

Oppgaver

Her finner du oppgaver som hører til Kodeskolens introduksjonskurs i programmering, og dette dekker begge dagene. Tema for første dag er *introduksjon til Python, variabler, input, betingelser* og *løkker*. Tema for andre dag er *funksjoner* og *plotting*.

Dersom du står fast er det bare å spørre. I tillegg anbefaler vi å lese i kompendiet hvis det er noen temaer du synes er spesielt vanskelige. Oppgaver markert som bonusoppgaver er litt mer utfordrende og du velger selv om du har lyst til å prøve deg på dem.

God koding!

Innhold

1	Variabler og regning	2
2	Betingelser	5
3	Løkker	7
4	Funksjoner	11
5	Plotting	14

1 Variabler og regning

Oppgave 1 *Printing*

- a) Lag et program som skriver ut teksten «Hei, verden!», til skjermen.
- b) Lag et program der du først lagrer navnet ditt i en variabel, og så få programmet ditt til å skrive ut en hilsen direkte til deg.
- c) Hvis du bruker kommandoen `len()` på variabelen din får du ut antall bokstaver i navnet ditt. Endre programmet ditt så det også skriver ut denne informasjonen.
- d) Du kan bruke funksjonen `input()` til å stille brukeren et spørsmål. Bruk dette til å lage et program som spør brukeren om navnet deres, og deretter skriver ut en beskjed som bruker navnet de har oppgitt. Skriv også ut hvor mange bokstaver det er i navnet til vedkommende.

Oppgave 2 *Kvadrattall*

Kvadrattall er heltall som er blitt kvadrert, altså ganget med seg selv, eller opphøyet i annen. I Python kan du regne ut kvadratet av et tall n enten ved å skrive `n*n` eller `n**2`.

- a) Skriv et program som spør brukeren om et tall, og deretter skriver ut kvadratet av tallet brukeren ga.

Pass på at når vi spør om et tall må vi gjøre om svaret til fra brukeren til en tallvariabel ved å skrive `int` på følgende måte: `int(input())`. Dette er fordi «int» står for «integer», som er engelsk for heltall.
- b) Bruk programmet ditt og prøve-feile metoden til å finne det minste tallet som har et kvadrat på over 1000.

Oppgave 3 *Finn fire feil!*

Her følger det fire programmer som har blitt skrevet feil. Finn feilen i hver

programsnutt. Du kan godt kjøre programmet inn på din egen maskin, og kjøre det, da kan kanskje feilmeldingen hjelpe deg å skjønne hva som er galt.

Når du tror du skjønner hva som er galt, rett feilen på din egen maskin, og kjør programmet for å sjekke at det fungerer som det skal.

a)

```
1 print(Hei , Verden!)
```

b)

```
1 Print("Hei, Verden!")
```

c)

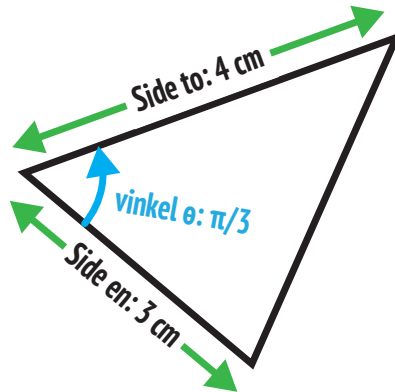
```
1 person = input("Hva heter du?")
2 print("Hei på deg, {navn}")
```

d)

```
1 pi = 3,14
2 radius = 4
3 areal = radius*pi**2
```

Oppgave 4 Arealsetningen

Vi kjenner til to sider og vinkelen mellom dem i en trekant:



Arealsetningen sier at arealet er gitt ved:

$$\frac{1}{2} \times \text{side1} \times \text{side2} \times \sin \theta$$

- Vi ser at vi kommer til å trenge `sin`-funksjonen. Importer `sin` og `pi` fra `math` biblioteket.
- Lag variablene `side1`, `side2` og `vinkel` og gi dem verdiene 3, 4 og $\frac{\pi}{3}$ (**OBS:** vinkelen er gitt i radianer)
- Regn ut arealet til trekanten og skriv resultatet ut til terminalen
- Modifiser programmet ditt til å først spørre brukeren om verdiene til `side1`, `side2` og `vinkel` med `float` og `input` og så skrive ut arealet til terminalen.

2 Betingelser

Oppgave 5 *Finn størst tall*

I denne oppgaven skal vi øve på `if`-tester ved å finne det største av to tall.

