IN1000

UKE 6 - GRUPPE 1

MIAU@IFI.UIO.NO

For vs. while

For-løkke

Når du går gjennom noe, som en liste, en fil eller en range

While-løkke

Når du vil gjenta noe så lenge en betingelse er sann

For vs. while

For-løkke

Når du går gjennom noe, som en liste, en fil eller en range

```python
for i in range(3):
    print(i)
```

While-løkke

Når du vil gjenta noe så lenge en betingelse er sann

```python
i = 0
while i < 3:
    print(i)
    i += 1
```

Nøstede løkker

- Når vi har en løkke inni en annen løkke
- Den indre løkken kjører helt ferdig for hver runde av den ytre

```
for i in range (1, 4):
    for j in range(i):
        print(i, j)
```

Hva skrives ut her?

Oppgave

Hva skrives ut her?

```
for i in range(3):
    linje = ""
    for j in range(3):
        if i == j:
            linje += "a"
        else:
            linje += "b"
    print(linje)
```

Funksjoner

Parameter er variabelen(e) i definisjonen



Funksjoner

Argumenter er verdiene du setter inn i kallet på funksjonen



Hva skjer når en funksjon kalles?

1. Argumentene legges inn i parameterene

```
def sum(a, b):
    c = a + b
    return c


tall = sum(10, 5)
```

Hva skjer når en funksjon kalles?

1. Argumentene legges inn i parameterene
2. Programmet hopper inn i funksjonen og kjører koden

```
def sum(a, b):
    c = a + b
    return c


tall = sum(10, 5)
```

Hva skjer når en funksjon kalles?

1. Argumentene legges inn i parameterene
2. Programmet hopper inn i funksjonen og kjører koden
3. Ved return (eller når funksjonen er ferdig) hopper vi tilbake til der vi sto

```
def sum(a, b):
    c = a + b
    return c


tall = sum(10, 5)
```

Hva skjer når en funksjon kalles?

1. Argumentene legges inn i parameterene
2. Programmet hopper inn i funksjonen og kjører koden
3. Ved return (eller når funksjonen er ferdig) hopper vi tilbake til der vi sto
4. Kallet ‘erstattes’ med returverdien

```
def sum(a, b):
    c = a + b
    return c


tall = sum(10, 5)
```

print vs. return

print viser noe på skjermen, men programmet bruker ikke verdien videre

return sender verdien tilbake, slik at den kan lagres og brukes videre i programmet ditt

Oppgave

Hva er feil med denne koden, og hvorfor er det feil?

```
def inneholder(liste, verdi):
    for element in liste:
        if element == verdi:
            return True
        else:
            return False
```

Oppgave

Burde heller sett slik ut:

```
def inneholder(liste, verdi):
    for element in liste:
        if element == verdi:
            return True
    return False
```

Tips for å løse en oppgave

- Hva tar funksjonen min inn, og hva skal den returnere?
- Løs problemet for ett element eller en linje først, så utvid logikken til resten (med f.eks. en løkke)
- Kjør koden din regelmessig så du finner ut av eventuelle feil tidlig
- Om du er usikker på hvordan ulike samlinger eller løkker fungerer, test de med print-setninger

Lesing av linje, steg for steg

```
linje = "Ola 12 15 9\n"
linje.strip()                    # "Ola 12 15 9"
deler = linje.strip().split()    # ["Ola", "12", "15", "9"]
deler[0]                         # "Ola"
deler[1]                         # "12"
deler[2]                         # "15"
deler[3]                         # "9"
```

Oppgave

Du har denne filen:

```
1    Ola 12 15 9
2    Kari 18 14 16
3    Selma 10 20 13
```

1. Lag les_fil(filnavn) som returnerer en ordbok med navn som nøkkel og en liste med poeng (int) som verdi
2. Lag snitt(liste) som returnerer gjennomsnittet poeng hver person har fått
3. Bruk funksjonene til å skrive ut gjennomsnittet til hver person



Oppgave 1

Skriv en funksjon, *tell_grader(fag, bsc, msc)*, som tar inn tre argumenter av datatypen string og sjekker om en bachelorgrad (bsc) og/eller en mastergrad (msc) er innen et bestemt fag.

Dersom fag har samme verdi som bsc og msc, skal funksjonen returnere 2. Dersom fag har samme verdi som enten bsc eller msc, skal funksjonen returnere 1. Dersom fag hverken er lik bsc eller msc, skal funksjonen returnere 0.

For eksempel skal kallet tell_grader("informatikk", "informatikk", "informatikk") returnere 2, mens kallet tell_grader("historie", "informatikk", "informatikk") skal returnere 0. Test funksjonen med flere eksempler på fag, bsc og msc i et hovedprogram som er definert som hovedprogram().

Oppgave 2

Skriv en funksjon, forkort_lagliste(lagliste), som tar som argument en liste av strenger som er lagnavn og returnerer en ny liste hvor ingen lagnavn finnes flere ganger i lista.

Med andre ord, dersom samme streng opptrer flere ganger i lista som funksjonen mottar som argument, skal denne bare opptre én gang i lista som blir returnert.

For eksempel skal kallet forkort_lagliste(["Brann", "Molde", "Brann"]) returnere en liste ["Brann", "Molde"]. Test funksjonen med flere eksempler på lagliste i et hovedprogram som er definert som hovedprogram().

Oppgave 3

Skriv en funksjon, fjern_vokaler(setning, vokalliste), som tar inn en setning (datatype string) og en liste med vokaler (hvor hvert element i lista er en string av lengde 1).

Funksjonen skal returnere en kopi av setning, hvor alle vokaler fra vokalliste er fjernet. Du kan anta at både setning og vokalliste kun inneholder små bokstaver.

For eksempel skal kallet fjern_vokaler("ha det fint", ["a", "e", "i", "o", "u"]) returnere teksten "h dt fnt". Test funksjonen med flere eksempler på setning og vokalliste i et hovedprogram som er definert som hovedprogram().

Oppgave 4

Skriv en funksjon, er_sortert(talliste), som tar en liste med tall som argument og som returnerer en boolsk verdi.

Funksjonen skal sjekke om alle tallene i lista er i stigende rekkefølge (sorterte). Dersom alle verdiene er i stigende rekkefølge, skal funksjonen returnere True, ellers skal funksjonen returnere False. Anta at alle tallene i lista er ulike. Funksjonen trenger altså ikke ta hensyn til eventuelle like verdier.

For eksempel skal er_sortert([2, 4, 6, 8, 10]) returnere True og er_sortert([7, 3, 1, 9]) returnere False. Test funksjonen med flere tallister i et hovedprogram som er definert som hovedprogram().

Oppgave 5

Denne oppgaven bygger videre på oppgave 5 ovenfor. Lag et hovedprogram som legger inn alle sorterte lister i én ordbok og alle usorterte lister i en annen ordbok. Nøklene i ordboka for de sorterte listene er på formen 1s, 2s, 3s, o.s.v. I ordboka for de usorterte listene er nøklene på formen 1u, 2u, 3u, o.s.v.

Når de to ordbøkene er laget, skal dere bruke ordbøkene til å lage en liste med lengde lik 2. Det første elementet i lista er en liste med sorterte lister, og det andre elementet i lista er en liste med usorterte lister.

Til slutt skal programmet skrive ut en oversikt over de sorterte listene og en oversikt over de usorterte listene. Husk passende overskrifter i utskriften, og sørg for at hver av listene skrives ut på nye linjer. Dere skal ikke skrive ut elementene i de sorterte og usorterte listene - bare de sorterte og usorterte listene i seg selv.