde_flis5}, vil ikke passe
            inn i målene {høyde_totalt} x {
            bredde_totalt} cm.")
43
44  print("")
45
46
47
48  if (høyde_totalt % høyde_flis6 == 0 and
        bredde_totalt % bredde_flis6 == 0):
49      print(f"Flis nr 6 vil passe akkurat i målene {
            høyde_totalt} x {bredde_totalt} cm.")
```

```
50
51      antall_høyde = høyde_totalt // høyde_flis6
52      antall_bredde = bredde_totalt // bredde_flis6
53      antall = antall_høyde*antall_bredde
54
55      print(f"Totalt trenger vi {antall} fliser for
            å dekke hele flaten.")
56
57  else:
58      print(f"Flis nr 6, med høyde {høyde_flis6} cm
            og bredde {bredde_flis6}, vil ikke passe
            inn i målene {høyde_totalt} x {
            bredde_totalt} cm.")
```

f) Vi endrer på `høyde_totalt` og `bredde_totalt`, og finner ut at for 90×360 vil flis 4–6 passe, og for 81×360 vil flis 5 passe.

2.3 Naturfag

Oppgave 19 *Vann og sjokolade ved forskjellige temperaturer.*

Smeltepunktet til et stoff markerer den temperaturen som gjør at stoffet endrer fasetilstand mellom fast og flytende form. Vann har smeltepunkt ved 0 grader celsius, og sjokolade har smeltepunkt ved ca. 40 grader celsius.

Lag et program som spør om en temperatur, og lag `if-tester` for å sjekke hvilken fasetilstand vann og sjokolade er i ved denne temperaturen. Print til slutt ut en setning som nevner temperaturen og tilhørende fasetilstand for stoffene.

Løsning oppgave 19 *Vann og sjokolade ved forskjellige temperaturer.*

```
1  temperatur = int(input("Hva er temperaturen?"))
2
3  if temperatur <= 0:
4    fasetilstand_vann = "fast"
5    fasetilstand_sjokolade = "fast"
```

```
elif temperatur > 0 and temperatur < 50:
  fasetilstand_vann = "flytende"
  fasetilstand_sjokolade = "fast"
elif temperatur >= 50 and temperatur < 100:
  fasetilstand_vann = "flytende"
  fasetilstand_sjokolade = "flytende"

print(f"Ved {temperatur} grader er vann {
   fasetilstand_vann} og sjokolade {
   fasetilstand_sjokolade}.")
```

Oppgave 20 *Beskrive tilstanden til kjemiske stoff*

En venn av deg har lagd et program som forteller deg om vann fryser eller ikke basert på temperatur. Men det finnes jo flere stoffer enn vann, og du har lyst til å lage et program som kan fortelle deg om de er fast form, væske eller gass.

Programmet til vennen din ser du her:

```
kokepunkt = 100  # grader Celcius
frysepunkt = 0  # grader Celcius
stoff = 'vann'

temperatur = float(input(f'Hvilken temperatur har {
   stoff}?'))

if temperatur < frysepunkt:
    print('Det er et fast stoff')
elif temperatur < kokepunkt:
    print('Det er en væske')
else:
    print('Det er en gass')
```

a) Utvid programmet til at du bruker `input` funksjonen til å spørre brukeren om hvilket stoff den vil vite fasen til.

b) Bruk en `if` og en betingelse til å sjekke om brukeren skriver vann.

Hvis brukeren skrev vann skal `kokepunkt` variabelen settes til 100 og `frysepunkt` variabelen settes til 0. Hvis brukeren skrev noe annet enn `'vann'` skal `'Jeg vet ikke om dette stoffet'` skrives ut til brukeren. NB: Hvis brukeren skriver inn noe annet enn vann så vil denne beskjeden printes ut, og så vil vi få en feilmelding!

c) Nå må vi passe på at programmet avsluttes uten å krasje dersom brukeren skriver inn noe annet enn vann. For å gjøre det trenger vi `exit` funksjonen fra `sys` biblioteket. Start derfor programmet ditt med å hente denne funksjonen ved å skrive `from sys import exit`.

d) Modifiser `else`-blokken til programmet ditt, slik at du kaller på funksjonen `exit()` etter at `'Jeg vet ikke om dette stoffet'` skrives ut til brukeren. Test programmet ditt, hva skjer om du skriver inn et stoff det ikke kjenner til (f.eks. saltvann)

e) Legg til en mulighet for å skrive *håndsprit*. Håndsprit fryser på ca -89 grader og koker på ca 83 grader.

f) Utvid programmet til å støtte kvikksølv og luft. Kvikksølv fryser på ca -39 grader og koker på ca 357 grader, luft fryser på ca -210 grader og koker på ca -196 grader.

Løsning oppgave 20 *Beskrive tilstanden til kjemiske stoff*

Under ser vi hele programmet som ble lagd i oppgaven over, og hver nye linje har en kommentar som sier hvilken deloppgave den ble skrevet i.

```
1  from sys import exit                                    # c)
2
3  stoff = input("Hvilket stoff har vi?")                  # a)
4
5  if stoff == 'vann':                                     # b)
6      kokepunkt = 100                                     # b)
7      frysepunkt = 0                                      # b)
8  elif stoff == 'håndsprit':                              # e)
9      frysepunkt = -89                                    # e)
10     kokepunkt = 83                                      # e)
11 elif stoff == 'kvikksølv':                              # f)
12     frysepunkt = -39                                    # f)
13     kokepunkt = 357                                     # f)
```

```
elif stoff == 'luft':                                   # f)
    frysepunkt = -210                                   # f)
    kokepunkt = -196                                    # f)
else:                                                   # b)
    print('Jeg kjenner ikke til dette stoffet.')        # b)
    exit()                                              # d)


temperatur = float(input(f'Hvilken temperatur har {stoff}'))

if temperatur < frysepunkt:
    print('Det er et fast stoff')
elif temperatur < kokepunkt:
    print('Det er en væske')
else:
    print('Det er en gass')
```

3 Løkker

3.1 Grunnleggende programmering

Oppgave 21 *For-løkker for hånd*

For hver løkke, gå igjennom for hånd og forutsi hva som skrives ut. Etterpå kan du kjøre løkkene og se hva om du hadde rett:

a)

```
for tall in range(1, 5):
    print(tall)
```

b)

```
sum = 0
for tall in range(1, 5):
    sum += tall
    print(sum)
```

c)