- a) Skriv et program som ber brukeren om å skrive inn to tall og lagre dem i to variabler, `tall1` og `tall2`. Gjør om tallene til flyttall ved hjelp av `float`.
- b) Utvid programmet ditt ved hjelp av `if` slik at dersom `tall2` er større enn `tall2`, skriver du ut en beskjed om at `tall1` er størst.
- c) Utvid programmet ditt videre ved hjelp av `elif`, slik at dersom `tall2` er større enn `tall1`, får brukeren beskjed om at `tall2` er størst.
- d) Fullfør til slutt koden ved å legge til en `else`-blokk, slik at dersom tallene er like, vil brukeren få beskjed om det.

Oppgave 6 *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:

$$|a| = \begin{cases} a, & \text{hvis } a \geq 0 \\ -a, & \text{hvis } < 0 \end{cases} \quad (1)$$

- a) Lag et program 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

Oppgave 7 *Vinkeltyper*

Vi kan dele vinkler inn i tre forskjellige typer:

1. En vinkel som er mindre enn 90° kalles en *spiss* vinkel.
2. En vinkel som er større enn 90° kalles en *stump* eller *butt* vinkel.
3. En vinkel som er akkurat 90° kalles en *rett* vinkel

Be brukeren om å gi en vinkel og bruk **if**, **elif** og **else** til å fortelle brukeren om vinkelen er *spiss*, *stump* eller *rett*

Oppgave 8 *ABC-formelen*

ABC-formelen for å løse annengradsformler er som følger:

La a , b og c være reelle tall, hvor $a \neq 0$. Da har likningen $ax^2 + bx + c = 0$ løsningene

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

- a) Lag et program som finner løsningene til en annengradslikning med $a = 1$, $b = 2.5$ og $c = 1$.
- b) Modifiser programmet ditt så det spør brukeren om verdier for a , b og c og skriver ut de tilhørende løsningene.
- c) Dersom $b^2 - 4ac < 0$ har den tilhørende annengradslikningen ingen reell løsning. Bruk en **if**-betingelse til å informere brukeren om at det ikke finnes noen reell løsning dersom $b^2 - 4ac < 0$
- d) Dersom $b^2 - 4ac = 0$ har likningen kun en løsning. Bruk en **elif** til å sjekke om dette er tilfelle og i så tilfelle informere brukeren om at det kun er en løsning

3 Løkker

Oppgave 9 *For-løkker*

Bruk en **for**-løkke til å gjennomføre disse oppgavene.

- a) Print meldingen 'Hei, verden' 5 ganger.
- b) Print alle tallene fra 1 til 101.
- c) Modifiser programmet slik at det også printer ut alle kvadrattallene (x^2) mellom 1 og 10000 (100^2). Husk at du kan opphøye tallet `tall` i andre med kommandoen `tall**2`.

Oppgave 10 *Finne kvadrattall og kubikktall*

Lag et program som regner ut kvadrattallet og kubikktallet av tallene 1-5, og printer ut på en linje tallet og tilhørende kvadrat og kubikk. Bruk en **for**-løkke og legg til mellomrom mellom tallene.

Oppgave 11 *Fakultet*

Si at du har 5 forskjellige små skulpturer du skal sette opp på rekke. Hvor mange ulike måter kan du gjøre dette på? For den første har du 5 valg, deretter har du 4 valg, så 3, og så videre. Dermed blir antall kombinasjoner til

$$5 \cdot 4 \cdot 3 \cdot 2 \cdot 1 = 120.$$

I matematikken skriver vi dette mer kompakt som $5!$, som vi kaller *fakultet*. Å rangere n ting kan altså gjøres på $n!$ måter, som blir

$$n! = n \cdot (n - 1) \cdot \dots \cdot 2 \cdot 1.$$

For små tall er det enkelt å regne ut fakultet for hånd, men dersom n begynner å bli litt større vil dette ta fryktelig lang tid.

- a) Du skal nå skrive et program som kan regne ut fakultet for oss. Du

kan enten prøve å gjøre dette helt selv, eller følge vår «steg-for-steg»-instrukser under.

- Spør brukeren om hva n er, lagre svaret i en variabel. Husk å konvertere svaret med `int(input())`.
- Lag en variabel `fakultet`, som du gir verdien 1.
- Lag en for-løkke som går fra 1 til n .
- For hver iterasjon av løkka, gang `fakultet`-variabelen din med tallene du løkker over.
- Skriv ut det endelige svaret så brukeren kan se det.

- b) Test at programmet ditt gir $5! = 120$.
- c) En vanlig kortstokk har 52 unike kort. Hvor mange mulige rekkefølger kan vi få om vi stokker en kortstokk?
- d) I noen kortspill legger man til 2 jokere i kortstokken, slik at det nå er 54 kort i kortstokken. Hvor mange måter blir det nå å stokke kortstokken på?

