UNIVERSITETET I BERGEN

INNFØRING I PROGRAMMERING

INF100

HØST 2022

Torstein Strømme

HVEM ER VI?

Torstein Strømme

- Emneansvarlig
- Kontor: 404P2
  (4. etg. i datablokk på Høyteknologisenteret)

- BSc i elektronikk
- MSc og PhD i informatikk
- Jobbet i oljen og i mediebransjen
- Fast ansatt ved UiB i 1 år
- To barn
- Kan utfordres i bordtennis

HVEM ER VI?

Eirik Rekve Thorsheim

- Adminstrator for INF100
- Kontor: 407M2
  (4. etg. i datablokk på Høyteknologisenteret)

- Master i sammenlignende politikk
- Har laget radio om amerikansk sport
- Kan utfordres i quiz

HVEM ER VI?

David Grellscheid
- Faglærer

Sondre Bolland
- Faglærer

Maryam Yousefian
- Undervisningsassistent

Gutama Ibrahim Mohammad
- Undervisningsassistent

GRUPPELEDERE

-- lang liste med navn her --

EN UKE I INF100

Tors	Fre

	Man	Tirs	Ons	Tors	Fre
8-10					
10-12					
12-14					
14-16	Fo.les.				

	Man	Tirs
8-10		
10-12		
12-14		
14-16		

Fredag 16:00
Påmelding åpner
for fysisk oppmøte
til forelesning.

Mandag 16:00
Ukesoppgaver publiseres

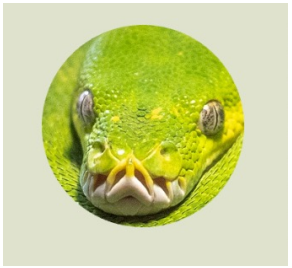
I løpet av uken
- 2x2 timer gruppe +
- 4-10 timer egenstudie

Mandag 23:59
Innleveringsfrist

PLATTFORMER

mitt.uib.no

- Kunngjøringer
- Oppgaveinnlevering
- Strømming av forelesning
- Opptak av forelesning



inf100.ii.uib.no

- Kursnotater
- Oppgavetekster

Discord

- Spørsmål og svar
- Tips og triks

ARBEIDSKRAV

For å ta eksamen må du enten

- Få mer en 80% riktig på kartleggingsprøve, eller
- Bestå minst 4/6 av de første obligatoriske ukesoppgavene

I tillegg må du

- Bestå minst 3/5 av de siste obligatoriske ukesoppgavene

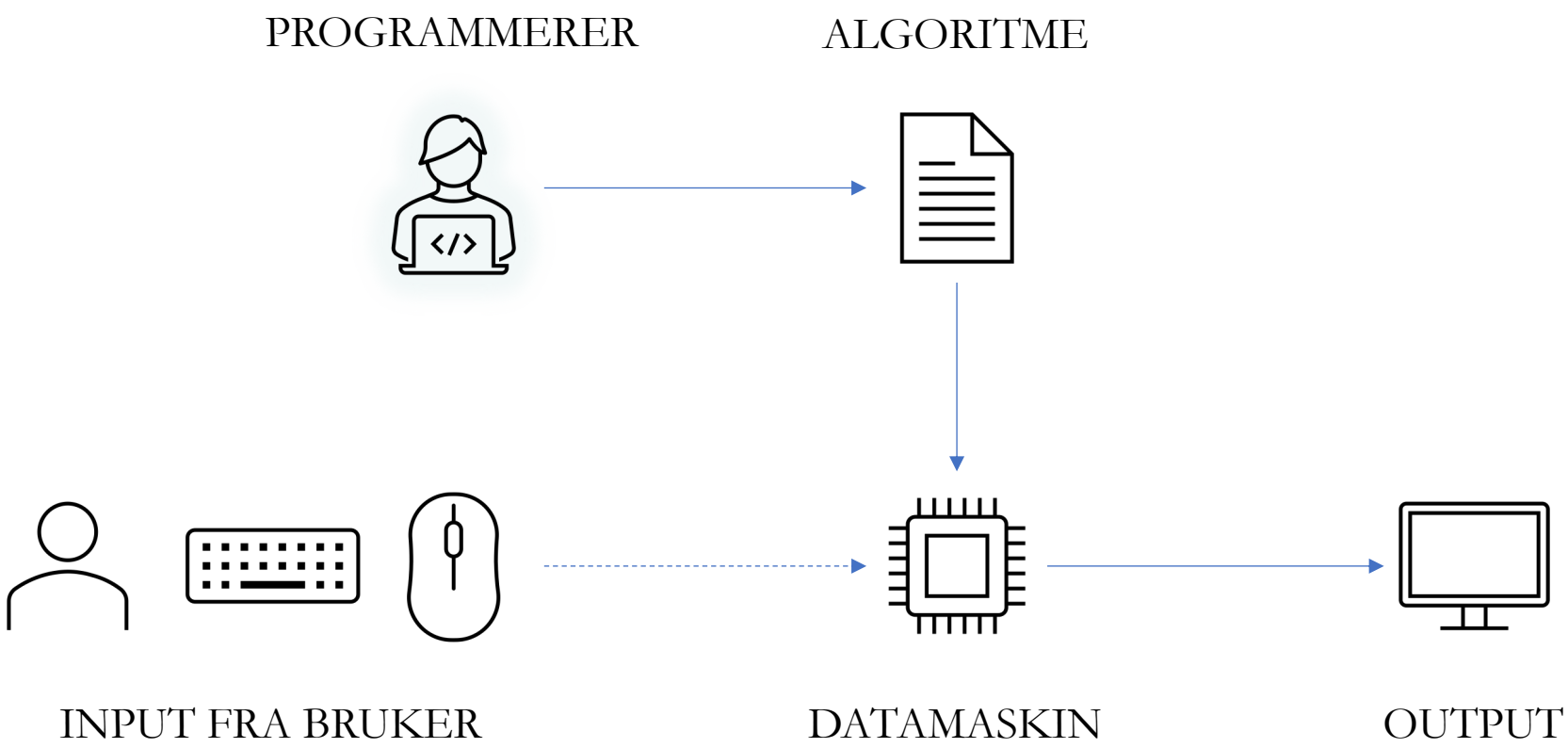
Det vil ikke være mulig å få utsettelser annet enn i helt spesielle og ekstraordinære tilfeller som varer over lengre tid.

KARTLEGGINGSPRØVE

- Frivillig
- Automatisk rettet
- Over 80% → du slipper de 6 første obligatoriske innleveringene

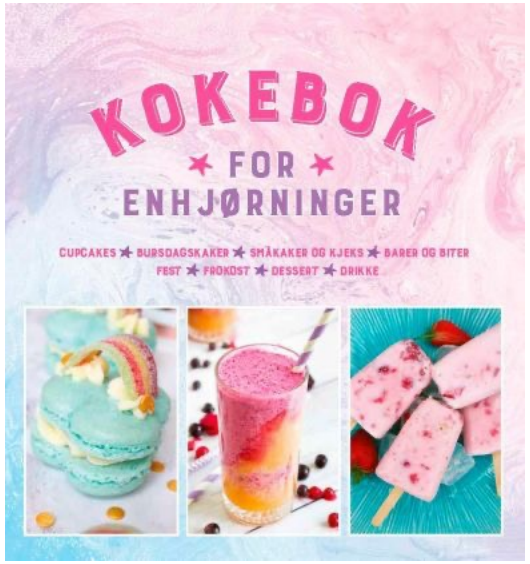
- Fredag 26. august
  - Audiotorium "Egget" på Studentsenteret
  - 14:15 – 16:00
  - 16:15 – 18:00
  - Påmeldingsfrist onsdag 24. august 23:59

PROGRAMMERING


PROGRAMMERER
ALGORITME
INPUT FRA BRUKER
DATAMASKIN
OUTPUT

ALGORITME

- Instruksjoner for hvordan noe skal gjøres

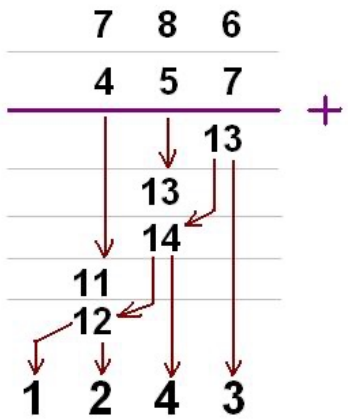


https://www.forbes.com › melissahouston › 2021/08/11

How To Get Rich From Absolutely Nothing - Fo

Aug 11, 2021 — To build **wealth** you need to **have** some fundamer

mindset is everything · 2. Millionaires still budget · 3. Money mana

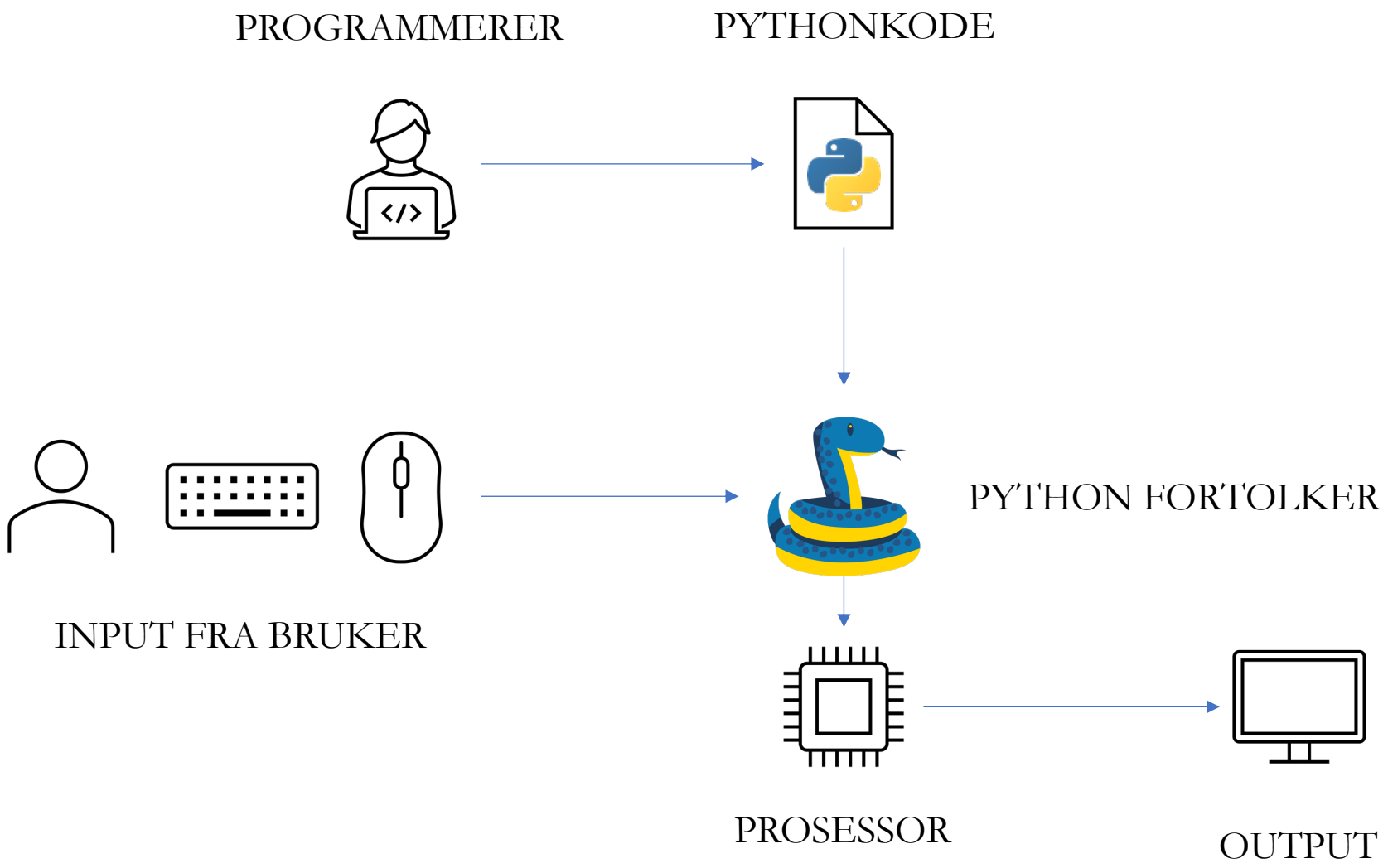


Male gulv - Slik gjør du det

YouTube · Byggmakker

Oct 7, 2020

PYTHON


PROGRAMMERER
PYTHONKODE
PYTHON FORTOLKER
INPUT FRA BRUKER
PROSESSOR
OUTPUT

I DAG:

- ordbok
  - setning
  - funksjon
  - verdi
  - operasjon
  - verdi
  - variabel

- funksjoner
- variabler

ORDLISTE

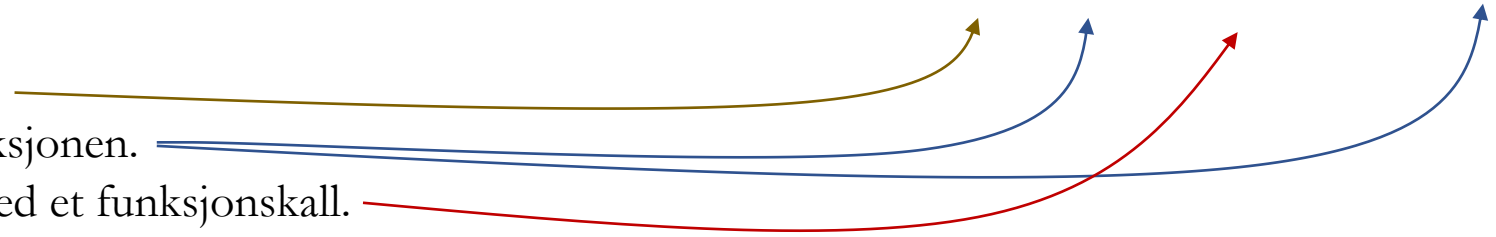
setning (engelsk: statement). ett steg i et program. Ofte tilsvarer hver linje en setning.

```
print("Hello World")
```

funksjon. en navngitt sekvens av setninger

kall. instruksjon om å utføre funksjonen.

argument. verdi gitt som input ved et funksjonskall.



verdi. "noe" (data) som benyttes i programmet. Kalles også objekt.

```
"Hello"    42    3.14    True
```

operasjon. en måte å kombinere to verdier for å produsere en ny verdi.

```
+ - * / // % == != <
> <= >= and or not
```

uttrykk (engelsk: expression). et regnestykke som evaluerer til én verdi. Er en kombinasjon av verdier, operasjoner, variabler og funksjoner.