```
produkt = 0
for tall in range(1, 5):
    produkt *= tall
    print(produkt)
```

d)

```
produkt = 1
for tall in range(1, 5):
    produkt *= tall
    print(produkt)
```

e)

```python
x = 1
for _ in range(10):
    x *= 2
    print(x)
```

Løsning oppgave 21 *For-løkker for hånd*

a)

```
1
2
3
4
```

b)

```
1
3
6
10
```

c)

```
0
0
0
0
```

d)

```
1
2
6
24
```

e)

```
2
4
8
16
32
64
128
256
512
1024
```

3.2 Matematikk

Oppgave 22 *Renter*

Bank 1 gir fast 3 prosent rente på sin sparekonto. Bank 2, derimot, gir 3,3 prosent rente de første fem årene før de skifter til 2,8 prosent rente. Du skal sette 10000,– i en bank i morgen. I denne oppgaven skal du bruke `for`-løkker til å simulere hva som skjer med pengene i de ulike bankene.

a) Hvilken bank er best å bruke hvis du skal spare i ti år?

b) Hvor lenge må du ha pengene i bank 1 for at det skal lønne seg fremfor bank 2?

Naboen din bestemmer seg for å heller sette inn 1000,– hver januar, istedenfor å sette inn en engangssum slik som du gjør.

c) (Bonusoppgave) Hvilken bank er det best for naboen din å bruke hvis han skal spare i ti år?

d) (Bonusoppgave) Hvor lenge må han ha pengene i bank1 for at det skal lønne seg fremfor bank 2?

Løsning oppgave 22 *Renter*

a)

```
rente_bank_1 = 3/100
rente_bank_2_første_fem_år = 3.3/100
rente_bank_2_siste_fem_år = 2.8/100

inskudd = 10_000

penger_i_bank_1 = inskudd
penger_i_bank_2 = inskudd

for år in range(10):
    penger_i_bank_1 *= (1 + rente_bank_1)
    if år < 5:
        penger_i_bank_2 *= (1 + rente_bank_2_fø
           rste_fem_år)
    else:
        penger_i_bank_2 *= (1 + rente_bank_2
           _siste_fem_år)

print(f"Etter 10 år står det {penger_i_bank_1:.2f}
    kr i bank 1, og {penger_i_bank_2:.2f} kr i
   bank 2.")
# >>> Etter 10 år står det 13439.16 kr i bank 1,
   og 13504.15 kr i bank 2.
```

b)

```
rente_bank_1 = 3/100
rente_bank_2_første_fem_år = 3.3/100
rente_bank_2_siste_fem_år = 2.8/100

inskudd = 10_000

penger_i_bank_1 = inskudd
penger_i_bank_2 = inskudd
år = 0
while penger_i_bank_1 <= penger_i_bank_2:
    penger_i_bank_1 *= (1 + rente_bank_1)
    if år < 5:
```

```
        penger_i_bank_2 *= (1 + rente_bank_2_før
            rste_fem_år)
    else:
        penger_i_bank_2 *= (1 + rente_bank_2
            _siste_fem_år)
    år += 1

print(f"Etter {år} år står det {penger_i_bank_1:.2
    f} kr i bank 1, og {penger_i_bank_2:.2f} kr i
    bank 2.")
#Etter 13 år står det 14685.34 kr i bank 1, og 146
    70.55 kr i bank 2.
```

c)

```
rente_bank_1 = 3/100
rente_bank_2_første_fem_år = 3.3/100
rente_bank_2_siste_fem_år = 2.8/100

inskudd = 0
årlig_inskudd = 1000

penger_i_bank_1 = inskudd
penger_i_bank_2 = inskudd

for år in range(10):
    penger_i_bank_1 += årlig_inskudd
    penger_i_bank_2 += årlig_inskudd
    penger_i_bank_1 *= (1 + rente_bank_1)
    if år < 5:
        penger_i_bank_2 *= (1 + rente_bank_2_før
            rste_fem_år)
    else:
        penger_i_bank_2 *= (1 + rente_bank_2
            _siste_fem_år)

print(f"Etter 10 år står det {penger_i_bank_1:.2f}
    kr i bank 1, og {penger_i_bank_2:.2f} kr i
    bank 2.")
# >>> Etter 10 år står det 11807.80 kr i bank 1,
```

```
   og 11770.25 kr i bank 2.
```

d)

```
rente_bank_1 = 3/100
rente_bank_2_første_fem_år = 3.3/100
rente_bank_2_siste_fem_år = 2.8/100

inskudd = 1000
årlig_inskudd = 1000

penger_i_bank_1 = inskudd
penger_i_bank_2 = inskudd
år = 0
while penger_i_bank_1 <= penger_i_bank_2:

    penger_i_bank_1 *= (1 + rente_bank_1)
    if år < 5:
        penger_i_bank_2 *= (1 + rente_bank_2_fø
           rste_fem_år)
    else:
        penger_i_bank_2 *= (1 + rente_bank_2
           _siste_fem_år)

    penger_i_bank_1 += årlig_inskudd
    penger_i_bank_2 += årlig_inskudd
    år += 1
print(f"Etter {år} år står det {penger_i_bank_1:.2
   f} kr i bank 1, og {penger_i_bank_2:.2f} kr i
   bank 2.")
# >>> Etter 9 år står det 11463.88 kr i bank 1, og
    11449.66 kr i bank 2.
```

Oppgave 23 *Kjøpe telefon på kreditt*

Hallgeir har veldig lyst på en ny telefon som koster 2999 kroner. Problemet er bare at han har brukt opp sparepengene sine på andre ting. For å få kjøpt telefonen skaffer Hallgeir et kredittkort som har 30% rente. Etter å ha kjøpt telefonen går det tre år før Hallgeir betaler tilbake kreditten. I denne oppgaven

skal vi undersøke hvor mye Hallgeir blir nødt til å betale da.

a) Opprett variablene `renter`, `originalpris` og `antall_år` og gi med verdiene `30`, `2999` og `3`

b) Regn ut vekstfaktoren til renta og lagre den i en variabel `rentevekstfaktor`

c) Opprett en variabel, `lån`. Denne variabelen skal holde orden på hvor stort lånet til Hallgeir er. Til å begynne med er lånet like stort som prisen på telefonen. Sett altså `lån`-variabelen til å ha verdien `2999`

d) Bruk en `for`-løkke til å simulere hvordan lånet vokser for hvert år.
Hint: for hvert år skal lånet ganges med vekstfaktoren du regnet ut i oppgave b)

e) Oppdater programmet ditt til å skrive ut størrelsen på lånet for hvert år. Hvor mye skylder Hallgeir etter tre år?

f) Hvor mye ekstra kostet telefonen i forhold til originalprisen?

g) Gå inn på `https://kredittkort.com/` og se hvilket kort som kommer øverst og noter deg renta. Endre renta i programmet ditt til å matche denne renta. Gå inn på `https://www.prisjakt.no/category.php?k=103` og se hvilken telefon som er mest populær. Endre orginalprisen i programmet ditt til å matche prisen til den telefonen. Kjør programmet nå. Hva ville denne telefonen og dette kredittkortet kostet Hallgeir?

Løsning oppgave 23 *Kjøpe telefon på kreditt*

a)

