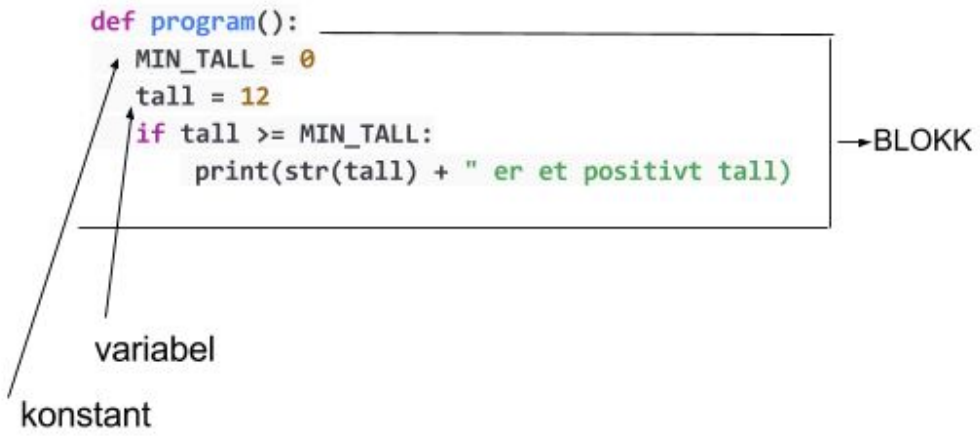
Variabler, konstanter, blokker og skop

Repetisjonskurs IN1000 H18

Kristine Jevne Berge (kristijb)

Introduksjon



Variabler

int - Heltall	heltall = 2
float - Flyttall	flyttall = 2.0
boolean - sannhentsverdi	sant = True
String - tekst	ord = “Hei”

- Brukes til å **lagre en verdi**.
- Har en **type** og et **navn.**
- Vi kan **forandre** verdien så mange ganger vi ønsker.

Uttrykk

```
tall = 10*2
# Uttrykket evalueres og tall får da verdien
# 20.

erDetSant = 5 < 10 and tall > 7
# evalueres og verdien til erDetSant blir
# True.
```

- Sammensettningen av flere **verdier** og **operatorer** som evalueres til en eneste verdi.

- Kan være **aritmetiske** (“matematiske”) eller **boolske** (“logiske”).

Aritmetiske uttrykk

Beregning:

- Akkurat som i matematikk beregnes paranteser først.
- Så * og /.
- Sist + og -.

+ Addisjon	tall = 2 + 2
- Subtraksjon	tall = 6 - 2
* Multiplikasjon	tall = 2 * 2
/ Divisjon	tall = 5/2 #får verdien 2.5
// Heltallsdivisjon	tall = 5//2 #får verdien 2
+= Pluss-er-lik	tall = 2, tall += 3
-= Minus-er-lik	tall = 6, tall -= 2
% Modulo	tall = 14 % 10 #gir resten etter heltallsdivisjon.

Boolske uttrykk

- Evalueres til enten True eller False.
- Brukes i if/elif og løkker.

== Likhet	erSann = 1 == 1	erUsann = 2 == 3
> Større enn	erSann = 6 > 2	erUsann = 6 > 6
>= Større enn, eller lik	erSann = 6 >= 6	er Usann = 5 >= 6
< Mindre enn	erSann = 7 < 3	erUsann = 4 < 2
<= Mindre enn, eller lik	erSann = 2 <= 5	erUsann = 2 <= 1
and - Og	erSann = 1 == 1 and 6 > 3	erUsann = True and 2 < 1
or - Eller	erSann = 1 == 1 or 6 < 3	erUsann = 4 < 2 or 2 == 3
not - Ikke	erSann = not (1==1 and 6==2)	erUsann = not(1==1)

Bruk av boolske uttrykk

```
inn = input("Skriv inn ett tall")
inn_tall = int(inn)

if inn_tall < 10 and inn_tall > 0:
    print("Dette er melom 0 og 10")
elif inn_tall < 20 and inn_tall > 0:
    print("Dette er over 10, mindre enn 20")
else:
    print("Dette er over 20 eller mindre enn 0")
```

- Evalueres til enten True eller False.
- Brukes i if/elif og løkker.

#Antar at input er 7

```
if 7 < 10 and 7 > 0:
    print("Dette er melom 0 og 10")
...
```

```
if True and True:
    print("Dette er melom 0 og 10")
...
```

```
if True:
    print("Dette er melom 0 og 10")
...
```

Bruk av boolske uttrykk

```
inn = input("Skriv inn ett tall")
inn_tall = int(inn)

if inn_tall < 10 and inn_tall > 0:
    print("Dette er melom 0 og 10")
elif inn_tall < 20 and inn_tall > 0:
    print("Dette er over 10, mindre enn 20")
else:
    print("Dette er over 20 eller mindre enn 0")
```

- Evalueres til enten True eller False.
- Brukes i if/elif og løkker.

#Antar at input er 12

```
if 12 < 10 and 12 > 0:
```

```
if False and True:
```

```
if False:
#Siden false > gaar videre til elif
```

```
elif 12 < 20 and 12 > 0:
    print("Dette er over 10, mindre enn 20")
```

```
elif True and True:
    print("Dette er over 10, mindre enn 20")
```

```
elif True:
    print("Dette er over 10, mindre enn 20")
```

Konstant

```
def program():
      MIN_TALL = 0
      tall = 12
      if tall >= MIN_TALL:
            print(str(tall) + " er et
positivt tall”)
```

Ikke så stort fokus på konstanter i IN1000.

MEN:

Kodeskikk å benytte store bokstaver dersom man skal representere variabel som skal være konstant (og ikke forandres gjennom programmet).

Skop

Skop handler om tilgangen til variabler, hvor variblene er “synlige” for andre deler av programmet.

En variabel må være laget/deklarert før den kan brukes.

Global variabel - tilgjengelig for hele programmet.

Lokal variabel - variabel opprettet i en funksjon og dermed kun mulig (og synlig) å bruke inne i funksjonen.

Kodeskikk i Python:

Unngå globale variabler!

Dersom det i en funksjon finnes en global og en lokal variabel med samme navn, vil den lokale brukes først av programmet.

Skop, noen eksempler

```
Eksempel 1:
def f():
    print(s)
s = "I love Paris in the summer!"
f()
```

> I love Paris in the summer!

```
Eksempel 2:
def f():
    s = "I love London!"
    print(s)

s = "I love Paris!"
f()
print(s)
```

> I love London!
> I love Paris!

Skop, noen eksempler

```
Eksempel 3, kombinasjon av 1 og 2:

def f():
    print(s)
    s = "I love London!"
    print(s)

s = "I love Paris!"
f()
```

> Traceback (most recent call last):
  File "tull.py", line 45, in <module>
    f()
  File "tull.py", line 40, in f
    print(s)
UnboundLocalError: local variable 's' referenced before assignment

Blokker

```
def funksjon():
    heltall = 12
    if (heltall == 12):
        tekst = "Tallet er 12!"

class Hund:
    def __init__(self, navn):
        self._navn = navn
        self._alder = 0

    def haBursdag(self):
        for i in range(3):
            print("Bjeff")
        self._alder += 1
```

En blokk vil være det som er på samme **indentering**snivå, inni en funksjon eller klasse.

Man kan ha en blokk inni en annen blokk.

En blokk definerer et **skop.**

Spørsmål?

... og oppgaver

Informasjon hentet fra forelesning og repetisjonskurs 2016.