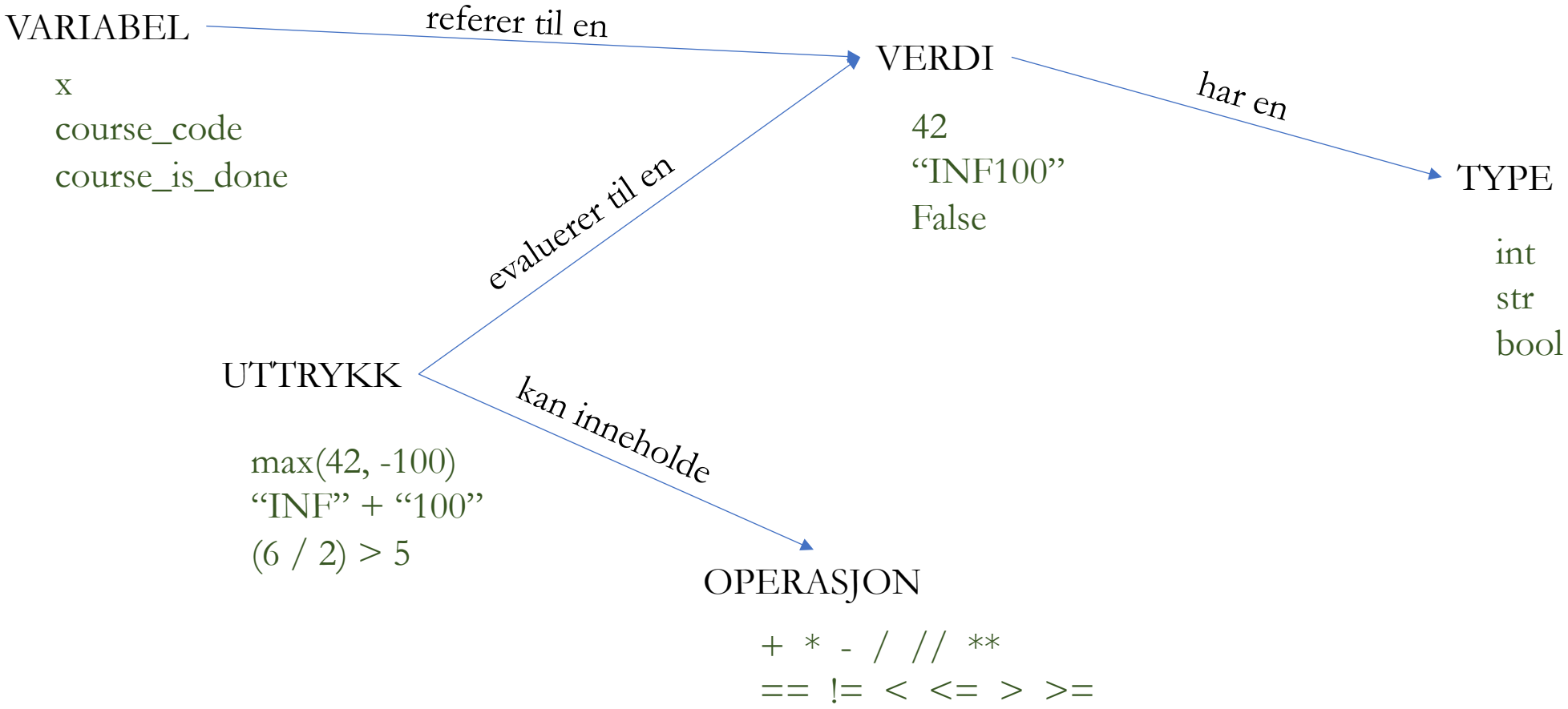
```
"Hei "+name    6*7    x/2    (3*(10+4)) == x
```

type. alle verdier har en type. Typen bestemmer hva en operasjoner betyr.

```
str    int    float    bool
```

variabel. en navngitt referanse til en verdi. Vil evaluere til gitt verdi om benyttet i et uttrykk.

```
x = 42          name = "Torstein"
```


VARIABEL
x
course_code
course_is_done
referer til en
VERDI
42
“INF100”
False
har en
TYPE
int
str
bool
evaluerer til en
UTTRYKK
max(42, -100)
“INF” + “100”
(6 / 2) > 5
kan innehold
OPERASJON
+ * - / // **
== != < <= > >=

REPETISJON

www.menti.com

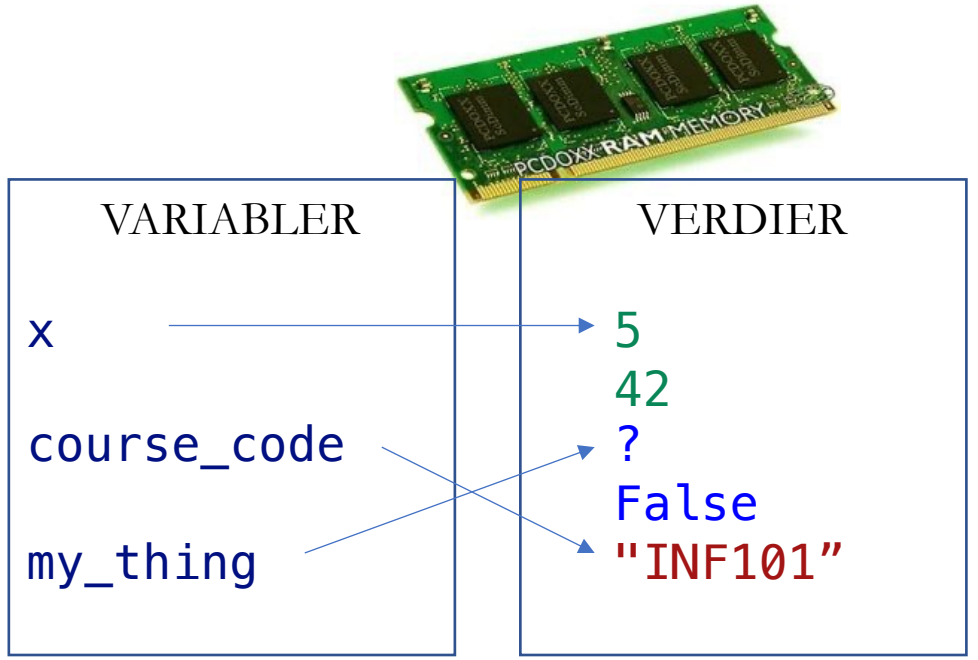
4719 4538

VARIABLER

```
x = 5
```

```
course_code = "INF101”
```

- En navngitt referanse til en verdi
- Tolkes som verdi



VARIABLER

```
x = 10
print(x)

x = True
x = "Hei"
x = 11
x = x + 1
print(x)
x + 1
print(x)

y = x + 5
print(y)

x = x + 10
print(x)
print(y)
```

VARIABLER

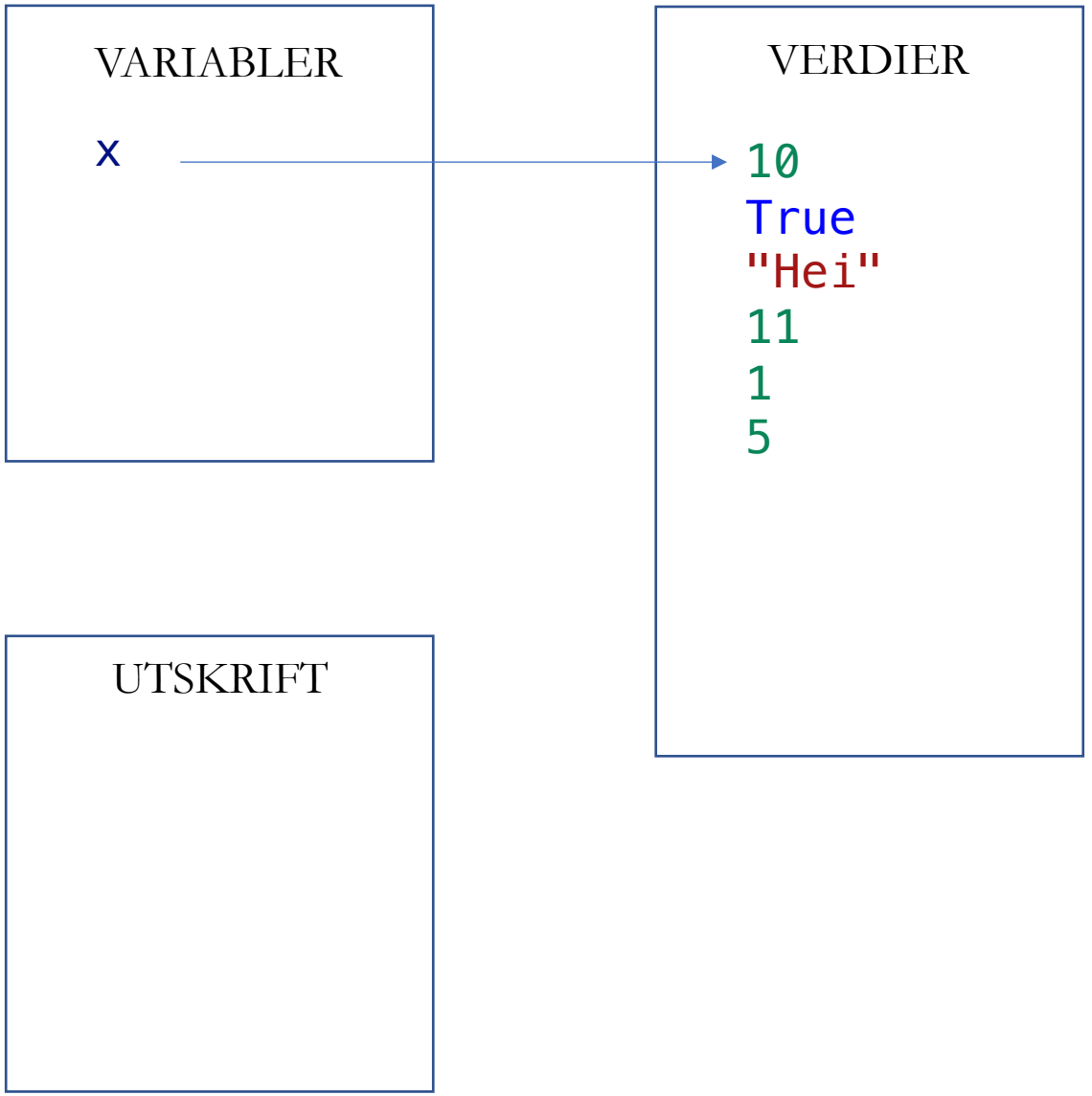
UTSKRIFT

VERDIER

```
10
True
"Hei"
11
1
5
```

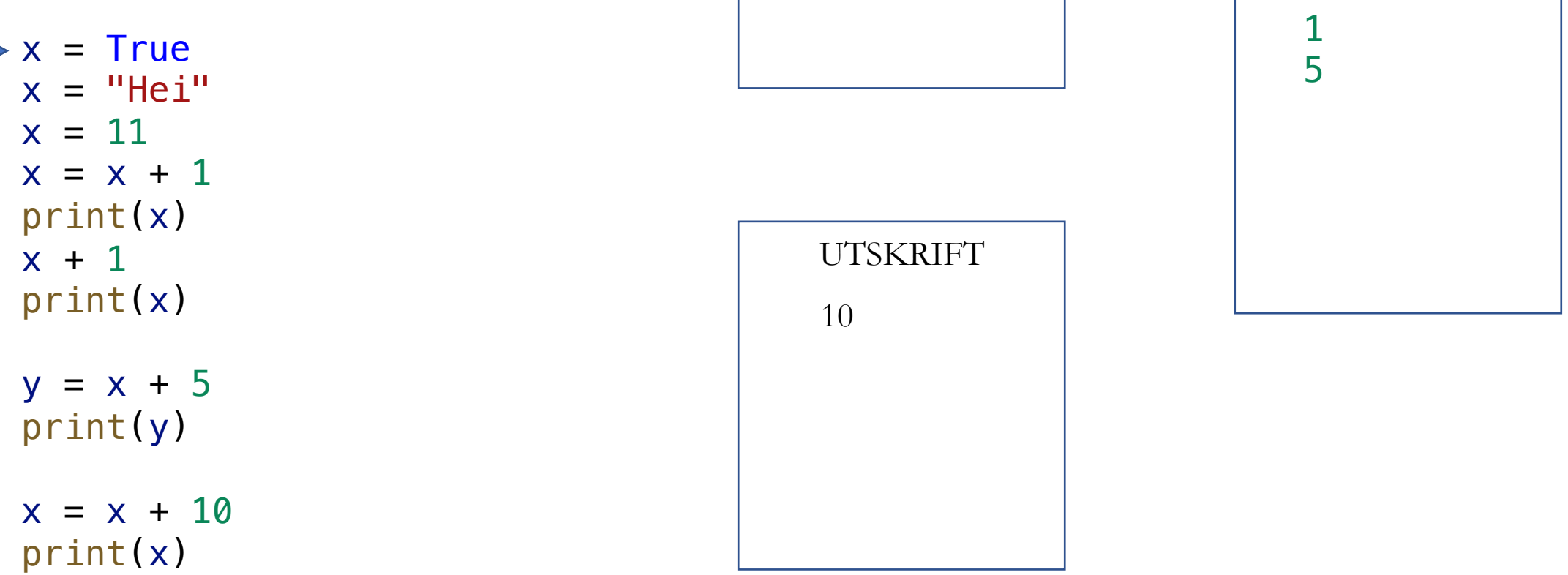
VARIABLER

```
x = 10
print(x)

x = True
x = "Hei"
x = 11
x = x + 1
print(x)
x + 1
print(x)

y = x + 5
print(y)

x = x + 10
print(x)
print(y)
```



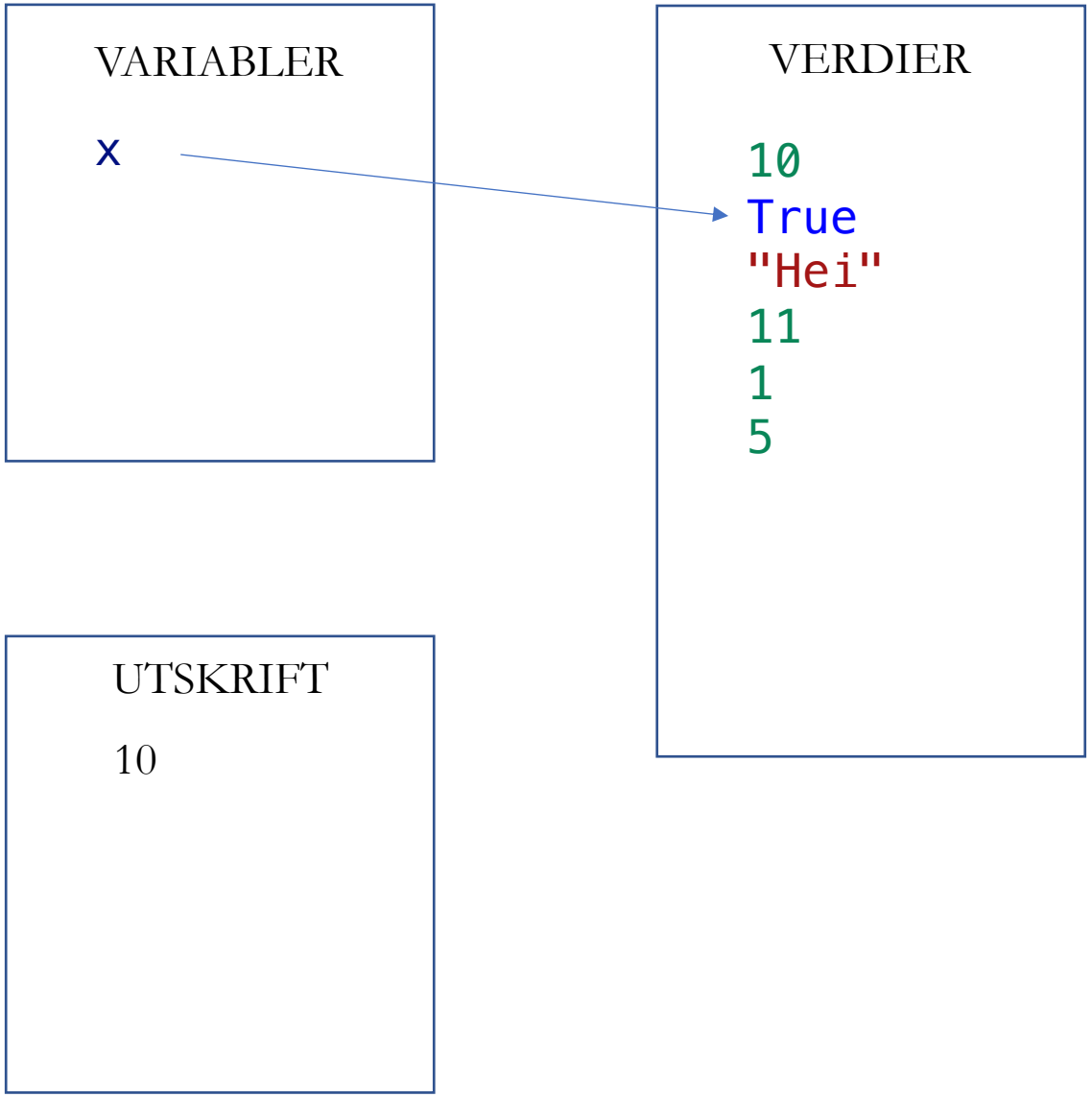
VARIABLER

```
x = 10
print(x)

x = True
x = "Hei"
x = 11
x = x + 1
print(x)
x + 1
print(x)

y = x + 5
print(y)

x = x + 10
print(x)
print(y)
```