```
1 renter = 30
2 originalpris = 2999
3 antall_år = 3
```

b)

```
rentevekstfaktor = 1 + renter/100
```

c)

```
lån = originalpris
```

d)

```
for år in range(antall_år):
    lån *= rentevekstfaktor

print(f"Hallgeir skylder {lån:.2f} kroner etter {
   antall_år} år")
```

```
Hallgeir skylder 6588.80 kroner etter 3 år
```

e) Det kostet 3589.80 kroner mer enn originalprisen å kjøpe telefonen på kreditt.

f) Det kredittkortet med lavest rente har en rente på $23,1\%$

g) Den vanligste telefonen er en iPhone 11 64GB og koster 7990 kroner. Om vi bruker disse tallene i koden får vi at telefonen koster $14904,62$ kroner, det er nesten dobbelt så mye som originalprisen!

Oppgave 24 *Gangetabell-quiz*

a) Lag et program som trekker to tilfeldige tall og spør brukeren om hva produktet av dem er. (Hint: legg til linjen `import random` på første linje. Da vil `tilfeldig_tall =random.randint(1,10)` være et tilfeldig tall mellom 1 og 10.)

b) Modifiser programmet ditt slik at du sjekker om svaret var riktig, og om så var tilfelle så får brukeren et poeng. (Hint: lag en variabel `poeng =0` som du øker med en hvis svaret er rett.)

c) Bruk det du har programmert så langt sammen med en `while`-løkke. La

programmet spørre brukeren flere ganger om hva produktet av to tall er, og hvis svaret er riktig så får brukeren et poeng. Programmet skal slutte når brukeren får mindre enn - 10 poeng eller mer enn 10 poeng.

d) Skriv ut en melding helt til slutt om brukeren vant (mer enn 10 poeng) eller tapte (mindre enn -10 poeng).

Løsning oppgave 24 *Gangetabell-quiz*

a)

```
import random

tilfeldig_tall1 = random.randint(1,10)
tilfeldig_tall2 = random.randint(1,10)

svar = int(input(f"Hva er produktet av {
   tilfeldig_tall1} og {tilfeldig_tall2}? >>> "))
```

```
import random

tilfeldig_tall1 = random.randint(1,10)
tilfeldig_tall2 = random.randint(1,10)
poeng = 0

svar = int(input(f"Hva er produktet av {
   tilfeldig_tall1} og {tilfeldig_tall2}? >>> "))

if svar == tilfeldig_tall1*tilfeldig_tall2:
    print("Det er riktig!")
    poeng += 1
```

b)

```
import random


poeng = 1

while poeng < 10 and poeng > -10:
```

```
    tilfeldig_tall1 = random.randint(1,10)
    tilfeldig_tall2 = random.randint(1,10)

    svar = int(input(f"Hva er produktet av {
       tilfeldig_tall1} og {tilfeldig_tall2}? >>>
       "))

    if svar == tilfeldig_tall1*tilfeldig_tall2:
        poeng += 1
        print(f"Det er riktig! Du har {poeng}
           poeng")
    else:
        poeng -= 1
        print(f"Det var nok feil. Du har {poeng}
           poeng")
```

d) Fortsetter i samme program som i forrige deloppgave:

```
if poeng >= 10:
    print("Du vant!")
else:
    print("Du tapte. Du må nok øve mer på
       gangetabellen.")
```

Oppgave 25 *Tverrsum*

Tverrsummen av et tall er tallet du får dersom du plusser alle sifrene i tallet med hverandre.

a) Hvor mange 3-sifrede tall finnes det som har tverrsum 5? Løs for hånd.

b) Skriv et program som finner alle 3-sifrede tall med tverrsum 4. Start med å skrive en løkke for alle tall fra 1 (hvorfor fra 1?) til 9; i denne løkken lager du enda en løkke; og i denne løkken trenger du enda en. I den innerste løkken trenger du en betingelse, og til slutt en `print`.

c) Legg inn en teller i programmet ditt, som først er 0, og deretter øker hver gang du finner et tall med tverrsum 4. Hvor mange tall er det tilsammen? Stemmer det med det du fikk når du regnet for hånd?

d) Tror du det er flest 3-sifrede tall med tverrsum 4, eller flest 4-sifrede tall med tverrsum 3 – eller er det like mange? Utvid programmet ditt til å telle begge deler, og sjekk om du hadde rett.

Løsning oppgave 25 *Tverrsum*

a) Vi kan løse dette ved å gå frem systematisk. Vi kan starte med 103, deretter finne 112, osv.; til slutt får vi tallrekken *103, 112, 121, 130, 202, 211, 220, 301, 310, 400*. Dette utgjør 10 tall.

b)

```
tverrsum = 4

for i in range(1, 9):
  for j in range(9):
    for k in range(9):
        if (i + j + k) == tverrsum:
          print(f"{i}{j}{k}")
```

c) Tar bort print for å få en mer oversiktlig output:

```

teller = 0
tverrsum = 4

for i in range(1, 9):
  for j in range(9):
    for k in range(9):
        if (i + j + k) == tverrsum:
          teller += 1

print(f"Det er {teller} 3-sifrede tall med
   tverrsum {tverrsum}.")
```

d)

```

teller = 0
```

```
tverrsum = 4

for i in range(1, 9):
  for j in range(9):
    for k in range(9):
        if (i + j + k) == tverrsum:
          print(f"{i}{j}{k}")
          teller += 1

print(f"Det er {teller} 3-sifrede tall med tverrsum {tverrsum}.")