Oppgave 12 *Trekanttall*

Et trekanttall er summen av tallrekken

$$1, 2, \dots, n.$$

For eksempel så er

$$1 + 2 + 3 + 4 + 5 = 15,$$

så da er 15 et trekanttall. Siden det var summen av de 5 første tallene i tallrekka, så sier vi at det er det *femte* trekanttallet.

La oss si at vi vil vite hva det hundrede trekanttallet er, da må vi legge sammen alle tallene fra 1 til 100. Det blir fort slitsomt og kjedelig å gjøre for hånd. Så la oss bruke programmering.

For å gjøre det lettere å sjekke om programmet vårt, så startet vi med å prøve å regne ut summen fra 1 til 5.

- a) Lag en løkke som skriver ut tallene

1, 2, 3, 4 og 5

til skjermen.

- b) Endre nå løkka så du isteden finner summen av tallene 1 til 5. Da må du først lage en variabel utenfor løkka, og for hvert tall, legge det til variabelen din. Husk at du kan legge noe til en varibel med `+=`.
- c) Sammenlign svaret programmet ditt gir med det vi fant for hånd. Er de to like? Hvis de ikke er det så er det noe galt!
- d) Hvis programmet ditt fungerte som forventet kan du nå endre sånn at du regner summen av de første 100 tallene

$$1 + 2 + 3 + \dots + 100.$$

Det er en kjent matematiker, Carl Friedrich Gauss, som fikk denne oppgaven av sin mattelærer når han gikk på skolen på 1700-tallet. Læreren tenkte nok at dette skulle holde Gauss opptatt en god stund med å legge sammen tall etter tall. Gauss hadde ikke tilgang til en datamaskin, så han kunne ikke automatisere jobben slik vi har gjort, men ha la merke til et mønster i tallene. Gauss la merke til at om vi starter på begge endene av rekka får vi et mønster

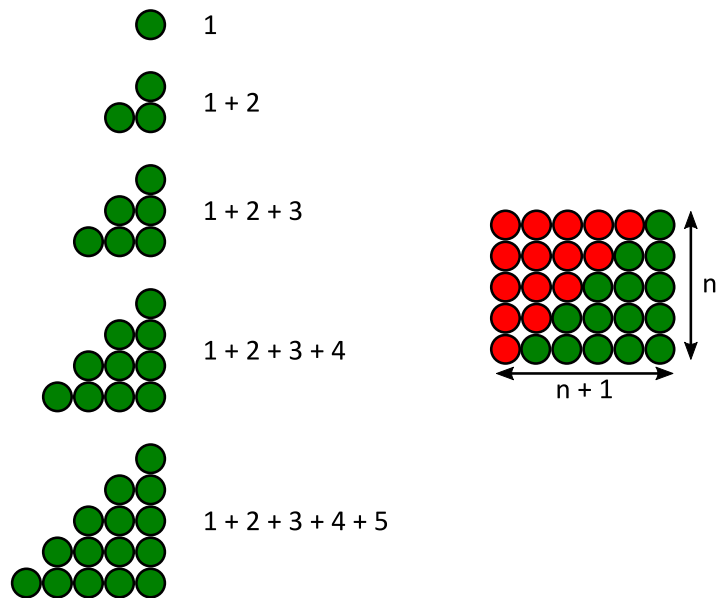
$$1 + 100 = 101, 2 + 99 = 101, 3 + 98 = 101, \dots 50 + 51 = 101.$$

Fra dette mønsteret klarte Gauss å finne en formel for summen av tallene fra 1 til n , og uttrykket hans var

$$T_n = \frac{n(n+1)}{2}.$$

-
- e) Bruk formelen til Gauss og sjekk at du får samme svar som programmet ditt for $n = 100$.
- f) Gjør det samme for $n = 1000$, så $n = 1000000$ (én million).
-

For å skjønne hvorfor denne formelen er som den er, så kan det lønne seg å skjønne hvorfor de kalles trekantall. Om vi tegner opp summene som antall baller, og tenger først 1, så 2, og så 3 bortover, sånn som dette:



Så ser vi at de ulike summene blir trekanter. Vi kan så gjøre om en slik trekant til en firkant ved å legge på like mange nye. Sånn som vist på høyre side av figuren. Denne firkanten har n baller i høyden, og $n + 1$ baller bortover. Da er antall baller i hele firkanten $n(n + 1)$. Men vi har jo doblet antall baller for å få en firkant, så om vi bare skal telle de grønne ballene må vi dele på to.