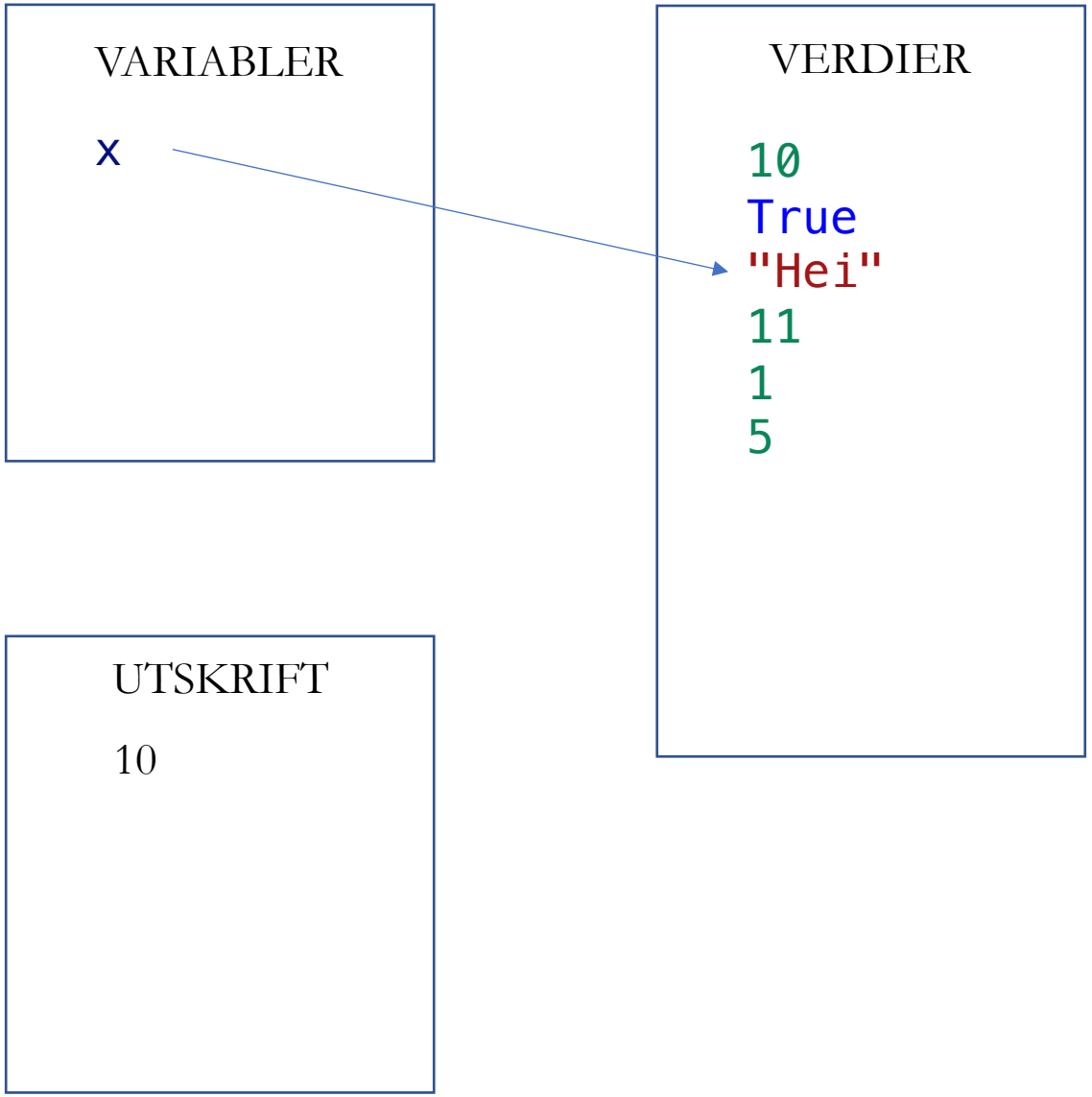
VARIABLER

```
x = 10
print(x)

x = True
x = "Hei"
x = 11
x = x + 1
print(x)
x + 1
print(x)

y = x + 5
print(y)

x = x + 10
print(x)
print(y)
```



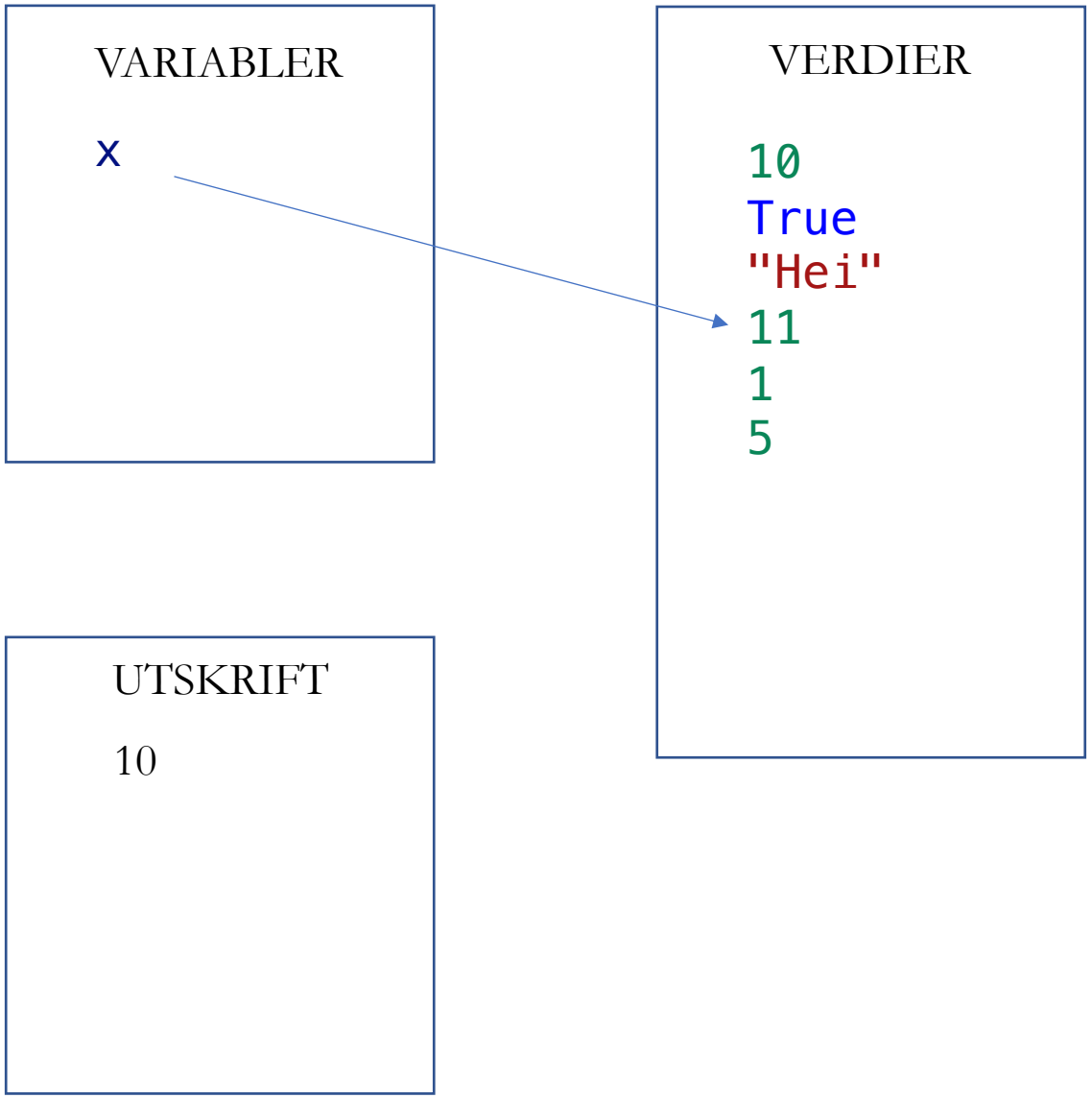
VARIABLER

```
x = 10
print(x)

x = True
x = "Hei"
x = 11
x = x + 1
print(x)
x + 1
print(x)

y = x + 5
print(y)

x = x + 10
print(x)
print(y)
```



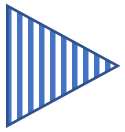
VARIABLER

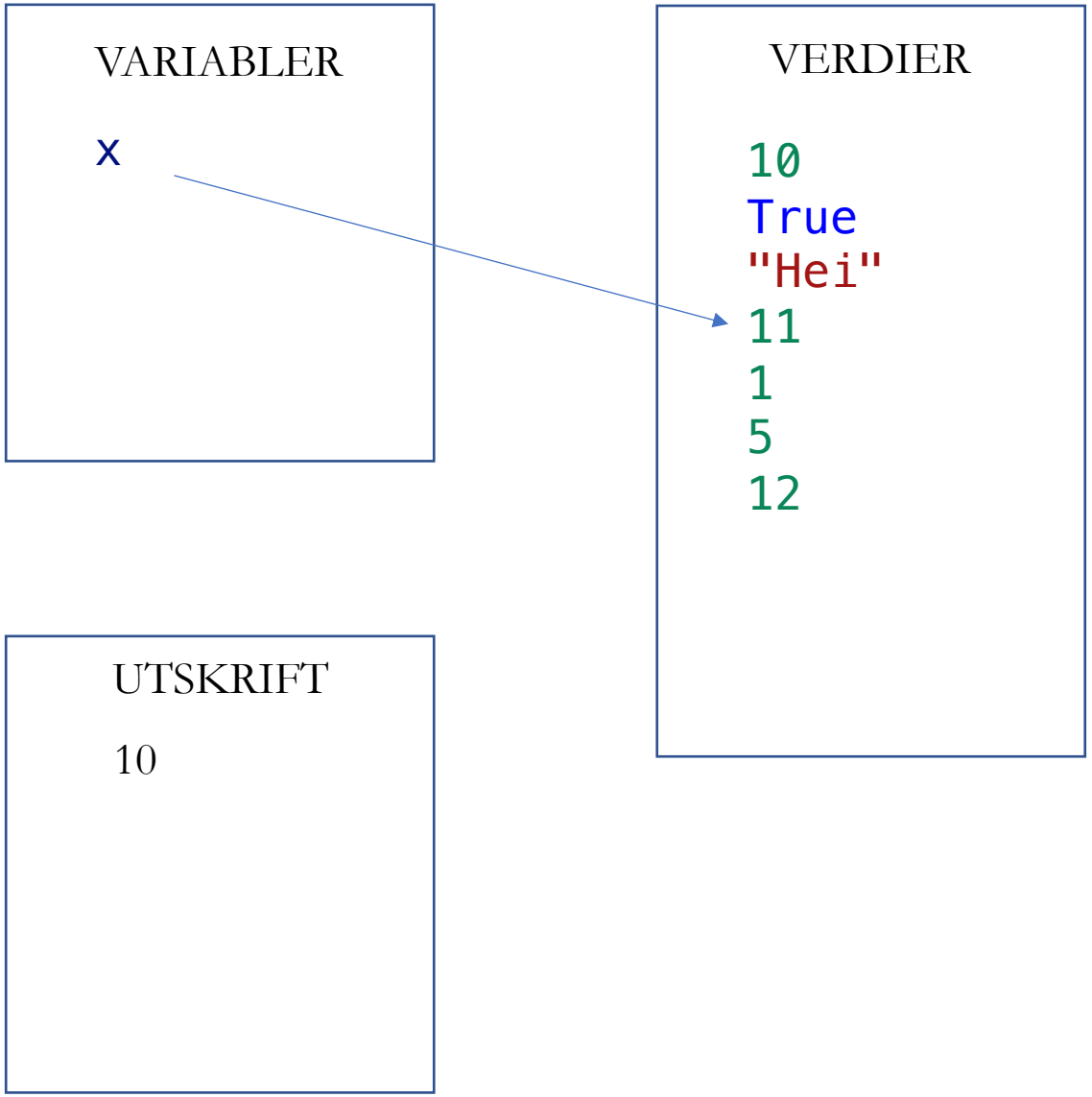
```
x = 10
print(x)

x = True
x = "Hei"
x = 11
x = x + 1
print(x)
x + 1
print(x)

y = x + 5
print(y)

x = x + 10
print(x)
print(y)
```