teller = 0
tverrsum = 3

for i in range(1, 9):
  for j in range(9):
    for k in range(9):
      for m in range(9):
        for n in range(9):
          for o in range(9):
            if (i + j + k + m + n + o) == tverrsum:
              teller += 1

print(f"Det er {teller} 4-sifrede tall med tverrsum {tverrsum}.")
```

e) Det er faktisk like mange. Tror du det samme gjelder for antall 3-sifrede tall med tverrsum 5 / antall 5-sifrede tall med tverrsum 3?

Oppgave 26 *Fibonacci rekken*

Fibonacci rekken er en kjent tallfølge som naturlig oppstår mange steder i naturen. Tallfølgen går som følger:

$$1, 1, 2, 3, 5, 8, 13, 21, ... \tag{4}$$

Vi kan skrive Fibonacci rekken som en rekursiv tallfølge etter denne formelen:

$$a_n = a_{n-1} + a_{n-2}, \tag{5}$$

hvor $a_0 = 1$ og $a_1 = 1$.

En interessant egenskap ved Fibonacci rekken er at forholdet mellom to etterfølgende tall i følgen går mot det gyldne snitt, $\phi = \frac{1+\sqrt{5}}{2} \approx 1.62$. Det vil si at

$$\frac{a_n}{a_{n-1}} \to \phi. \tag{6}$$

a) Du skal nå lage et program som skriver ut de første 10 tallene i Fibonacci rekka. Start med å opprette en variabel `forrige_tall =1` og en variabel `fibonacci_tall =1`.

b) Skriv så ut `forrige_tall` og `fibonacci_tall` til brukeren av programmet.

c) Bruk en `for`-løkke som repeteres 8 ganger (10 tall - 2 start-tall) som skriver ut de resterende 8 tallene i Fibonacci-rekka. HINT: Her kan det være lurt å opprette det nye Fibonacci tallet i en variabel `nytt_fibonacci_tall` før du oppdaterer verdien til `forrige_tall` og `fibonacci_tall`.

d) Endre programmet slik at du bruker input til å be brukeren om hvor mange Fibonacci tall du skal skrive ut til skjermen.

e) Oppdater programmet slik at du og skriver ut forholdet mellom `forrige_tall` og `fibonacci_tall`. Blir dette forholdet ca lik 1.62?

f) Endre start-tallene fra $a_0 = 1$ og $a_1 = 1$ til noe annet (f.eks $a_0 = 2$ og $a_1 = 1$), hvordan endrer det oppførselen til rekka?

Løsning oppgave 26 *Fibonacci rekken*

a)

```
1 forrige_tall = 1
2 fibonacci_tall = 1
```

b)

```
print(forrige_tall)
print(fibonacci_tall)
```

c)

```
for tallnummer in range(8):
    nytt_fibonacci_tall = fibonacci_tall +
       forrige_tall
    forrige_tall = fibonacci_tall
    fibonacci_tall = nytt_fibonacci_tall
    print(fibonacci_tall)
```

d)

```
antall_tall = int(input('Hvor mange Fibonacci tall
    ønsker du? '))

forrige_tall = 1
fibonacci_tall = 1
print(forrige_tall)
print(fibonacci_tall)

for tallnummer in range(antall_tall - 2):
    nytt_fibonacci_tall = fibonacci_tall +
       forrige_tall
    forrige_tall = fibonacci_tall
    fibonacci_tall = nytt_fibonacci_tall
    print(fibonacci_tall)
```

e)

```
for tallnummer in range(antall_tall - 2):
    nytt_fibonacci_tall = fibonacci_tall +
       forrige_tall
    forrige_tall = fibonacci_tall
    fibonacci_tall = nytt_fibonacci_tall
    forhold = fibonacci_tall / forrige_tall
```

```
6       print(f'Fibonacci tall: {fibonacci_tall},
            forhold mellom nåværende og forrige
            Fibonacci tall: {forhold}')
```

f) Vi ser at forholdet fortsatt går mot det gyldne snitt. I tillegg, hvis vi bruker $a_0 = 2$ og $a_1 = 1$ får vi de mindre kjente, men mer interessante *Lucas tallene*. Du kan lære mer om Lucas Tallene i denne fine YouTube videoen https://www.youtube.com/watch?v=PeUbRXnbmms av Numberphile (Brady Haran) og Matt Parker.

```
1  antall_tall = int(input('Hvor mange Fibonacci tall
       ønsker du? '))
2
3  forrige_tall = 1
4  fibonacci_tall = 1
5  print(forrige_tall)
6  print(fibonacci_tall)
7
8  for tallnummer in range(antall_tall - 2):
9      nytt_fibonacci_tall = fibonacci_tall +
           forrige_tall
10     forrige_tall = fibonacci_tall
11     fibonacci_tall = nytt_fibonacci_tall
```

3.3 Naturfag

Oppgave 27 *Finne komplementær DNA-streng*

De to trådene som utgjør DNA-dobbelspiralen består av deoksyribose, en fosfatgruppe og de fire nitrogenbasene A, T, C og G. En base fra hver tråd er forbundet med en hydrogenbinding, og kombinasjonene er enten A og T eller C og G.

a) Lag en DNA-streng på 50 baser med tilfeldig rekkefølge av de 50 basene.

b) Lag en funksjon som kan ta inn en hvilken som helst DNA-streng og

generere den komplementære tråden.

Løsning oppgave 27 *Finne komplementær DNA-streng*

a)

```
from random import choice

baser = ["A", "T", "C", "G"]
DNA_streng = ""

for base in range(50):
  DNA_streng = DNA_streng + choice(baser)

print(DNA_streng)
```

b)

```
from random import choice

baser = ["A", "T", "C", "G"]
DNA_streng = ""

for base in range(50):
  DNA_streng = DNA_streng + choice(baser)

print(DNA_streng)

def DNA_generator(templat_tråd):
  komplementær_tråd = ""
  for base in templat_tråd:
    if base == "A":
      komplementær_tråd += "T"
    elif base == "T":
      komplementær_tråd += "A"
    elif base == "C":
      komplementær_tråd += "G"
    elif base == "G":
      komplementær_tråd += "C"
  return komplementær_tråd
```

```
23
24 print(DNA_generator(DNA_streng))
```

Oppgave 28 *Vektvekst for selunger*

Grøndlandsseler dier ungene sine intensivt de 12 første levedagene. Melken er veldig rik på fett, og selungene legger på seg ca. 2.2 kilo om dagen denne perioden. Nyfødt veier selungene 11 kg.

a) Bruk en løkke til å finne vekten for selungen de 12 første levedagene og legge de i en liste. Lag også en liste med dagene, 1-12. Print listene.

b) Finn fra listen over vekt

- vekten til selen på dag 12
- vektene for dagene 5-7

c) Lag et plot med vekt mot dager. Gi aksene og plottet et navn.

Løsning oppgave 28 *Vektvekst for selunger*

a)