- g) Hvordan kan du sjekke om et tall er et trekantall? Skriv et program som sjekker om et heltall er et trekantall, og hvis ja, skriver ut faktorene i dette. For eksempel bør programmet skrive ut $1 + 2 + 3 + 4 + 5$ hvis det blir gitt input 15.

4 Funksjoner

Oppgave 13 *Enkle funksjoner*

- a) Lag en funksjon `kvadrat(x)` som tar inn et tall x og returnerer kvadratet x^2 .
- b) Gjenta oppgaven over, men med funksjonen `kubikk(x)` som returnerer x^3 .
- c) Lag en funksjon `f(x)`, som returner $x^2 + 3x - 1$

Oppgave 14 *Annengradsfunksjon*

En annengradsfunksjon er definert slik:

$$f(x) = x^2 + 0.3x - 1$$

- a) Definer funksjonen som en Python-funksjon, `f(x)`, som tar inn en x -verdi og returnerer tilhørende y -verdi.
- b) Test funksjonen din ved å kalle på den med følgende x verdier og skriv ut resultatet til terminalen:
 - 1. $x = 0$
 - 2. $x = 1$
 - 3. $x = -0.4$

Oppgave 15 *Midtpunktfunksjon*

Midtpunktet mellom to tall, a og b er gitt ved:

$$\frac{a + b}{2}$$

- a) Lag en funksjon som tar inn to tall, a og b og returnerer midtpunktet mellom dem
- b) Bruk funksjonen til å finne midtpunktet mellom 34 og 86

Oppgave 16 *Kanonkule*

Året er 1384 i en fiktiv borg i Sør-England. Franskmennene har omringet borgen og kanonene er rullet frem. Dette har åpenbart gjort Lorden som eier borgen nervøs, og han spør deg, hans fremste lærdmann om kanonkulene vil nå borgen med den avstanden de har.

Heldigvis har spionene til kongen klart å stjele en av kanonene, og etter mye om og men, har du funnet ut at kulene forlater kanonen med en fart på ca 50 meter i sekundet. Kanonene er låst i en 30 graders vinkel, så posisjonen til en kanonkule kan beskrives med disse funksjonene

$$\begin{aligned}x(t) &= 25\frac{m}{s}t \\ y(t) &= 45\frac{m}{s}t - 5\frac{m}{s^2}t^2,\end{aligned}$$

hvor $x(t)$ er hvor langt kanonkula har bevegde seg bortover (i meter) t sekund etter at den ble skutt og $y(t)$ er høyden til kanonkula (i meter) t sekund etter at den ble skutt.

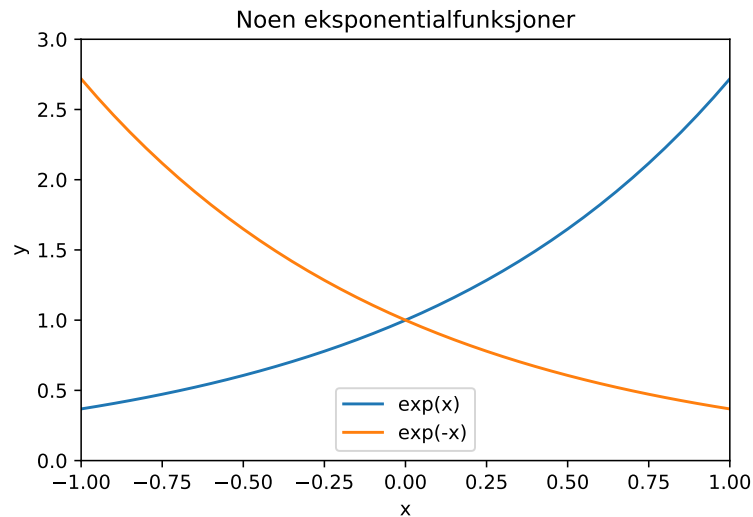
- a) Definer en funksjon `kanonkule_x(t)` som returnerer hvor langt kanonkula har bevegde seg bortover etter t sekund.
- b) Definer en funksjon `kanonkule_y(t)` som returnerer høyden til kanonkula etter t sekund.
- c) Definer en funksjon `kanonkule_posisjon(t)` som returnerer både hvor langt kula har bevegde seg bortover og høyden til den etter t sekund.
- d) Kanonene er plassert 200 meter unna borgen (i x retning), og er i samme høyde som borgen. Bruk funksjonen vi akkurat lagde og prøv ut forskjellige verdier for t . Treffer kanonkula borgen?
- e) (Bonusoppgave) Lag et program som bruker `input` funksjonen til å be om et antall sekunder og printer ut posisjonen til kanonkula så lang tid etter at den ble skutt.

f) (Bonusoppgave) Bruk en løkke til å printe ut posisjonen til kanonkula de ti første sekundene kanonkula er i luften.