VARIABLER

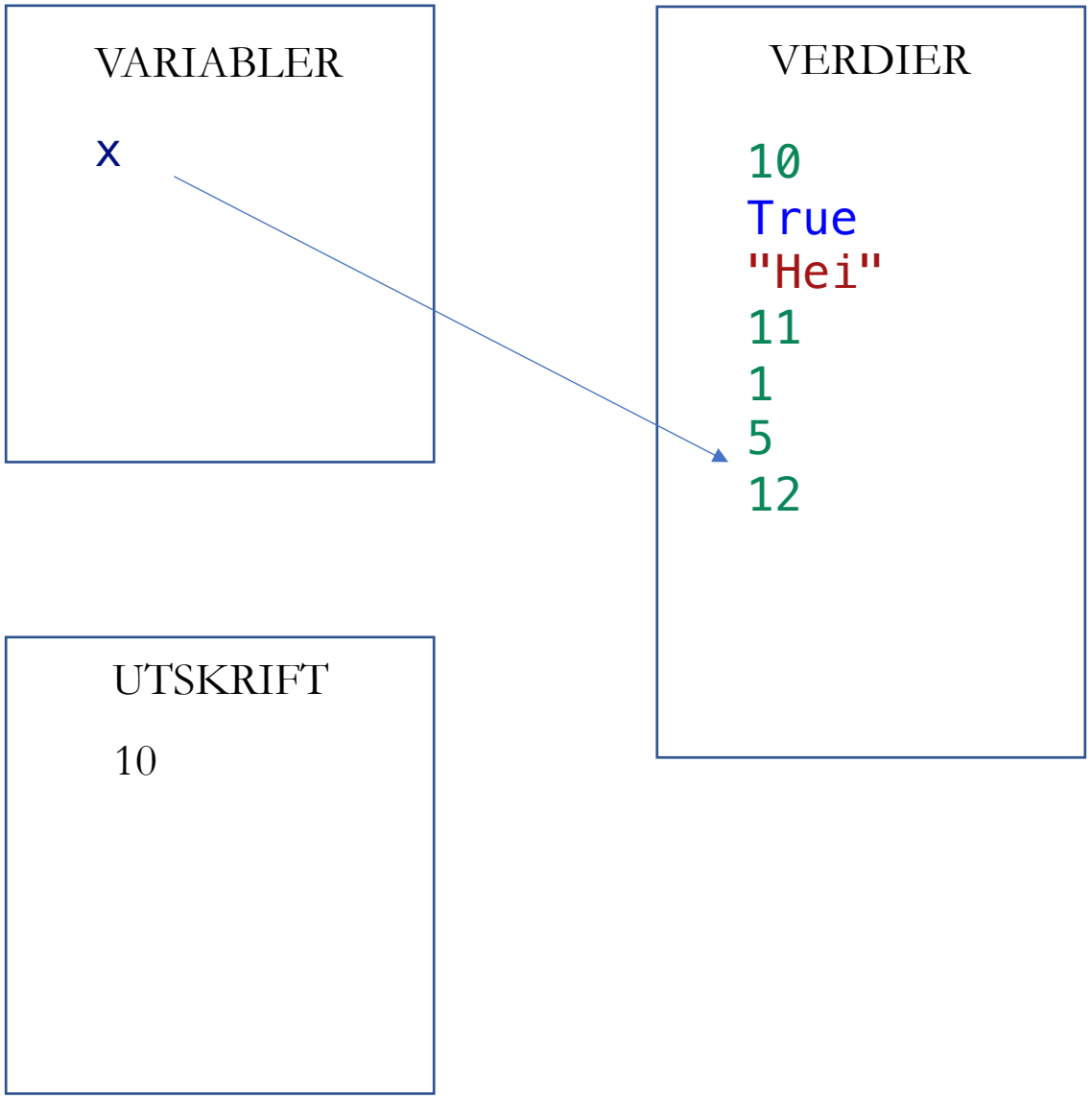
```
x = 10
print(x)

x = True
x = "Hei"
x = 11
x = x + 1
print(x)
x + 1
print(x)

y = x + 5
print(y)

x = x + 10
print(x)
print(y)
```





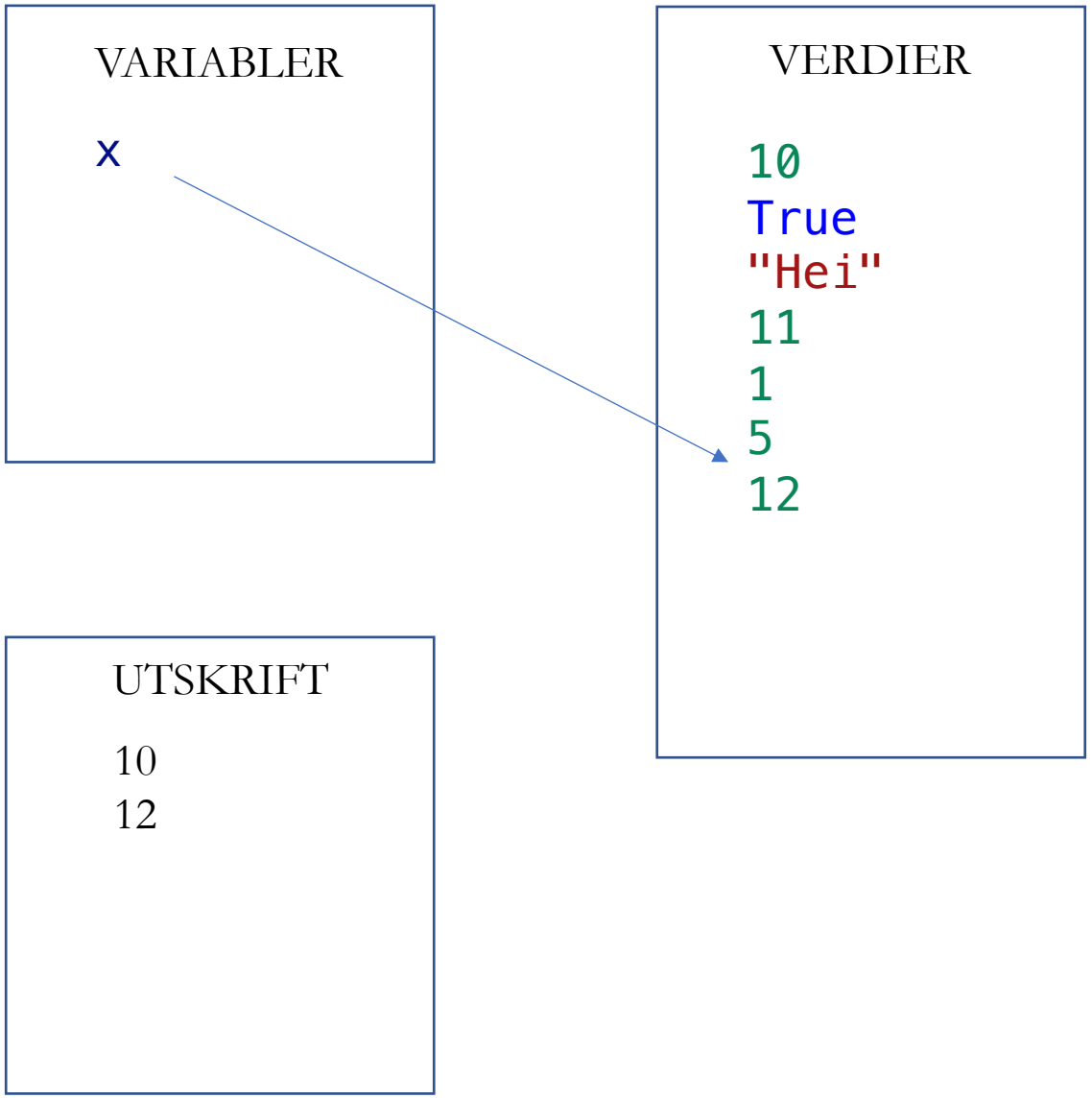
VARIABLER

```
x = 10
print(x)

x = True
x = "Hei"
x = 11
x = x + 1
print(x)
x + 1
print(x)

y = x + 5
print(y)

x = x + 10
print(x)
print(y)
```



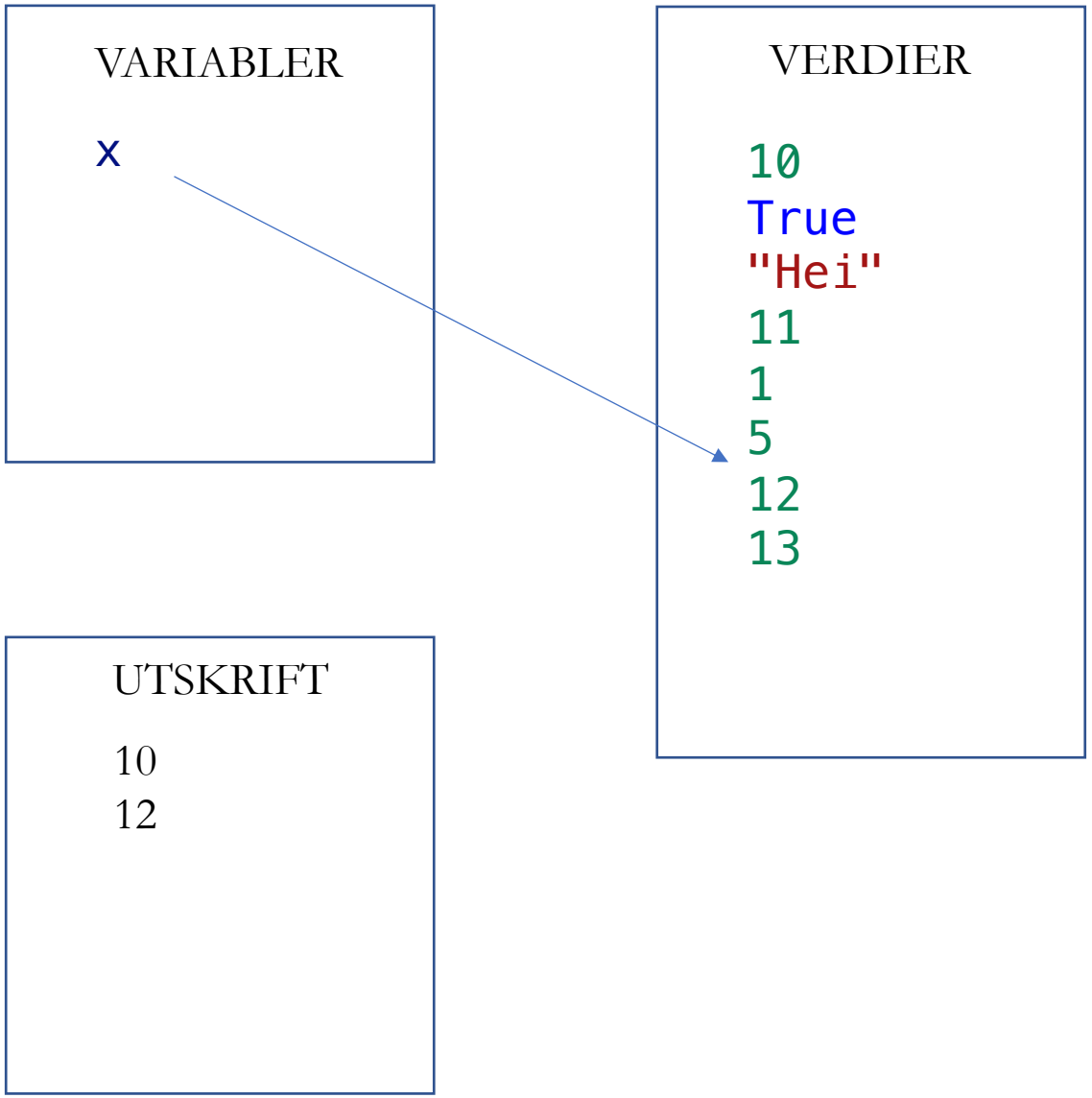
VARIABLER

```
x = 10
print(x)

x = True
x = "Hei"
x = 11
x = x + 1
print(x)
x + 1
print(x)

y = x + 5
print(y)

x = x + 10
print(x)
print(y)
```



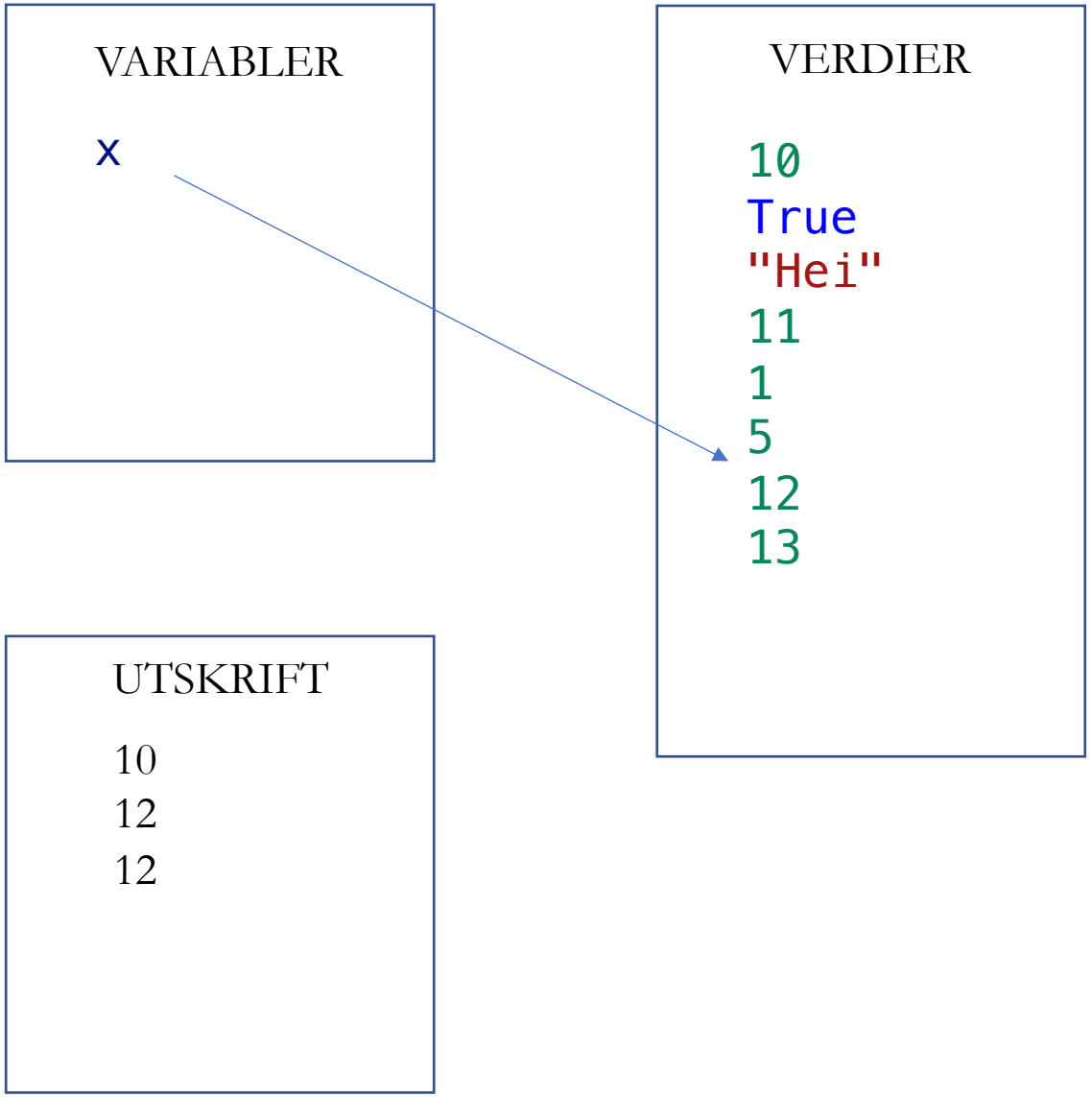
VARIABLER

```
x = 10
print(x)

x = True
x = "Hei"
x = 11
x = x + 1
print(x)
x + 1
print(x)

y = x + 5
print(y)

x = x + 10
print(x)
print(y)
```