```
vekt = 11
dag = 1
vekt_liste = [11]
dag_liste = [1]
while dag < 12:
  ny_vekt = vekt + 2.2
  vekt_liste.append(ny_vekt)
  vekt = ny_vekt
  dag = dag + 1
  dag_liste.append(dag)
print(vekt_liste)
print(dag_liste)
```

b)

- Vekt på dag 12

```
    print(vekt_liste[-1])
```

- Vektene for dagene 5-7

```
    print(vekt_liste[4:8])
```

```
from matplotlib.pyplot import *

plot(dag_liste, vekt_liste)
xlabel("Antall dager etter fødsel")
ylabel("Vekt")
title("Vekt hos sel de 12 første dagene etter fø
   dsel")
show()
```

4 Funksjoner

4.1 Grunnleggende programmering

c) **Oppgave 29** *Enkle funksjoner*

a) Lag en funksjon `pluss(a,b)` som tar inn to tall a og b og returnerer summen av dem.

b) Lag en funksjon `minus(a,b)` som tar inn to tall a og b og returnerer differansen av dem.

c) Lag en funksjon `kalkulator(operasjon,a,b)` som tar inn operasjon som enter er ”pluss"” eller ”"minus"” (i form av en tekststring), og to tall, og regner ut resultatet.

Løsning oppgave 29 *Enkle funksjoner*

```
1  def pluss(a,b):
2      return a+b
3
4  def minus(a,b):
5      return a-b
6
7  def kalkulator(operator,a,b):
8      if operator == "pluss":
9          return a+b
10     if operator == "minus":
11         return a-b
```

4.2 Matematikk

Oppgave 30 *Faktorisering*

Alle positive heltall er enten primtall, eller kan faktoriseres til et produkt av

flere primtall, som i

$$15 = 3 \cdot 5$$

$$20 = 2 \cdot 2 \cdot 5$$

Her skal vi lage et program som finner faktoriseringen av et positivt heltall for oss.

a) *Modulo*-operasjonen forteller oss om et tall er delelig med et annet tall eller ikke. Denne gir oss resten ved en divisjon, og hvis *a modulo b* (syntaks `a % b` i Python) er lik 0, er a delelig på b. Skriv en funksjon `delelig_paa` som tar inn to tall a og b, og returnerer `True` hvis a er delelig på b, `False` ellers.

b) Skriv en funksjon `finn_neste_faktor` som tar inn et heltall og returnerer det minste tallet (som er større enn 1) dette er delelig på. For eksempel vil man at `finn_neste_faktor(20)` skal returnere 2.

c) Skriv en funksjon `skriv_ut_faktorisering` som tar inn et heltall og sjekker om det er et primtall eller ikke – og hvis ikke, skriver ut primtallsfaktoriseringen av dette.

d) Hvorfor kan man ikke bruke programmet du skrev over til å finne alle primtall som finnes? Hva skjer hvis du gir større og større tall som input?

Løsning oppgave 30 *Faktorisering*

a)

```
1  def delelig_paa(a, b):
2      return a % b == 0
```

b)

```
1  def finn_neste_faktor(n):
2      for i in range(2, n):
3          if delelig_paa(n, i):
4              return i
5
6      return 1
```

c)

```
def print_faktorisering(n):

    siste_faktor = finn_neste_faktor(n)

    if siste_faktor == 1:
        print(f"{n} er et primtall!")
        return

    gjenvaerende = n
    faktoriseringsstr = ""

    while siste_faktor != 1:
        faktoriseringsstr += str(siste_faktor) + "
            * "
        gjenvaerende = gjenvaerende //
           siste_faktor
        siste_faktor = finn_neste_faktor(
           gjenvaerende)

    faktoriseringsstr += str(gjenvaerende)

    print(f"Primtallsfaktorisering av {n}: \
            {faktoriseringsstr}")
```

Merk at `//` gir *heltallsdivisjon*. Vi *kunne* ha skrevet f.eks. int(gjenvaerende / siste_faktor) isteden, og det fungerer som regel, siden vi bare skal ha heltall med her. Men dersom du prøver med veldig store tall vil du her oppleve at Python har strengere begrensninger på hvor store *float* kan være i forhold til *integers*.

d) Reflekteringsspørsmål – så mange svar kan være riktige. Rent teoreitsk går det ikke an å skrive ut uendelig mange tall i endelig tid. Det er også begrensninger på hvor store tall man kan lagre i en *float* i Python – det er sannsynligvis mange av dere kommer til å oppleve. Det er videre også begrenset hvor store tall du kan lagre på maskinen i hele tatt, du har begrenset minne. Selv om man får til å lagre tall på andre måter, så tar hver «sjekk» av delelighet og hvert forsøk på å finne *neste faktor litt* tid – for tall vi vanligvis jobber med merker man det ikke men etterhvert som tallene blir større vil dette også være merkbart.

4.3 Naturfag

Oppgave 31 *Elekstrisitet: Strøm – spenning – resistans*

Viktige egenskaper i en elektrisk krets er spenningen U, målt i volt, strømmen I, målt i ampere, og motstanden/resistansen R, målt i ohm (Ω). Forholdet mellom disse enhetene er gitt ved *Ohms lov*: $U = R \cdot I$

a) Lag tre funksjoner som regner ut strøm, spenning og resistansen i kretsen gitt at man vet de to andre størrelsene.

b) Bruke funksjonene dine til å regne ut:

- Spenningen når strømmen er 10 A og motstanden 1.7 Ω
- Resistansen når spenningen er 230 V og strømmen er 20 A
- Strømmen når spennignen er 5 V og motstanden er 200 Ω

c) Når flere motstander er koblet i seriekobling, blir den totale motstanden lik summen av alle motstandene i serien. $R_{tot} = R_1 + R_2 + \cdots + R_N$. Lag en funksjon `seriekobling` som tar inn en liste med motstander og returnerer den totale motstanden.

d) Dersom spenningen er 5 V, og motstandene på 10, 5, 2 og 11 Ω er koblet i serie, hva blir strømmen da?

e) **Utfordring:** Når motstander er koblet i parallell, er sammenhengen mellom dem $\frac{1}{R_{tot}} = \frac{1}{R_1} + \frac{1}{R_2} + \cdots + \frac{1}{R_N}$. Lag en funksjon `parallellkobling` som regner ut den totale motstanden i kretsen. Hva blir strømmen nå, dersom motstandene og spenningen er den samme?

Løsning oppgave 31 *Elekstrisitet: Strøm – spenning – resistans*

a) Vi definerer funksjonene:

```
1  def spenning(I,R):
2    return R*I
3
4  def strøm(U,R):
```

```
  return U/R

def motstand(U,I):
  return U/I
```

b) 17 V, 11.5 Ω, og 25 mA

c)

```
def seriekobling(motstander):
  tot_motstand = 0
  for motstand in motstander:
    tot_motstand += motstand
  return tot_motstand
```

d) Vi bruker funksjonene vi har laget:

```
motstander  = [10,5,2,11]
R = seriekobling(motstander)
U = 230
print(f"Strømmen er {strøm(U,R):.2f} A")

>>> Strømmen er 8.21 A
```

e) Vi lager en funksjone som er ganske lik som for seriekobling, men må dele på 1 i alle ledd:

```
def parallellkobling(motstander):
  tot_motstand_inv = 0
  for motstand in motstander:
    tot_motstand_inv += 1/motstand
  tot_motstand = 1/tot_motstand_inv
  return tot_motstand


motstander  = [10,5,2,11]
R = parallellkobling(motstander)
U = 230
print(f"Strømmen er {strøm(U,R):.2f} A")

>>> Strømmen er 204.91 A
```

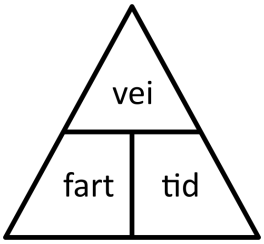
4.4 Fysikk

Oppgave 32 *Vei-fart-tid-kalkulator*

En viktig formel for å beskrive bevegelse er vei-fart-tid formelen,

$$\text{vei} = \text{tid} \cdot \text{fart}$$

Egentlig er det tre formler i ett, for vi kan stokke om på den avhengig av hva vi ønsker å regne ut. Av og til tegnes vei-fart-tid formelen som en pyramide:



a) Lag en funksjon som bruker vei-tid-fart-formelen gitt over, kall denne `vei(fart, tid)`. Vi vil at fart skal oppgis i kilometer i timen, vei i antall kilometer og tid i antall minutter – husk å regne om tiden fra antall timer til antall minutter.

b) Skriv opp uttrykket for å regne ut fart, gitt vei og tid. Lag en funksjon som bruker formelen, kall denne `fart(vei, tid)`.

c) Skriv opp uttrykket for å regne ut tid, gitt vei og fart. Lag en funksjon som bruker formelen, kall denne `tid(vei, fart)`.

d) Du skal nå lage en vei-fart-tid-kalkulator. Når dette programmet kjører skal det først spørre brukeren om de ønsker å regne ut vei, fart, eller tid. Avhengig av svaret skal så programmet spørre om de to andre størrelsene, og så regne ut og skrive ut svaret.

Løsning oppgave 32 *Vei-fart-tid-kalkulator*

a)

```
def fart(vei, tid):
    tid_i_timer = tid/60
    return vei/tid_i_timer
```

b)

```
def vei(fart, tid):
    tid_i_timer = tid/60
    return tid_i_timer*fart
```

c)

```
def tid(vei, fart):
    tid_i_timer = vei/fart
    return 60*tid_i_timer
```

d)

```
def input_fart():
    return float(input("Hvor fort, i km/t? "))

def input_vei():
    return float(input("Hvor langt, i km? "))

def input_tid():
    return float(input("Hvor lenge, i minutter? ")
        )

regnut = input("Skriv inn 'f' for å regne ut fart,
    " + \
        "'v' for å regne ut vei " + \
        "og 't' for å regne ut tid:\n")

if regnut=="f":
    print("Regner ut fart!")
    v = input_vei()
    t = input_tid()
    f = fart(v, t)

    print(f"Fart: {f} km / t")

elif regnut=="v":

```

```
24      print("Regner ut vei!")
25      f = input_fart()
26      t = input_tid()
27      v = vei(f, t)
28
29      print(f"Vei: {v} km")
30
31  elif regnut=="t":
32
33      print("Regner ut tid!")
34      v = input_vei()
35      f = input_fart()
36      t = tid(v, f)
37
38      print(f"Tid: {t} minutter")
39  regnut = input("Skriv inn 'f' for å regne ut fart,
        " + \
40          "'v' for å regne ut vei " + \
41          "og 't' for å regne ut tid:\n")
42
43  if regnut=="f":
44      print("Regner ut fart!")
45      v = input_vei()
46      t = input_tid()
47      f = fart(v, t)
48
49      print(f"Fart: {f} km / t")
50
51  elif regnut=="v":
52
53      print("Regner ut vei!")
54      f = input_fart()
55      t = input_tid()
56      v = vei(f, t)
57
58      print(f"Vei: {v} km")
59
60  elif regnut=="t":
61
62      print("Regner ut tid!")
```