5 Plotting

Oppgave 17 *Plotting*

I denne oppgaven skal vi ende opp med et plot som ser slik ut:



- Opprett en array med tallrekken som starter på 0, slutter på 1 og har et mellomrom på 0.05 mellom hvert element. Lagre den arrayen i en variabel du kaller `x`.
- Opprett en array som inneholder e^x for alle verdier i `x` variabelen din (hint: husk `exp` funksjonen i `pylab`) og lagre dette arrayet i en array med navnet `y1`.
- Lag et plot som viser x på førsteaksen og e^x på andreaksen for x -verdier mellom 0 og 1.
- Pynt på plottet ved å sette grenser for x -aksen og y -aksen med `xlim` og `ylim` funksjonene. Sjekk at aksene dine har endret seg siden oppgave b).
- Gi aksene dine merkelapper (f.eks x og y) med `xlabel` og `ylabel` funksjonene.

- f) Gi plottet ditt en tittel (f.eks $y = \exp(x)$) med `title` funksjonen.
- g) Opprett en ny variabel, `y2` og som inneholder e^{-x} for alle verdier i `x` arrayet ditt.
- h) Plot `y1` og `y2` i samme plot.
- i) (Bonusoppgave) Gi kurvene dine merkelapper som forteller hva de representerer og vis frem disse merkelappene med `legend` funksjonen.

Oppgave 18 *Plotte en annengradsfunksjon*

- a) Definer en funksjon for den matematiske funksjonen

$$f(x) = x^2 - 5x + 9$$

- b) Bruk funksjonen `linspace` for å generere `x`-verdier mellom 0 og 5. Lagre disse i en `array`. Finn tilsvarende `y`-verdier ved å sende `x`-verdiene inn som et argument (du kan sende med hele arrayet som ett argument). Skriv ut begge listene.
- c) Plot $f(x)$ mellom $x = 0$ og $x = 5$. Prøv å endre antall `x`-verdier du valgte i (b) og se hvordan det endrer plottet (du kan ta bort `print`-delen her hvis du vil).
- d) Legg til en tittel til plottet og sett navn på aksene.

Oppgave 19 *Grafisk løsning av likning*

I denne oppgaven skal vi løse en likning *grafisk*, dvs. ved å lage og lese av en graf.

- a) Definer en funksjon som returnerer $f(x) = \sin(x^2)$.
- b) Definer en funksjon som returnerer $g(x) = x^2 + x/5 - \exp(-x)$.
- c) Opprett arrays for å lagre `x`-verdier mellom `+` og 1.7, samt verdier man

får ved å kalle på f og g med disse x -verdiene.

- d) Tegn grafen til $f(x)$ for x -verdier mellom 0 og 1.7.
- e) Tegn grafen til $g(x)$ for x -verdier mellom 0 og 1.7 i samme plot som $f(x)$.
- f) For hvilken x er $f(x)$ og $g(x)$ like (ca)?
- g) Marker punktet hvor $f(x)$ og $g(x)$ er like med en rød sirkel.

Oppgave 20 *Fortsettelse kanonkule*

Fortsettelse fra kanonkuleoppgaven. Vi har akkurat fortalt lorden i borgen vår at kanonene kan treffe og han tror ikke på oss. Derfor bestemmer vi oss for å lage en tegning hvor vi viser kulas bane. Om du ikke fikk til forrige kanonkuleoppgave kan du bruke denne funksjonen:

```
1 def kanonkule_posisjon(t):  
2     x = 25*t                # kanonkule_x(t)  
3     y = 45*t - 5*t**2      # kanonkule_y(t)  
4  
5     return x, y
```

- a) Lag en array, t som inneholder en tallrekke som starter på 0, slutter på 9 og har en steglengde på 0.1 (Tips: hvis du vil at tallrekka skal slutte på 9 er det lurt å sette sluttposisjonen til $\text{slutt} + \text{steglengde}$, som i dette tilfellet er 9.1).
- b) Bruk `kanonkule_posisjon(t)` til å regne ut kanonkulas x -koordinater og y -koordinater for hvert tidspunkt i t array-et vårt.
- c) Lag et plot som viser x -posisjon på førsteaksen og y -posisjon på andreaksen.
- d) Ved å løse likningen $x(t) = 200$ ser vi at kanonkula treffer borgen etter åtte sekund. Marker punktet $(x(8), y(8))$ med et rødt punkt.