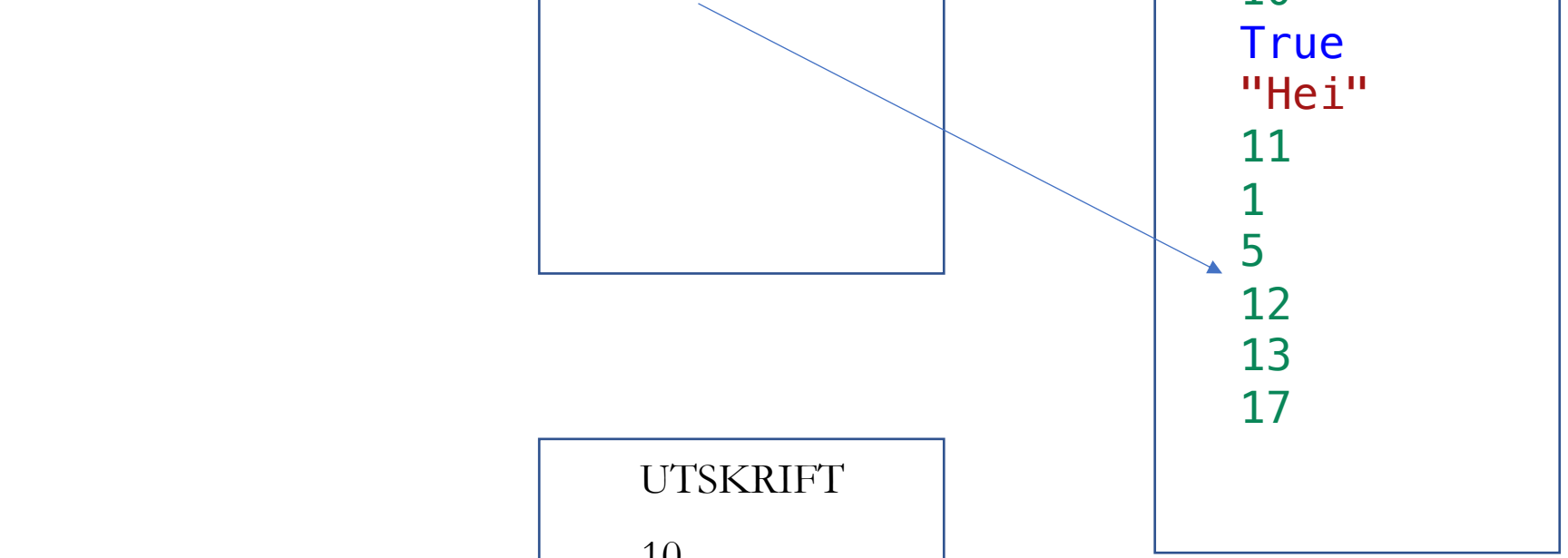
VARIABLER

```
x = 10
print(x)

x = True
x = "Hei"
x = 11
x = x + 1
print(x)
x + 1
print(x)

y = x + 5
print(y)

x = x + 10
print(x)
print(y)
```



VARIABLER

```
x = 10
print(x)

x = True
x = "Hei"
x = 11
x = x + 1
print(x)
x + 1
print(x)

y = x + 5
print(y)

x = x + 10
print(x)
print(y)
```

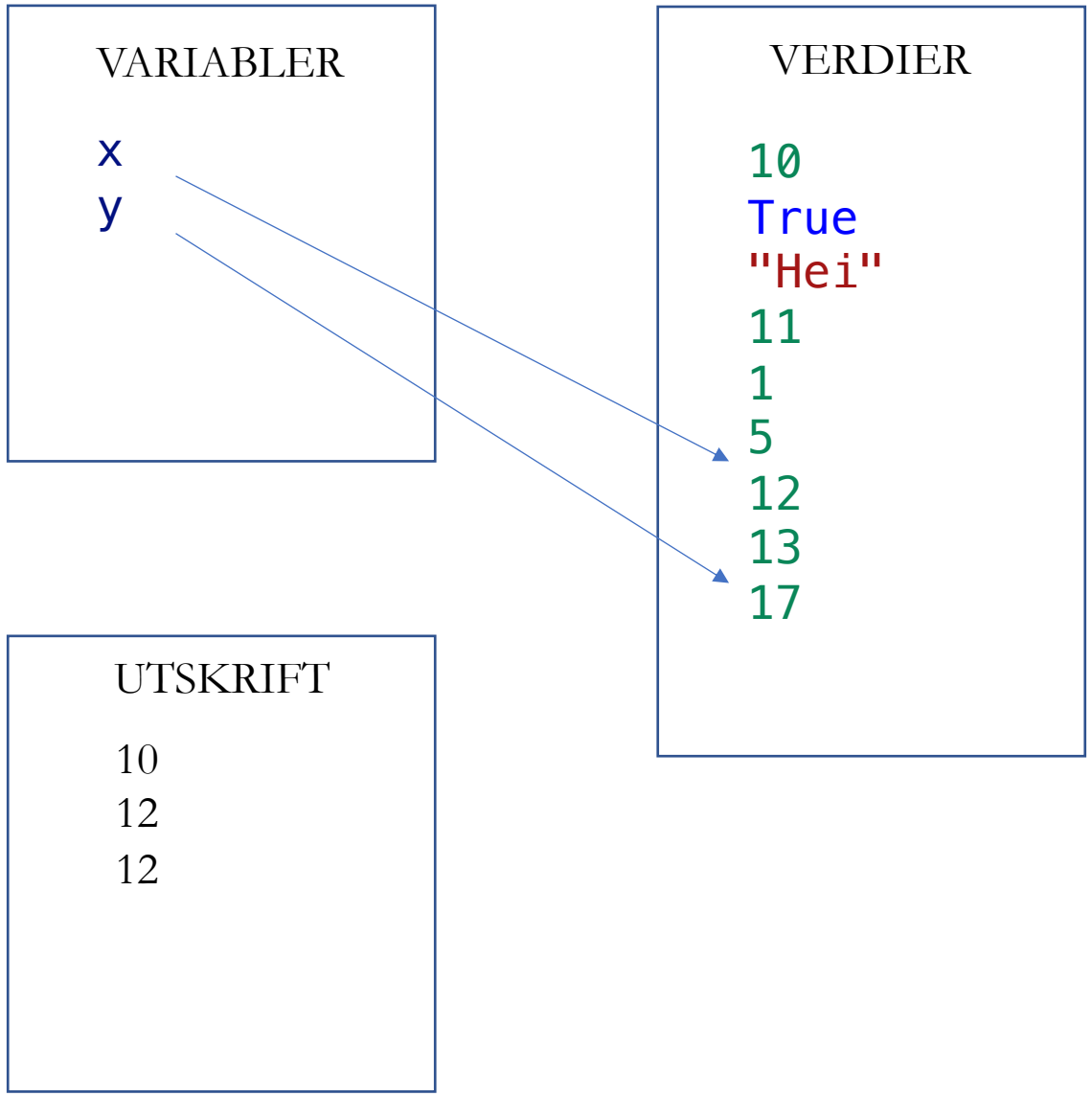


UTSKRIFT

10
12
12

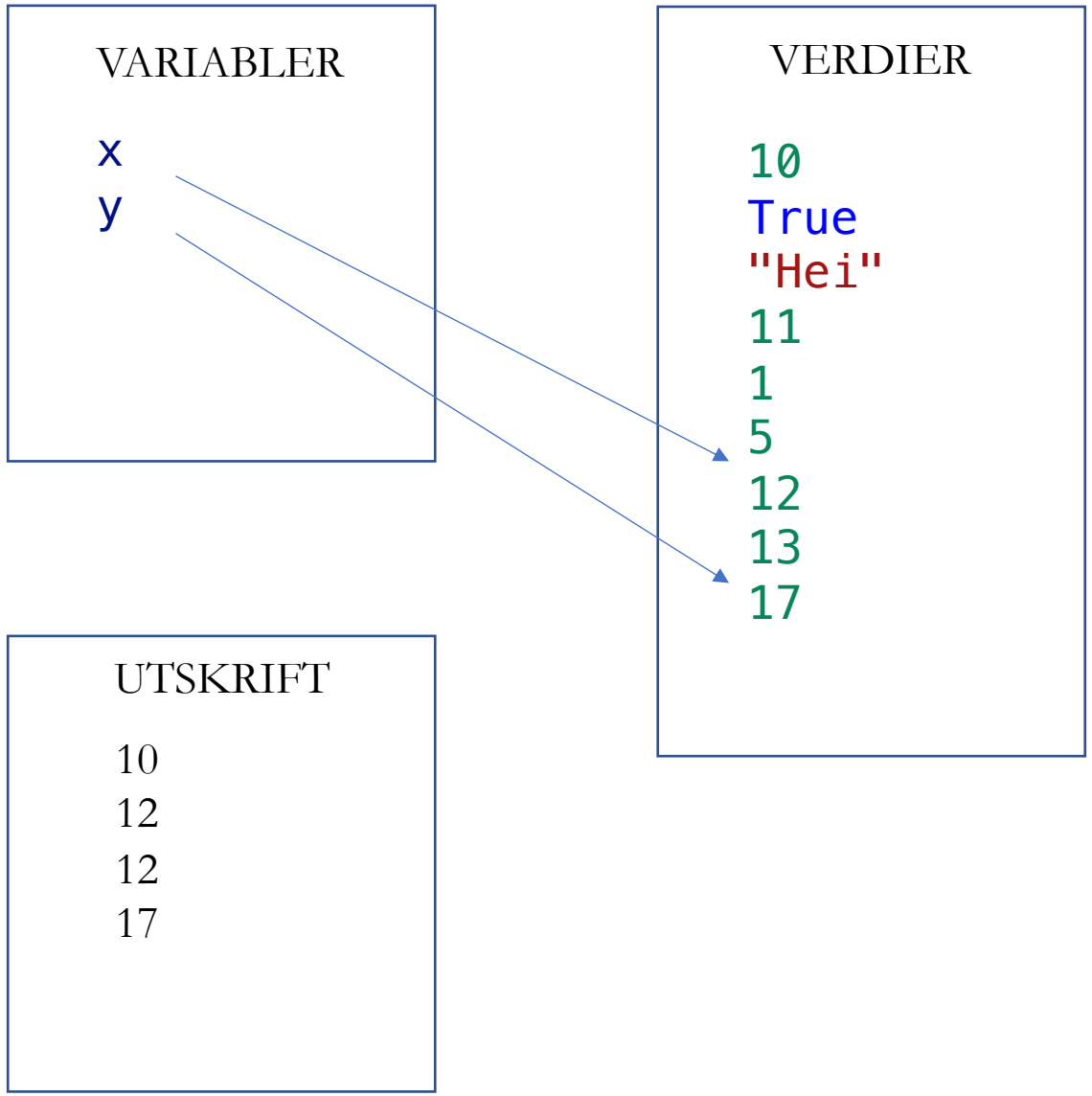
VARIABLER

```
x = 10
print(x)

x = True
x = "Hei"
x = 11
x = x + 1
print(x)
x + 1
print(x)

y = x + 5
print(y)

x = x + 10
print(x)
print(y)
```



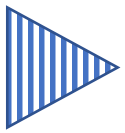
VARIABLER

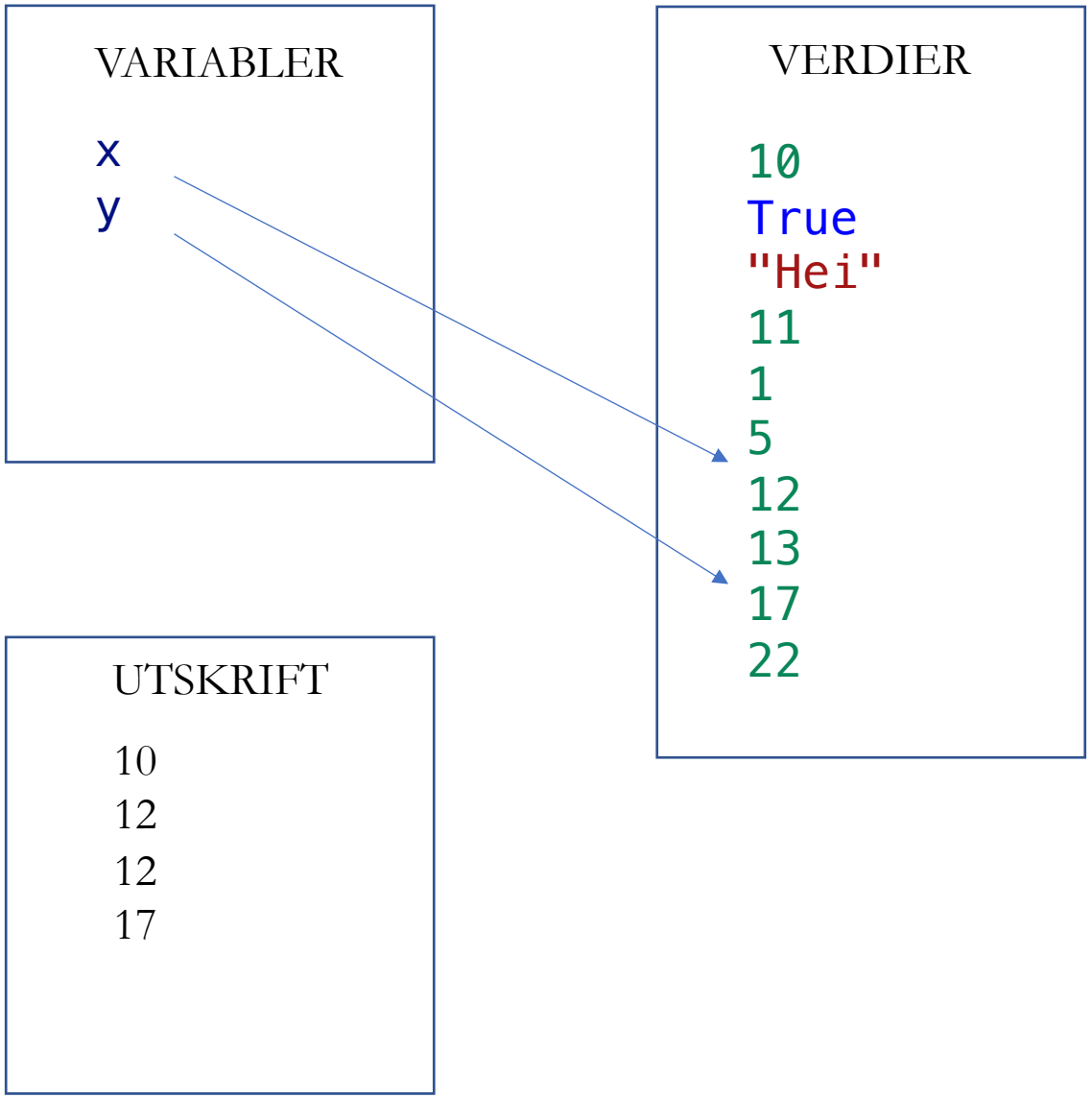
```
x = 10
print(x)

x = True
x = "Hei"
x = 11
x = x + 1
print(x)
x + 1
print(x)

y = x + 5
print(y)

x = x + 10
print(x)
print(y)
```



