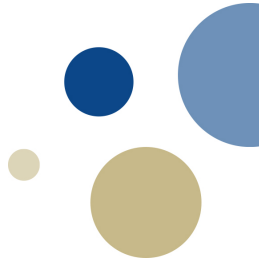
Iterables

Iterables er generelt grupper med objekter man kan *iterere* seg gjennom.

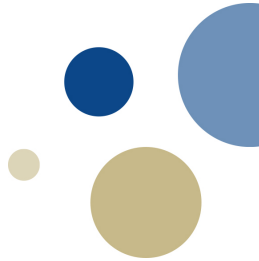
Lister



Lister er en gruppe med objekter som ligger etter hverandre i *minnet*.
De defineres med firkant-parenteser rundt objekter som er separert med komma.
De kan være av forskjellige typer, men det er ikke alltid lurt. Da kan man ikke behandle elementene likt.

```
liste = ["hei", 1, 0.1]
```

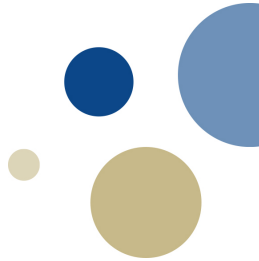
Lister



Lister kan endres.

```
liste = ["hei", 1, 0.1]
liste[1] = "alle"
```

Lister



Lister kan utvides.

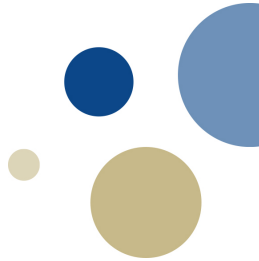
```
liste = ["hei", 1, 0.1]
liste = liste + ["alle", "sammen"]
```

Lister

Alle gjør endringer på den originale listen, **bortsett fra** *copy()*, som returnerer en ny liste. Dette er viktig dersom man vil beholde originalen.

Metode	Eksempel		Metode	Eksempel
append	liste.append("hei")		reverse	liste.reverse()
insert	liste.insert(2, "hei")		sort	liste.sort()
remove	liste.remove("hei")		copy	kopi = liste.copy()
pop	liste.pop(1)		clear	liste.clear()
count	liste.count("hei")			

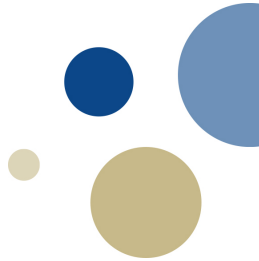
Lister



Når man bruker lister i funksjoner, så må man huske at hvis man endrer listen man tar inn, så endres den utenfor også!
Dette gjelder ikke for vanlige variabler/parametere.

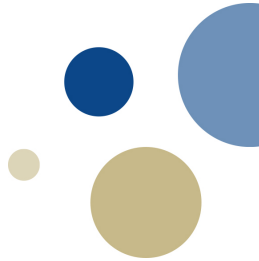
```
def sett_tall(tall):
    tall = 1
def legg_til(liste):
    liste.append(1)
a = "hei"
sett_tall(a) # a endres ikke
l = [1, 5.12]
legg_til(l) # l endres
```

Tupler



Tupler er basically det samme som lister, men de er ikke-muterbare, som strenger.

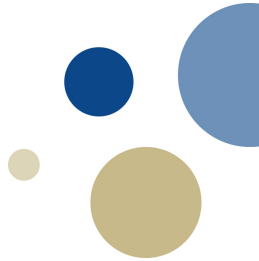
Tupler



Defineres på samme måte som lister, bare med parenteser istedenfor firkantparenteser.

```
liste = [1, 2, 3]
tuple = (1, 2, 3)
```

Tupler



Metoder: Siden tupler er ikke-muterbare, er det svært få metoder, og de er generelt ikke nødvendige å kunne.

- count
- index

```
t = (1, 2, 3, 1)
t.count(1) # -> 2
t.index(1) # -> 0
```

Tupler

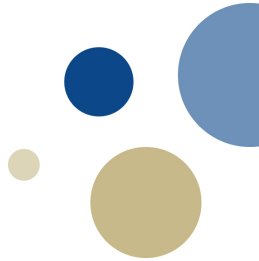
Hvorfor bruke tupler?

Tupler

Hvorfor bruke tupler?
Tupler kan brukes der du vet du har en mengde som er konstant

```
mynter = (20, 10, 5, 1)
penger = 19
liste  = [0] * len(mynter)
for i in range(len(mynter)):
    liste[i] = penger \\ mynter[i]
    penger  %= mynter[i]
```

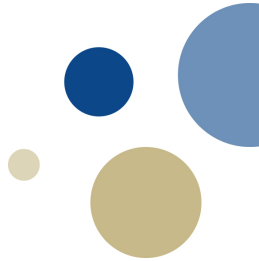
Tupler



Kan forenkle kode som trenger flere linjer. Tenk tilbake til tilordningsproblemet der vi ville bytte om på verdiene i to variabler.

```
a = "hei"
b = 2
temp = a
a = b
b = temp
```

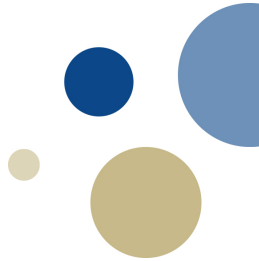
Tupler



Kan forenkle kode som trenger flere linjer. Tenk tilbake til tilordningsproblemet der vi ville bytte om på verdiene i to variabler. Dette kan nå forenkles.

```
a = "hei"
b = 2
a, b = b, a
```

Tupler



Mange nyttige funksjoner man kan få bruk for bruker også tupler.

```
for index, element in enumerate(liste)
for key, item in dictionary.items()
```

Tupler

Generelt: hvis du kan gjøre det med tupler, kan det gjøres med lister eller med annen kode. Tupler er hovedsakelig *ekstra*.