```
63      v = input_vei()
64      f = input_fart()
65      t = tid(v, f)
66
67      print(f"Tid: {t} minutter")
68
69  else:
70     print("Ukjent alternativ   prøv igjen!")
```

Oppgave 33 *Hvor mye koster morgendusjen din?*

For å varme opp vann trenger vi energi, og i Norge kommer den energien vanligvis fra strøm. Enheten som brukes for å måle hvor mye strøm vi har brukt er gjerne kilowattimer (kWh), som representerer at du har brukt 1000 Watt i en time.

Til vanlig måler vi energiforbruk i Joule (J), så derfor kan det være nyttig å ha en funksjon som oversetter energi fra kWh til J og motsatt. Det kan vi gjøre med denne formelen

$$[\text{kWh}] = 3.6[\text{MJ}], \tag{7}$$

hvor MJ er megajoule, eller 10^6 J.

Når vi varmer opp vann, så bruker vi 4.2 J per liter (L) vann. For å gjøre enhetsomregning lettere er det derfor vanlig å snakke om enheten *kalorier* (cal). En kalori representerer mengden energi som skal til for å varme opp en liter vann en grad Celcius (°C). Altså er en kalori gitt ved formelen

$$[\text{cal}] = 4.2[\text{kJ}], \tag{8}$$

hvor kJ er kilojoule, eller 1000 J.

a) Lag en funksjon `kWh_til_joule(energi_i_kWh)` som oversetter en energimengde oppgitt i kWh til en energimengde oppgitt i Joule.

b) Lag en funksjon `joule_til_kWh(energi_i_J)` som oversetter en energimengde oppgitt i Joule til en energimengde oppgitt i kWh.

c) Lag en funksjon `kalori_til_joule(energi_i_cal)` som oversetter en energimengde oppgitt i kalorier til en energimengde oppgitt i Joule.

d) Lag en funksjon `kalori_til_kWh(energi_i_cal)` som oversetter en energimengde oppgitt i kalorier til en energimengde oppgitt i kWh. (Tips: Bruk `joule_til_kWh`- og `kalori_til_joule`-funksjonene du allerede har laget).

e) Når vi dusjer bruker vi fort 60 L vann, og dette vannet blir gjerne varmet opp 35 grader. Bruk disse tallene for å estimere hvor mye energi (i kWh) du bruker hver gang du dusjer.

f) En tommelfingerregel er at hver kWh med energi vi kjøper koster ca 1 krone. Hvor mye penger bruker du da på å dusje hvis du dusjer hver dag i et år?

Løsning oppgave 33 *Hvor mye koster morgendusjen din?*

a)

```
1 def kWh_til_joule(energi_i_kWh):
2     return energi_i_kWh*1_000_000*3.6
```

b)

```
1 def joule_til_kWh(energi_i_joule):
2     return energi_i_joule/(1_000_000*3.6)
```

c)

```
1 def kalori_til_joule(energi_i_cal):
2     return energi_i_cal*4.2*1000
```

d)

```
1 def kalori_til_kWh(energi_i_cal):
2     energi_i_joule = kalori_til_joule(energi_i_cal
          )
3     return joule_til_kWh(energi_i_joule)
```

e) Vi må bruke ca 2.5 kWh energi for å varme opp vannet til en dusj.

f) Et år med dusjing daglig koster ca 900 kroner.

5 Plotting

5.1 Matematikk

Oppgave 34 *Innhegning*

Du har 100 meter gjerde og vil lage en rektangulær innhegning. Vi skal se på hvor stort areal denne innhegningen vil ha.

a) Tegn opp et rektangel på papir. Hvis du sier at den ene siden er x meter lang, hvor lang blir da de tre andre sidene i rektangelet? Skriv det opp på arket.

b) Skriv ut uttrykket for arealet til hele innhegningen.

c) Hvor stor kan x maksimalt være?

Du skal nå lage et Python-program som plotter arealet av innhegningen som en funksjon av x.

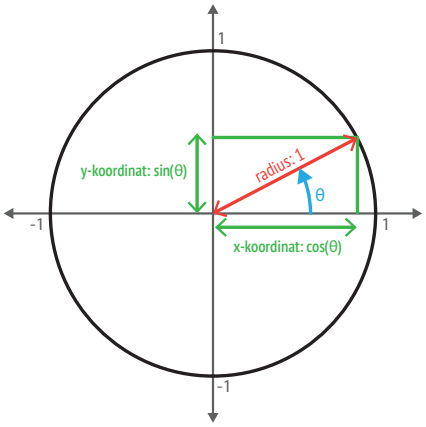
d) Fyll inn i skjelettkoden under for å lage programmet:

```
from pylab import *

x = arange(0, ..., 0.1)
A = ...

plot(x, A)
xlabel('x')
ylabel('Areal')
show()
```

e) Kjør programmet og se på figuren som tegnes. Hvilket valg av x er det som gir størst mulig areal? Hva slags innhegning er det vi ender opp med å lage?

Oppgave 35 *Sirkelparameterisering*

I denne oppgaven skal vi bruke Python til å tegne en sirkel med radius lik 1 som er sentrert i origo. En slik sirkel er gitt ved parametriseringen $x(\theta) = cos(\theta)$ og $y(\theta) = sin(\theta)$. Hvis du lurer på hvorfor sirkelen har denne parametriseringen er lurt å legge merke til at det vi ønsker å tegne rett og slett er en enhetssirkel. Og hvis man har vinkelen θ så er x- og y-koordinaten til punktet med den vinkelen gitt ved $x = cos(\theta)$ og $y = sin(\theta)$



a) Bruk `linspace` til å opprette en *array*, `theta`, som skal inneholde 100 tall fra 0 til 2π

b) Opprett en variabel `x` som inneholder x-koordinatene til punktene på sirkelen (x koordinatene er gitt ved cos(θ))

c) Opprett en variabel `y` som inneholder de korresponderende y-koordinatene til punktene på sirkelen (altså sin(θ)).

d) Tegn sirkelen med `plot`-kommandoen (Hint: husk `show`).

e) Se på sirkelen, er det noe som ikke stemmer?

f) For å få sirkelen til å være sirkulær, ikke elliptisk, så må en centimeter på x-aksen tilsvare en centimeter på y-aksen. For å få til dette kan vi skrive `axis("equal")`. Det tvinger aksene til å ha samme skalering. Legg til denne kommandoen i programmet ditt. Kjør koden og se på plottet, har vi en sirkel nå?

Løsning oppgave 35 *Sirkelparameterisering*

a)

```
from pylab import *
theta = linspace(0, 2*pi)
```

b)

```
x = cos(theta)
```

c)

```
y = sin(theta)
```

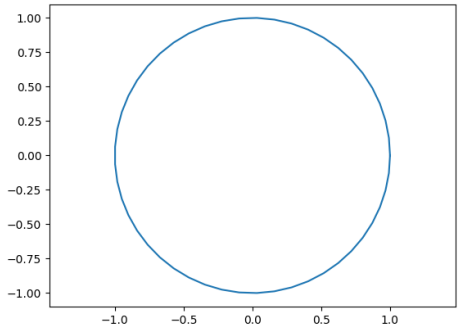
d)

```
from matplotlib.pyplot import *
plot(x, y)
show()
```

e)