VARIABLER

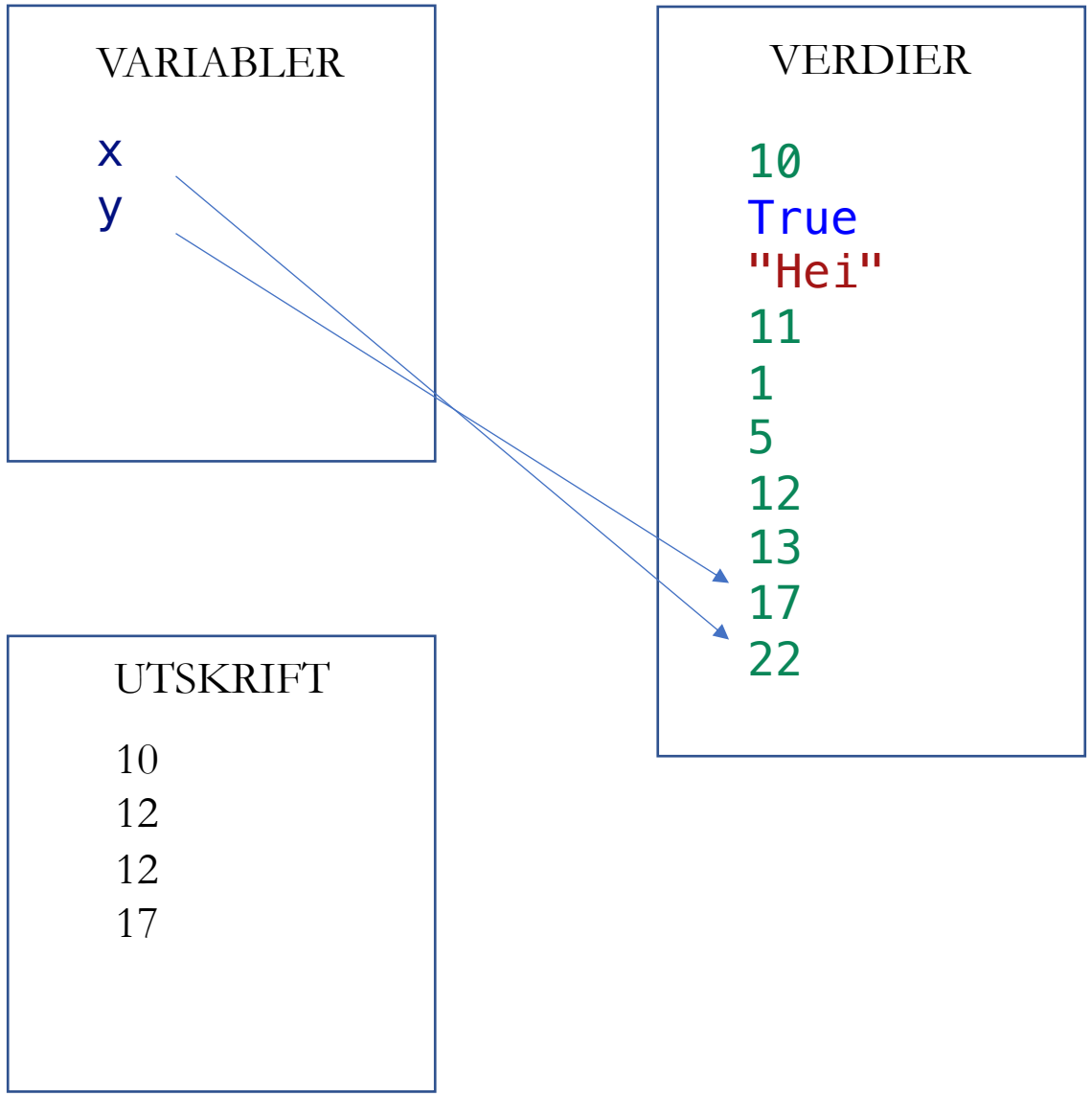
```
x = 10
print(x)

x = True
x = "Hei"
x = 11
x = x + 1
print(x)
x + 1
print(x)

y = x + 5
print(y)

x = x + 10
print(x)
print(y)
```





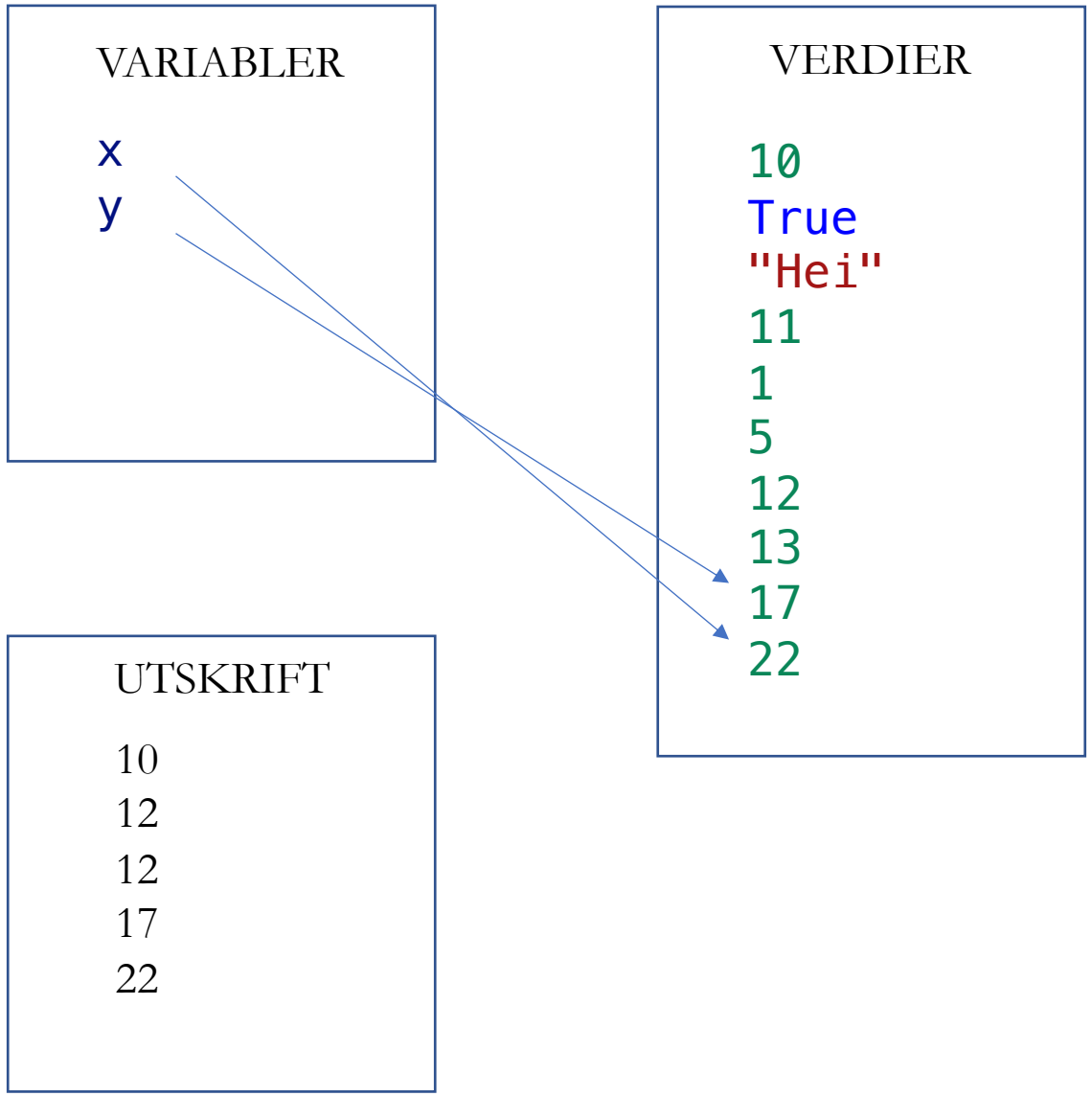
VARIABLER

```
x = 10
print(x)

x = True
x = "Hei"
x = 11
x = x + 1
print(x)
x + 1
print(x)

y = x + 5
print(y)

x = x + 10
print(x)
print(y)
```



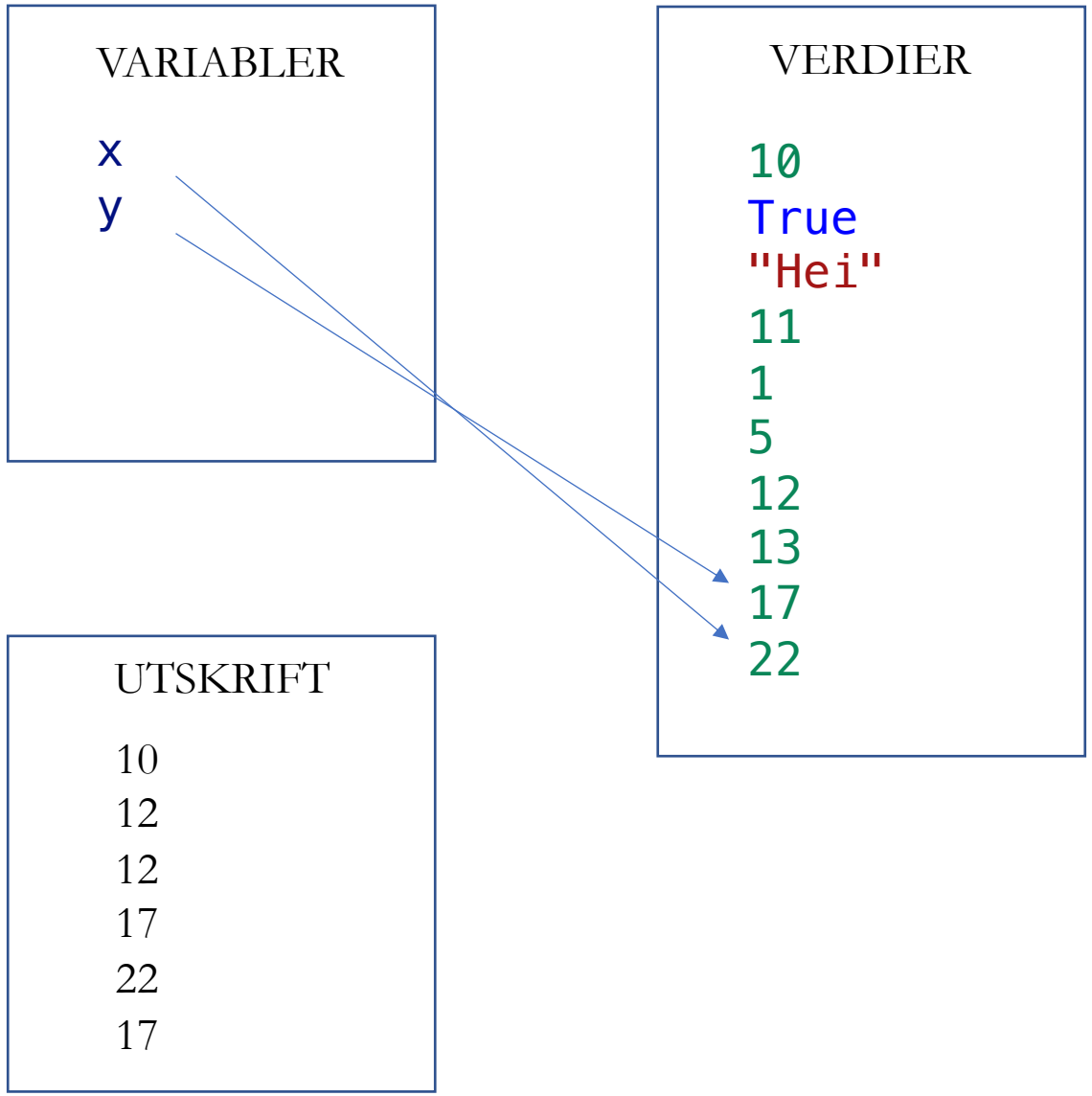
VARIABLER

```
x = 10
print(x)

x = True
x = "Hei"
x = 11
x = x + 1
print(x)
x + 1
print(x)

y = x + 5
print(y)

x = x + 10
print(x)
print(y)
```



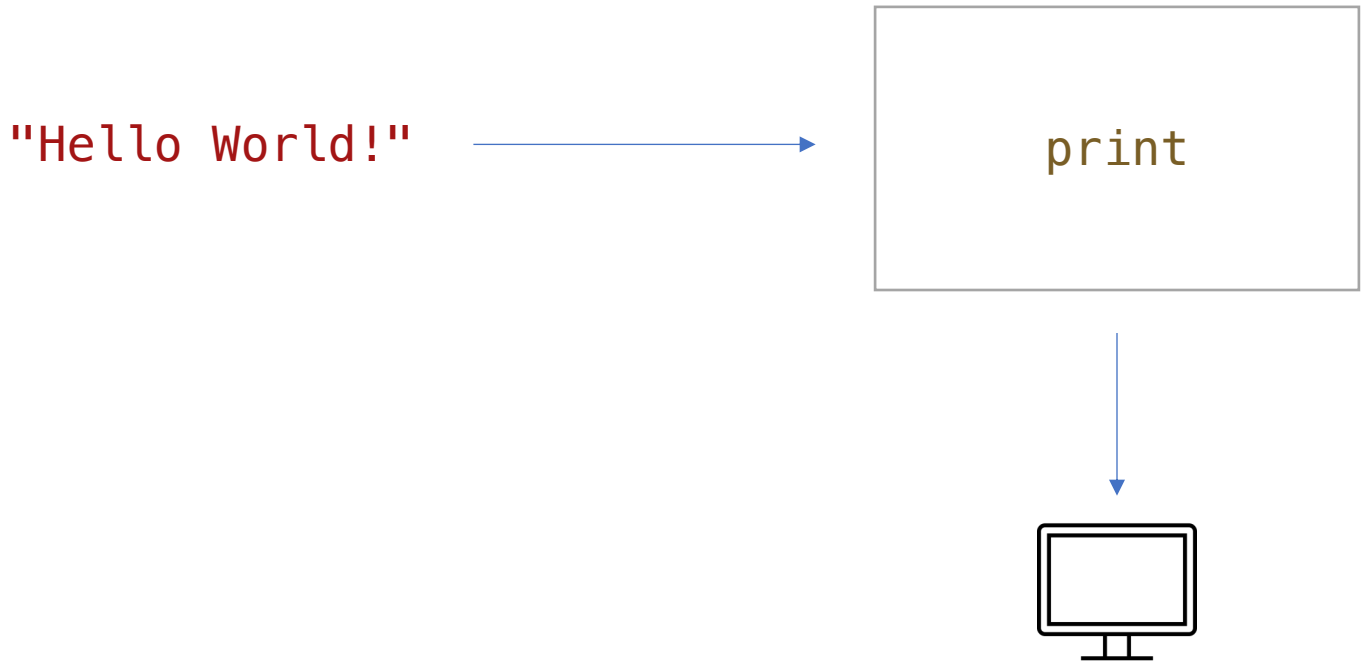
VARIABLER

```
x = 10
print(x)

x = True
x = "Hei"
x = 11
x = x + 1
print(x)
x + 1
print(x)

y = x + 5
print(y)

x = x + 10
print(x)
print(y)
```



FUNKSJONER

```
print("Hello World!")
```



FUNKSJONER MED RETURVERDI

```
max(1, 2, 3)
```

FUNKSJONER MED RETURVERDI

```
print(max(1, 2, 3))
```

INNEBYGDE FUNKSJONER

```
print("Skriv ut noe til terminalen")

len("Lengden av en streng")
sum(1, 2, 3)
min(1, 2, 3)
max(1, 2, 3)
abs(-3)
```

Har returverdi

Å DEFINERE EN FUNKSJON

Vi *definerer* en funksjon som heter print_twice

Mellom parentesene er *parametre* (input til funksjonen)

```
def print_twice(my_thing):
    print(my_thing)
    print(my_thing)

print_twice("Hello World!")
```

Kolon

Blokk. Kode etter kolon som er *indentert* blir utført når funksjonen kalles

my_thing er en *variabel* som refererer til en verdi bestemt av den som kaller funksjonen

EKSEMPELKJØRING

```
def print_twice(my_thing):
    print(my_thing)
    print(my_thing)

print_twice("foo")

x = "bar"
print_twice(x + "!")
```

