

▲ Lister og indekser

Skrevet av: Ole Kristian Pedersen, Kodeklubben Trondheim

Kurs: Python

Tema: Tekstbasert

Fag: Programmering

Klassestrinn: 5.-7. klasse, 8.-10. klasse

Introduksjon

Denne oppgaven handler om *lister*, altså å samle flere ting i en og samme variabel. Ettersom lister og løkker henger tett sammen i Python, bør du se på løkker i repetisjonsoppgaven (<https://oppgaver.kidsakoder.no/python/repetisjon/repetisjon>) dersom du har glemt hvordan løkker fungerer.

Hvordan lage lister?

Hver ting i en liste kalles et *element*. En liste lages ved å skrive elementer inni `[]`, med `,` mellom elementene:

```
>>> lst = ['egg', 'ham', 'spam']
>>> lst
['egg', 'ham', 'spam']
```

Vi har nå ei liste som inneholder ordene 'egg', 'ham' og 'spam'. Vanskeligere er det ikke! Vi kan også lage tomme lister:

```
>>> lst = []
>>> lst
[]
```

Ei liste kan inneholde alt mulig - tall, tekster og også andre lister:

```
>>> lst = [ 3, 'komma', [1, 4] ]
>>> lst
[3, 'komma', [1, 4]]
```

Legge til og fjerne elementer

Hva om vi ønsker å legge til eller fjerne elementene fra lista vi vår? Da kan vi bruke de to funksjonene `lst.append(elm)` og `lst.remove(elm)`, der `lst` er lista og `elm` er elementet vi ønsker å legge til eller fjerne.

`lst.append(elm)` legger til `elm` på slutten av `lst`, slik som illustrert i eksempelet:

```
>>> lst = []
>>> lst.append('Per')
>>> lst
['Per']
>>> lst.append('Ada')
>>> lst.append('Kim')
>>> lst
['Per', 'Ada', 'Kim']
```

`lst.remove(elm)` sletter det første elementet `elm` fra `lst`. Det vil si at dersom `elm` ligger flere ganger i `lst` slettes bare det første elementet lik `elm`:

```
>>> lst = ['Per', 'Ada', 'Kim', 'Ada']
>>> lst.remove("Ada")
>>> lst
['Per', 'Kim', 'Ada']
```

Vi skal nå lage et handleliste-program. Programmet skal be brukeren skrive inn matvarer, og deretter skrive ut matvarene når ferdig skrives inn. Programmet skal fungere slik:

```
>>>
Skriv inn en gjenstand: ost
Skriv inn en gjenstand: melk
Skriv inn en gjenstand: brød
Skriv inn en gjenstand: ferdig
ost
melk
brød
```

Dette må du gjøre:

Dette må du gjøre:

- ☐ Lag ei tom liste.
- ☐ Be om input.
- ☐ Så lenge input ikke er lik `ferdig` , legg til det nye elementet i lista.

Hint: Hva slags løkke kan vi bruke her?

- ☐ Skriv ut hvert element i lista.

Hint: Hva slags løkke kan vi bruke her?

Indekser

Tenk deg at vi har ei liste, og ønsker å hente ut det andre elementet i lista. Hvordan skal vi klare det? Da bruker vi noe kalt *indeks*. Indeks er plassen til elementet og skrives mellom `[]` rett etter variabelen: `lst[index]` . Her er et eksempel på en liste med tall:

```
>>> lst = [1, 2, 3, 4, 5]
>>> lst[1]
2
```

Du la kanskje merke til at vi skrev `1` , men fikk ut det andre elementet i lista? Det er fordi vi begynner å telle fra `0` . Dermed har det første elementet i lista indeks `0` , og det andre har indeks `1` . Datamaskiner begynner å telle på null! Du husker kanskje at det samme skjer når vi bruker `range()` ?

```
>>> list(range(5))
[0, 1, 2, 3, 4]
```

Til nå har vi brukt `for element in lst` for å gå igjennom elementene i lista, men noen ganger kan det i tillegg være praktisk å telle hvor langt vi er kommet i lista. Til dette kan vi bruke `enumerate()` , som gir oss både verdien og indeksen:

```
>>> lst = ['Per', 'Kim', 'Ada']
>>> for i, value in enumerate(lst):
    print(i, value)
```

```
0 Per
1 Kim
2 Ada
```

I eksempelet over får `i` verdien av indeksen, og `value` får verdien av elementet. Det er nesten som ei vanlig `for`-løkke, men vi får indeksen i tillegg.

Modifiser nå programmet fra forrige oppgave til å skrive ut indekser ved siden gjenstandene i handlelista. Slik skal det fungere:

```
>>>
Skriv inn en gjenstand: ost
Skriv inn en gjenstand: melk
Skriv inn en gjenstand: brød
Skriv inn en gjenstand: ferdig
0 ost
1 melk
2 brød
```

Dette må du gjøre:

- ☐ Bruk programmet fra oppgaven over.
- ☐ Bruk `enumerate` til for å få indeksen til hver element.
- ☐ Skriv ut indeksen på samme linje som elementet i lista.

Indekstrening

Vi vil nå la brukeren selv velge hvor mange gjenstander som skal skrives ut. Slik som i eksempelet:

```
>>>
```

```
Skriv inn en gjenstand: ost
```

```
Skriv inn en gjenstand: melk
```

```
Skriv inn en gjenstand: brød
```

```
Skriv inn en gjenstand: ferdig
```

```
Hvor mange gjenstander vil du skrive ut? 2
```

```
0 ost
```

```
1 melk
```

Dette må du gjøre:

- ☐ Begynn med programmet du allerede har.
- ☐ Før gjenstandene skrives ut, spør om hvor mye som skal skrives ut.
- ☐ Avbryt utskriften når antallet er lik det brukeren ba om.

Tekster og indekser

Vi kan også bruke indekser på tekster. For eksempel:

```
>>> s = "Ada"
```

```
>>> s[0]
```

```
'A'
```

Vi skal nå skrive et program som henter input fra brukeren og skriver ut annenhver bokstav. Det skal fungere som dette:

```
>>>
```

```
Skriv inn en setning: Hei på deg!
```

```
H
```

```
i
```

```
p
```

```
e
```

```
!
```

Dette må du gjøre:

- ☐ Hent input fra brukeren.
- ☐ Bruk en løkke for å hente ut hver bokstav og dens indeks.
- ☐ Hvis indeksen er et partall, skriv ut bokstaven.

Hint: `tall%2` er *resten* av `tall` delt på 2, hva gir `tall%2` når `tall` er et partall?

Lisens: CC BY-SA 4.0 (<http://creativecommons.org/licenses/by-sa/4.0/deed>)