```
plot(x, y)
axis("equal")
show()
```

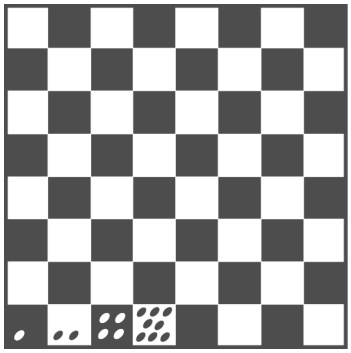
Resultatplott:



Oppgave 36 *Sjakk og riskornvekst*

Det finnes en nesten tusen år gammel legende om oppfinnelsen av sjakk som går slik: En veldig smart mann finner opp spillet sjakk og viser det til keiseren sin. Keiseren blir så imponert at han erklærer at oppfinneren kan velge sin egen belønning. Oppfinneren svarer at han er en ydmyk mann og ønsker kun ris. Og siden det er sjakk han blir belønnet for vil han ha et riskorn for den første ruta i brettet, to for den andre, fire for den tredje og så videre. Han ønsker altså at mengden ris skal dobles for hver rute i brettet.

Kongen synes dette er en beskjeden belønning og aksepterer den på stedet. Men når han forteller det til sin kasserer får han beskjed om at hele keiserdømmet vil gå konkurs!



a) Bruk en for-løkke til å finne ut hvor mange riskorn oppfinneren ba om. Er du enig med kassererens fortvilelse?

b) Det er omtrent 50 000 riskorn i et kilo med ris. Bruk en `while`-løkke til å finne ut hvor mange ruter man må belønne oppfinneren for for at han skal få minst et kilo med ris.

c) Lag et plot over veksten av riskorn. La x-aksen være antall sjakkruter og y-aksen være antall riskorn. Gjør plottet pent ved å gi navn til aksene med `xlabel(...)`/`ylabel(...)`), legg på en tittel med `title(...)` og et rutenett med `grid()`.

Løsning oppgave 36 *Sjakk og riskornvekst*

a)

```
antall_riskorn = 0

for i in range(64):
    antall_riskorn += 2**i

print(f"Antall riskorn: {antall_riskorn}")
```

b)

```
riskorn1kg = 50000
antall_riskorn = 0
antall_ruter = 0

while antall_riskorn < riskorn1kg:
    antall_riskorn += 2**antall_ruter
    antall_ruter += 1

print(f"Antall ruter: {antall_ruter}")
```

c)

```
from pylab import *

xer = range(1, 65)
yer = []
antall_riskorn = 0

for i in range(64):
    antall_riskorn += 2**i
```

```
    yer.append(antall_riskorn)

plot(xer, yer)
xlabel("Ruter")
ylabel("Antall riskorn")
legend(["Riskorn vs ruter"])
title("Legende")
grid()
show()
```

5.2 Naturfag

Oppgave 37 *Vekstfaktor*

I en petriskål lever en bakteriekultur som består av 1000 bakterier som formerer seg raskt nok til at mengden bakterier øker med 50% hver time. I denne opppgaven skal vi bruke plotting og python til å modellere og visualiserer veksten i bakteriekulturen over 10 timer

a) Opprett en variabel, `vekstfaktor`, som har vekstfaktoren tilhørende 50%

b) Bruk `arange` fra `pylab` til å opprette en array, `timer`, med tallene fra 1 til 10

Nå vil vi regne ut hvor mange bakterier som er i petri-skålen hver time. For å gjøre det, må vi først opprette en tom array og bruke en løkke til å "fylle" inn antall bakterie hver time. Men først, hva er egentlig en tom array? Vel, vi kan jo bruke en array hvor alle elementene er lik 0. Dette er det en funksjon for i `pylab` fra før av, og den heter `zeros`. Hvis vi skriver `ti_nuller =zeros(10)`, vil vi få en variabel `ti_nuller` som består av ti element som alle er lik null.

c) Bruk `zeros` fra `pylab` til å opprette en tom array, `bakteriemengde` med 10 elementer. Husk å importere `zeros` først!

Det neste steget er å endre verdiene til hvert element i arrayen vårt. Dette kan vi gjøre med *indeksering*. En array består av mange tall som er etter hverandre, og hvis du vil ha ut et tall fra en array bruker vi klammeparanteser. Hvis vi har en array-variabel, `x`, kan rett og slett skrive `x[0]` for å hente ut det første elementet i arrayen. Tilsvarende kan vi skrive `x[1]` for å hente ut det andre elementet i arrayen, osv. Under har vi et eksempel

```
from pylab import arange

tallrekke = arange(10)*2
første_tall = tallrekke[0]
femte_tall = tallrekke[4]

print(første_tall)
print(femte_tall)
```

```
0
8
```

Tilsvarende, kan vi starte med en tallrekke, og endre enkeltelement! Se eksempelet under.

```
from pylab import arange

tallrekke = arange(4)*2
print(tallrekke)

tallrekke[0] = -1
tallrekke[2] = 0

print(tallrekke)
```

```
[0 2 4 6]
[-1  2  0  6]
```

Vi ser her at vi bruker `tallrekke[i] =x` for å sette verdien til element nummer $i+1$ i arrayet lik x.

d) Bruk array-indeksering (klammeparanteser) til å sette det første elemen-

tet (element nummer 0) i `bakteriemengde` til `1000` som er startverdien.

e) Bruk en løkke `for time in range(1,10)` til å løkke igjennom alle timene fra 1 til 9

f) Bruk arrayindeksering til å sette bakteriemengden for tid `time` til å være lik vekstfart multiplisert med bakteriemengden for tid `time-1`. **Hint:** `bakteriemengde[time-1]`

g) Bruk `plot` og `show` funksjonene fra `pylab` til å plotte bakteriemengde på yaksen og timer på y-aksen

h) Bruk `xlabel` og `ylabel` funksjonene fra `pylab` til å legge merkelapper på x og y-aksen

i) Refleksjonsoppgave: Hva vil skje med bakteriene etterhvert som tiden går? Hva synes du om denne modellen? Er det noen svakheter ved en slik modell?

Løsning oppgave 37 *Vekstfaktor*

a)

```
1 vekstfaktor = 1.5
```

b)

```
1 from pylab import arange
2
3 timer = arange(10)
```

c)

```
1 from pylab import arange, zeros
2
3 bakteriemengder = zeros(10)
```

d)

```
1 bakteriemengder[0] = 1000
```

e)

```
for time in range(1, 10):
    bakteriemengder[time] = bakteriemengder[time-1
       ]*vekstfaktor
```

f)

```
from pylab import plot, show

plot(timer, bakteriemengder)
show()
```

g)

```
from pylab import plot, show, xlabel, ylabel

plot(timer, bakteriemengder)
xlabel("Timer")
ylabel("Antall bakterier")
show()
```