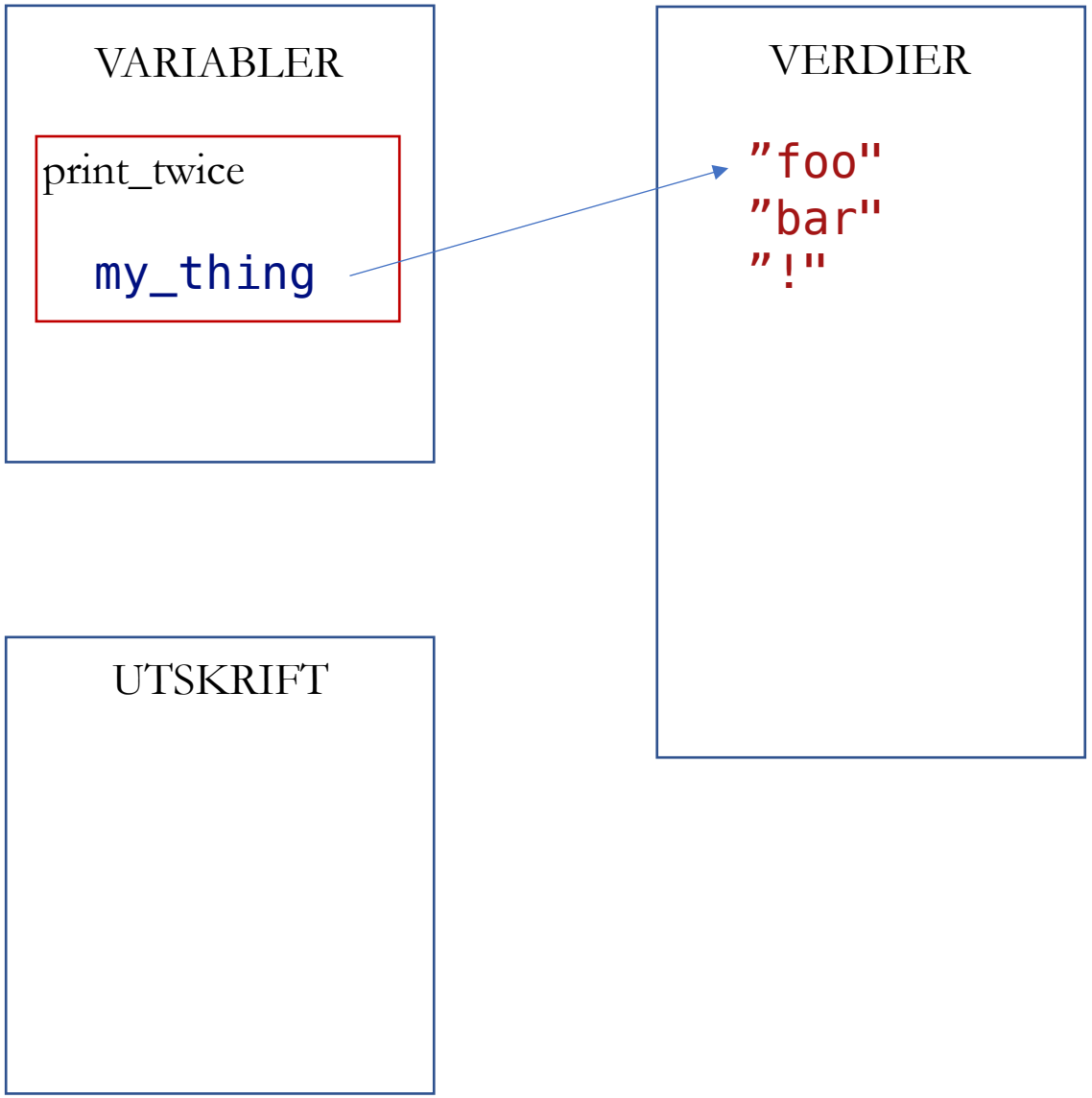
VARIABLER

UTSKRIFT

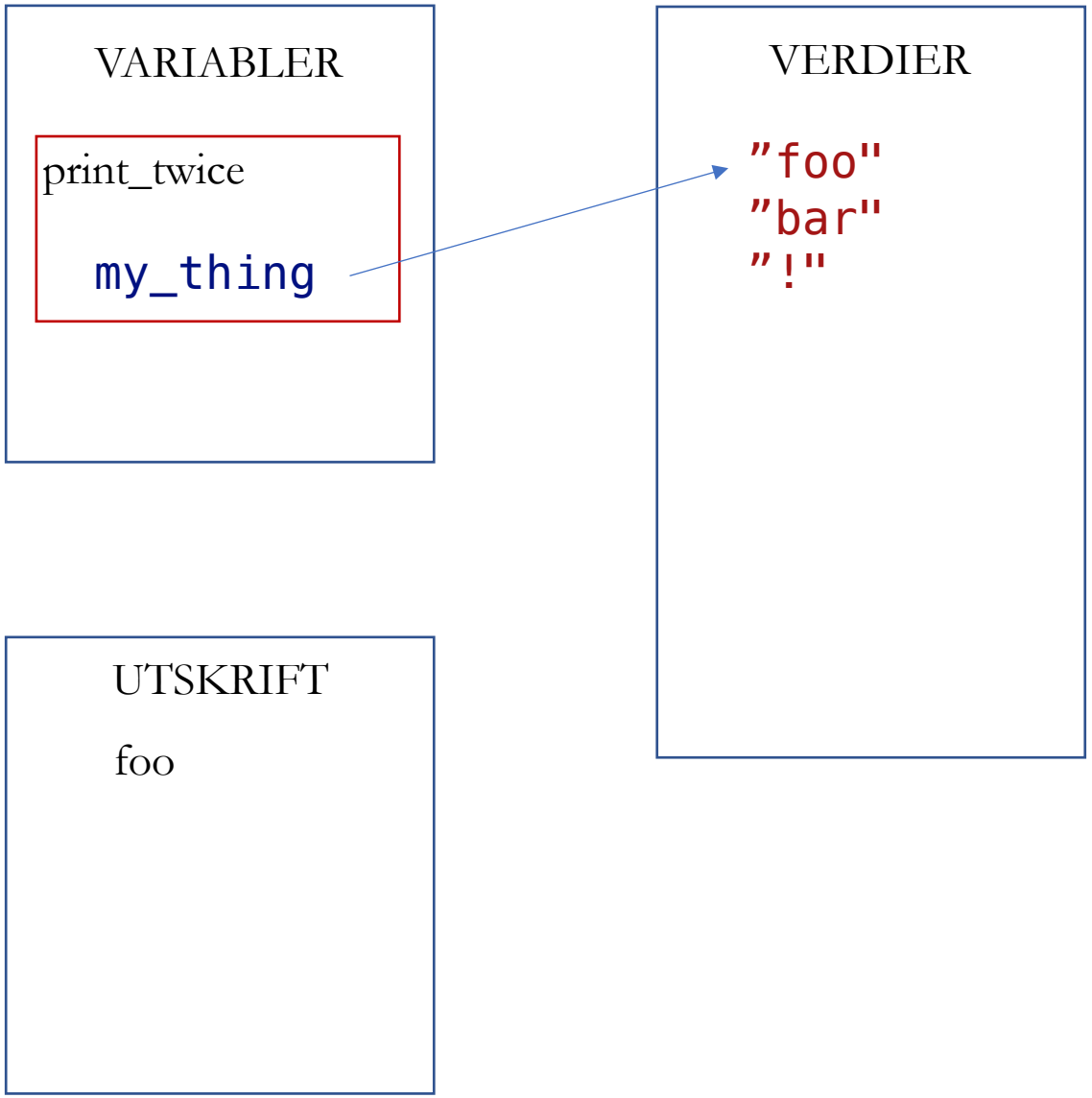
VERDIER

"foo"
"bar"
"!"

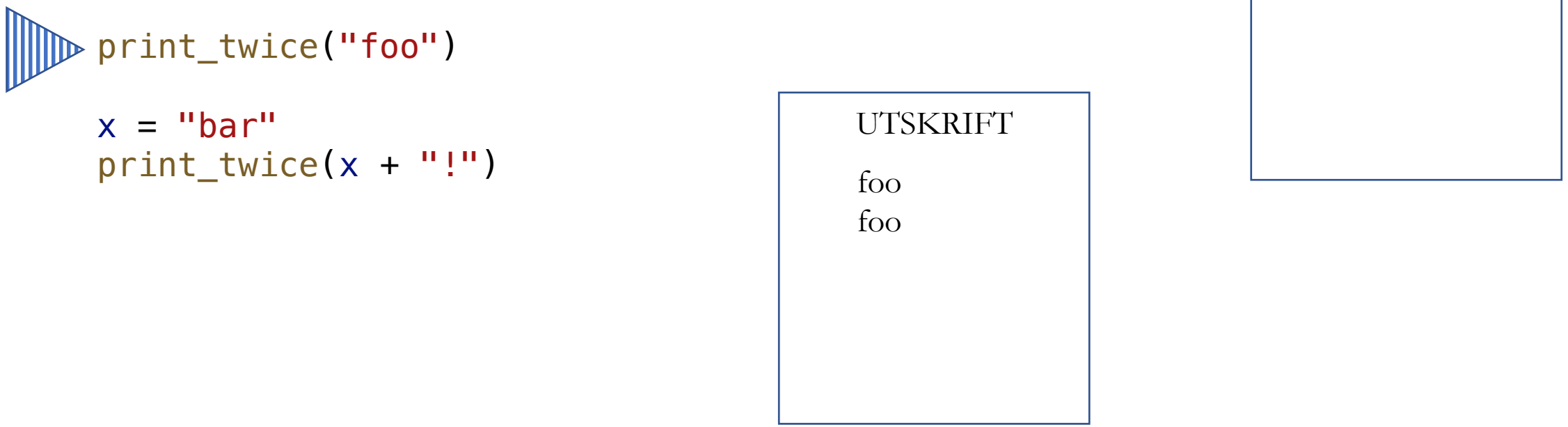
```
def print_twice(my_thing):
    print(my_thing)
    print(my_thing)

print_twice("foo")

x = "bar"
print_twice(x + "!")
```



```
def print_twice(my_thing):
    print(my_thing)
    print(my_thing)

print_twice("foo")

x = "bar"
print_twice(x + "!")
```



```
def print_twice(my_thing):
    print(my_thing)
    print(my_thing)

print_twice("foo")

x = "bar"
print_twice(x + "!")
```



```
def print_twice(my_thing):
    print(my_thing)
    print(my_thing)

print_twice("foo")

x = "bar"
print_twice(x + "!")
```

VARIABLER

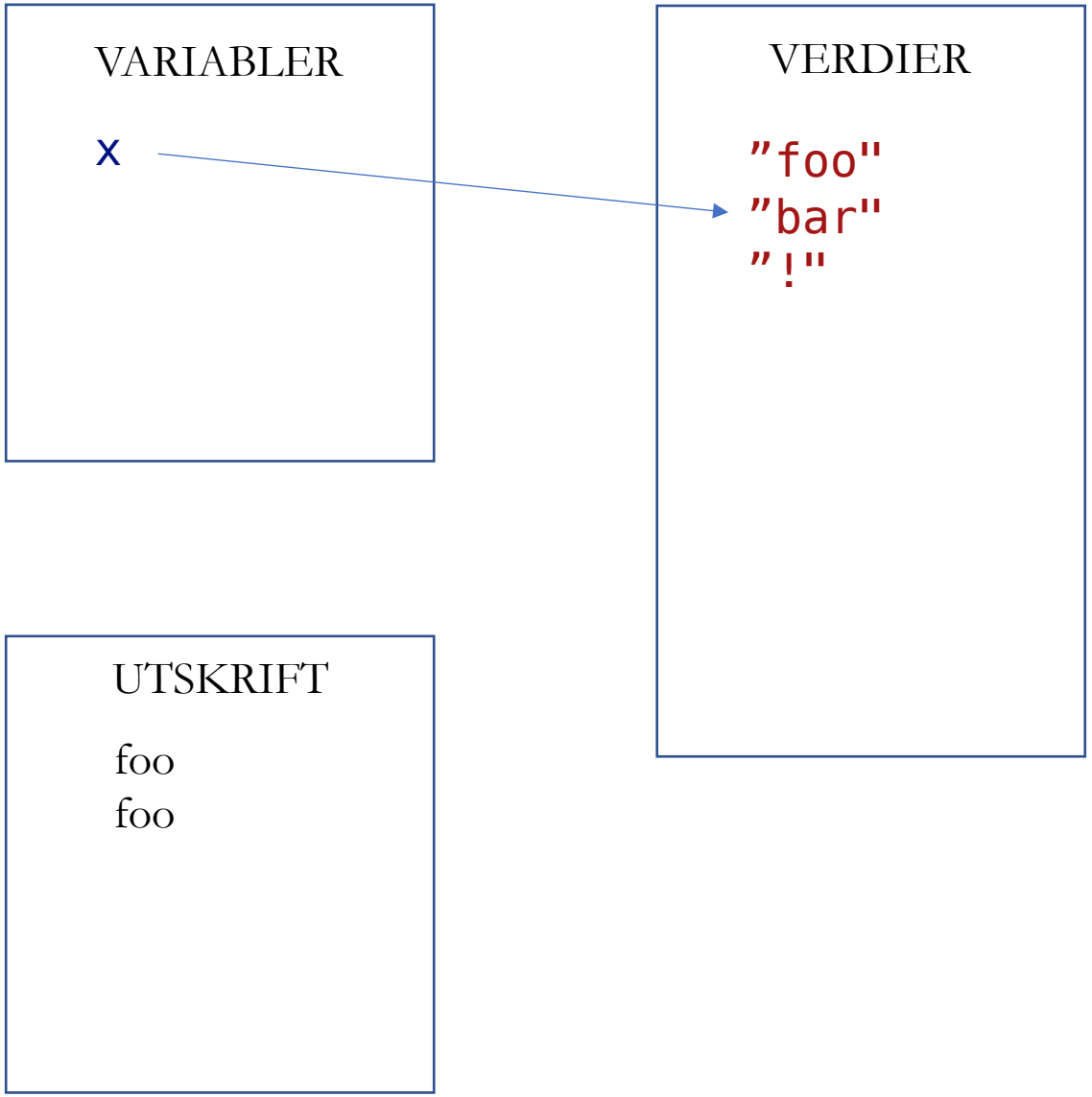
VERDIER

"foo"
"bar"
"!"

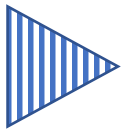
UTSKRIFT

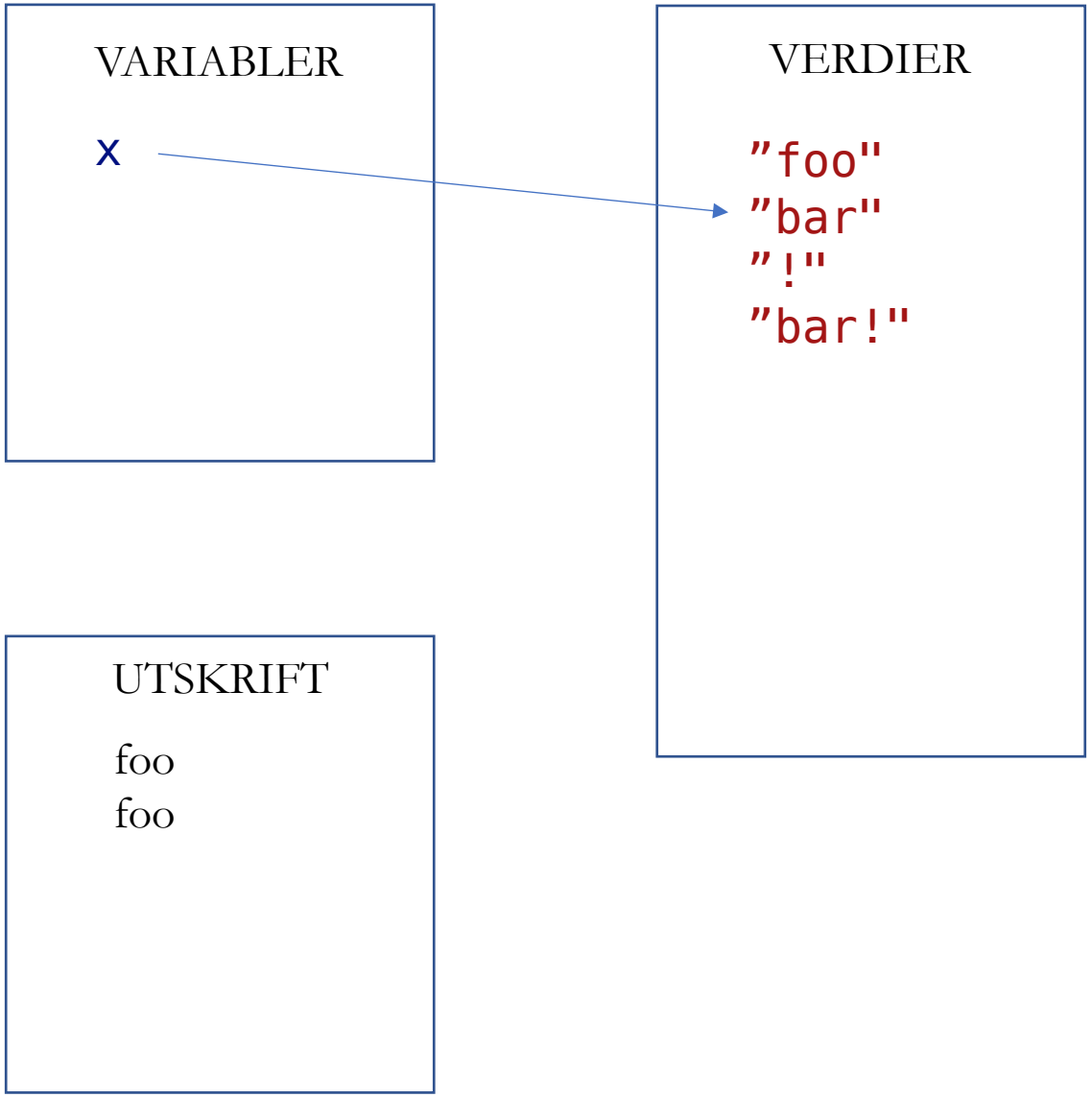
foo
foo

```
def print_twice(my_thing):
    print(my_thing)
    print(my_thing)

print_twice("foo")

x = "bar"
print_twice(x + "!")
```

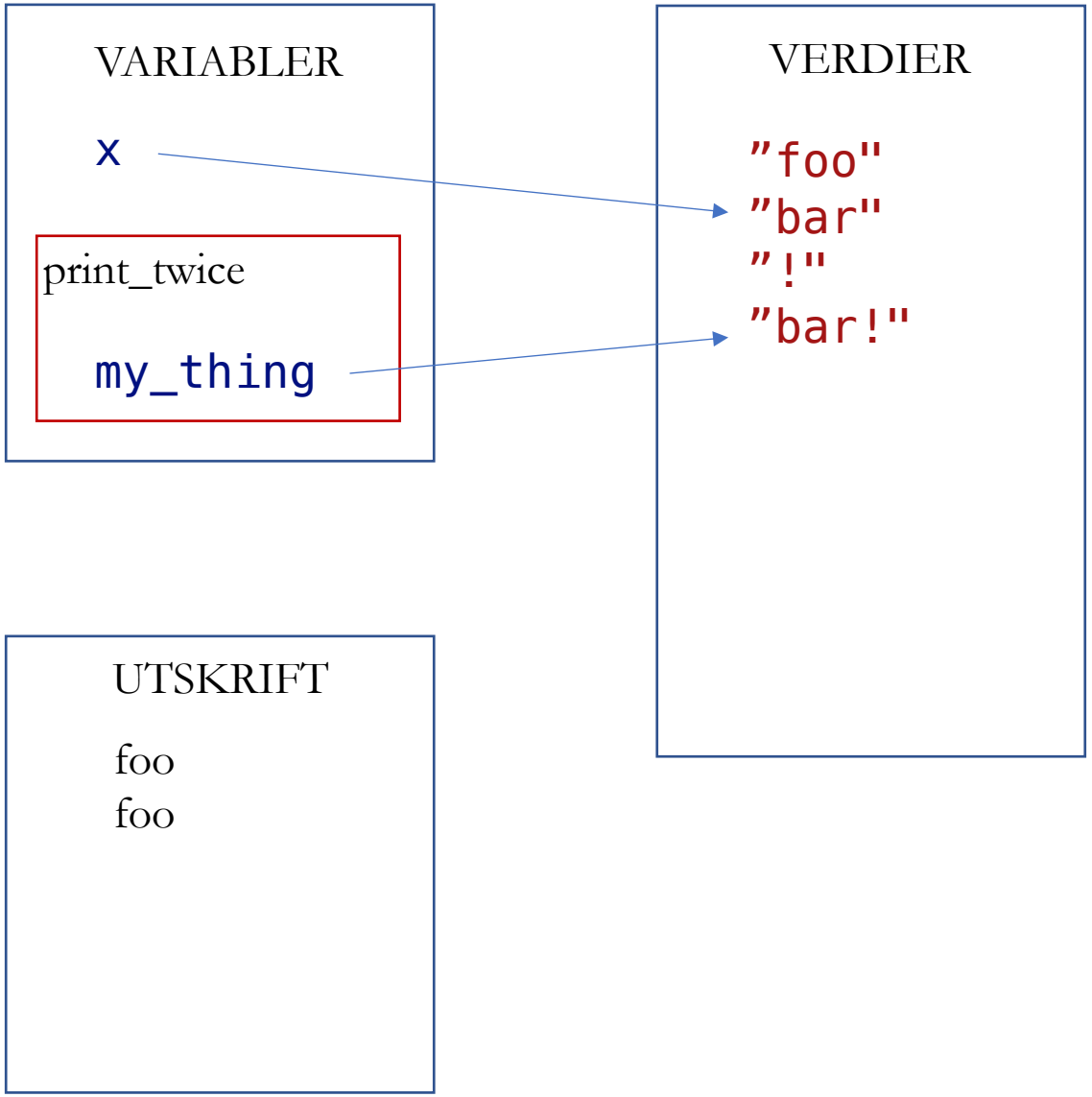


```
def print_twice(my_thing):
    print(my_thing)
    print(my_thing)

print_twice("foo")

x = "bar"
print_twice(x + "!")
```

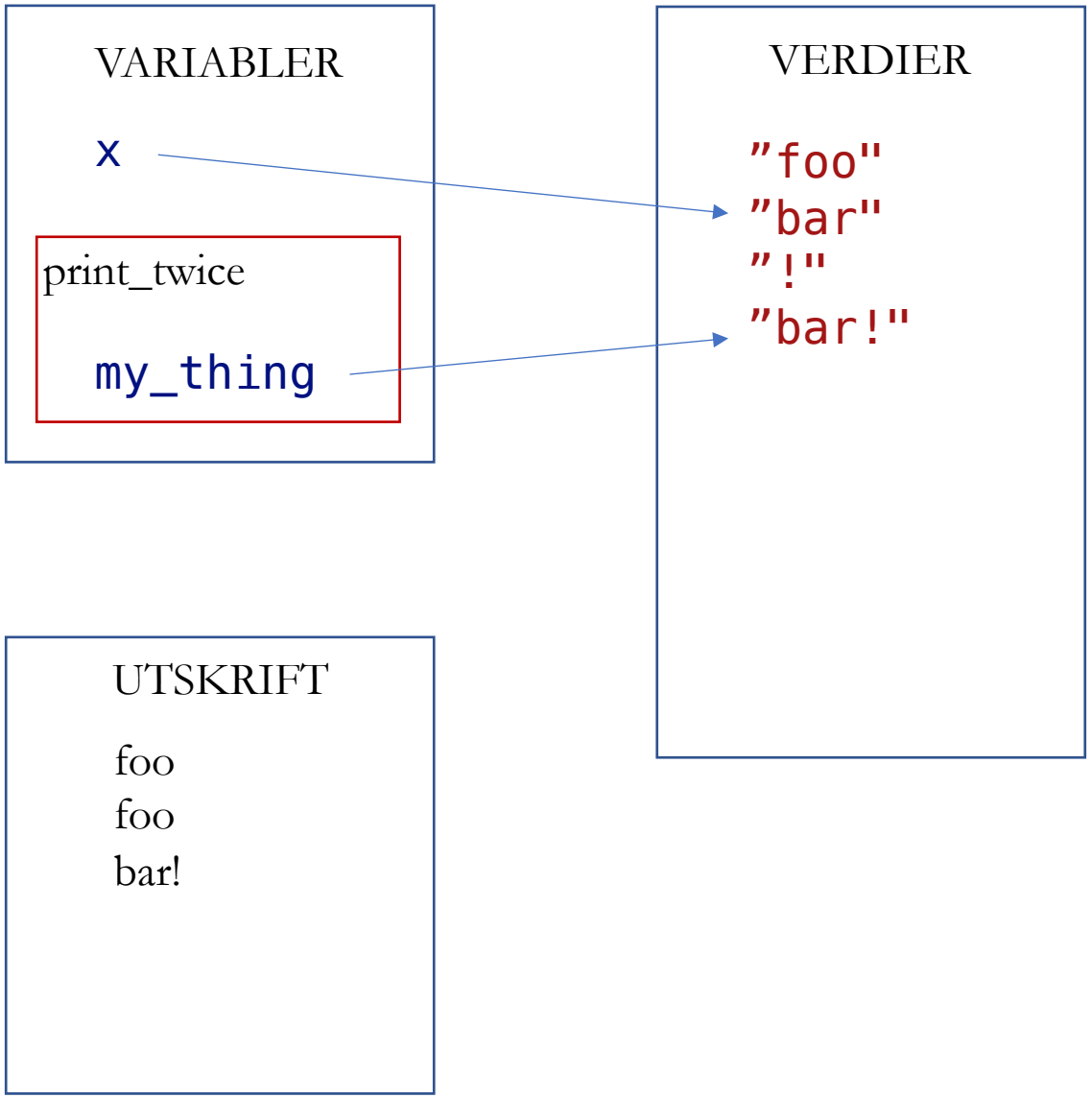




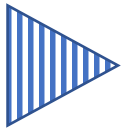
```
def print_twice(my_thing):
    print(my_thing)
    print(my_thing)

print_twice("foo")

x = "bar"
print_twice(x + "!")
```

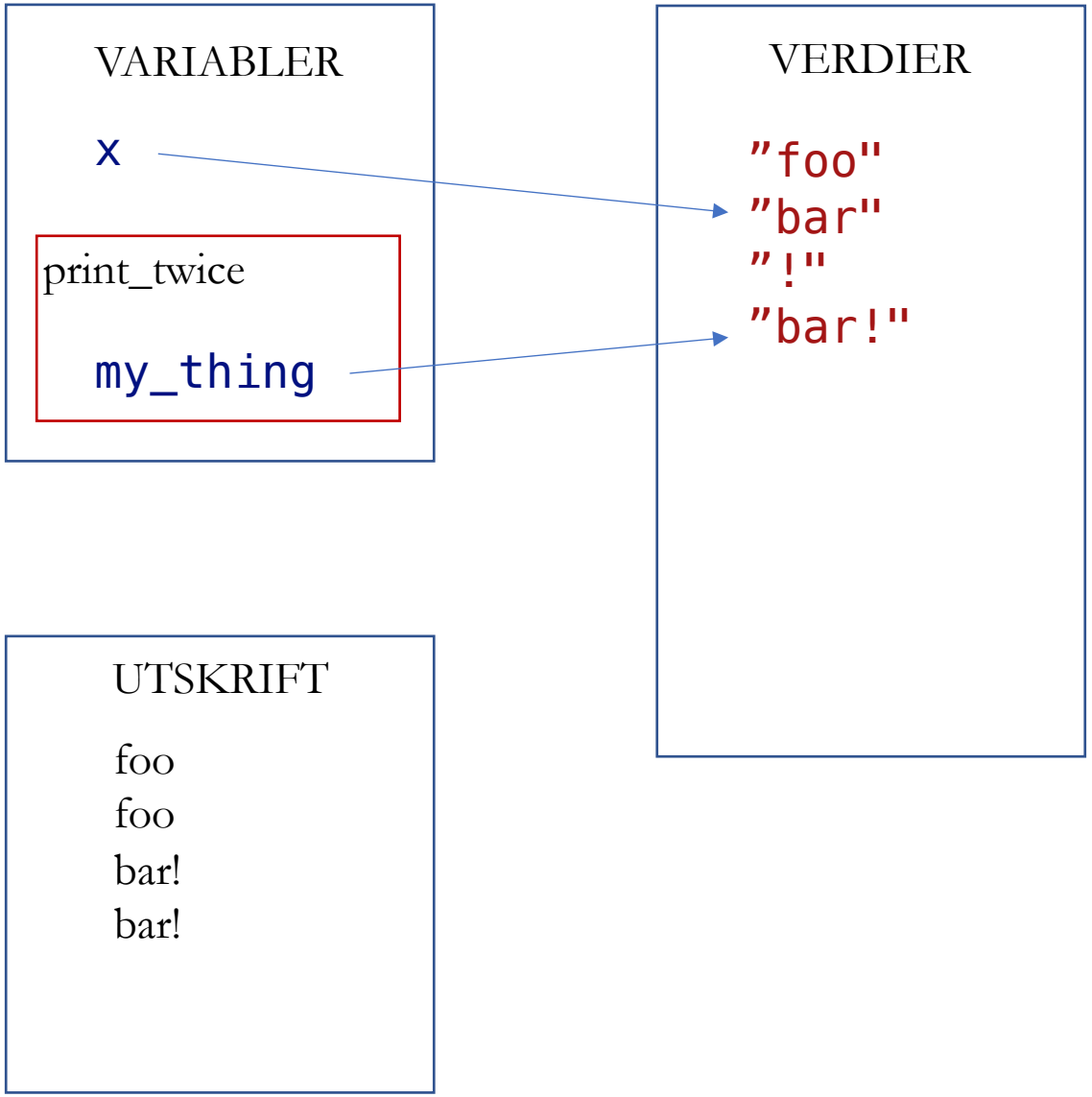


```
def print_twice(my_thing):
    print(my_thing)
    print(my_thing)

print_twice("foo")

x = "bar"
print_twice(x + "!")
```



```
def print_twice(my_thing):
    print(my_thing)
    print(my_thing)

print_twice("foo")

x = "bar"
print_twice(x + "!")
```





```
def print_twice(my_thing):
    print(my_thing)
    print(my_thing)

print_twice("foo")

x = "bar"
print_twice(x + "!")
```