

QUIZ2

- Problemløsning
- Problemløsning
- Forstå problemene
- Problemløsning

PROBLEMLØSNING

- Top-down design: løse “de store linjene” først
- Bottom-up: kjenne igjen små deler som kan løses separat

- Forstå problemet – forstå eksemplene
- Still spørsmål
- Samarbeid

- Øvelse, øvelse, øvelse

LAB2

- rotate_string(s, k)

rotate_string(s, k)

```
def rotate_string_by_one(s):  
    return s[1:] + s[0]
```

```
def rotate_string(s, k):  
    for i in range(k):  
        s = rotate_string_by_one(s)  
    return s
```

```
def rotate_string(s, k):  
    if k > len(s):  
        return rotate_string(s, k - len(s))  
    return s[k:] + s[:k]
```

```
def rotate_string(s, k):  
    while k > len(s):  
        k -= len(s)  
    return s[k:] + s[:k]
```

```
def rotate_string(s, k):  
    k %= len(s)  
    return s[k:] + s[:k]
```

I DAG

- Tupler
- Lister

- Mutasjon og alias

- Grafikk



UNIVERSITETET I BERGEN

LISTER

INF100

HØST 2022

Torstein Strømme

SAMLINGER

- Strenger, range, tupler og lister

```
s = "abc"  
r = range(3, 12, 2)  
t = (1, 2, "abc")  
a = [9, 8, "abc"]
```

Indeksering

```
s[0]  
r[1]  
t[1]  
a[2]
```

Beskjæring

```
s[:2]  
r[1:3]  
t[::-1]  
a[:2]
```

Løkker

```
for thing in s/r/t/a :  
    ...
```

Medlemskap

```
"a" in s  
5 in r  
"abc" in t  
8 in a
```

Funksjoner

```
min max  
len  
.count  
.index
```

SAMLINGER

- Strenger, range, tupler og lister

```
s = "abc"
```

```
r = range(3, 12, 2)
```

```
t = (1, 2, "abc")
```

```
a = [9, 8, "abc"]
```

Repetisjon

```
s * 0
```

```
t * 2
```

```
a * 3
```

Konkatenasjon

```
s + "def"
```

```
t + (True, 0.0)
```

```
a + ["a", -3]
```


Hvordan opprette tupler

```
# Tom tuple
```

```
t = ()
```

```
t = tuple()
```

```
# Tuple med ett element
```

```
t = (42,)
```

```
# Tuple med flere elementer
```

```
t = (42, 95, "abc",)
```

```
t = (42, 95)
```

```
# Konvertere andre samlinger til tuple
```

```
t = tuple("abc")
```

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

```
t = tuple(range(5))
```

Hvordan opprette lister

```
# Tom liste  
a = []  
a = list()
```

```
# Liste med ett element  
a = [42,]  
a = [42]
```

```
# Liste med flere elementer  
a = [42, 95, "abc",]  
a = [42, 95]
```

```
# Konvertere andre samlinger til liste  
a = list("abc")  
a = list((1, 2, 3))  
a = list(range(5))
```

```
# Splitting av en streng  
a = "1 2 5 9 4".split(" ")  
a = "1\n5 9\n4".splitlines()
```

```
for line in file_contents.splitlines():  
    ...
```

en liste



PROBLEMLØSNING MED LISTER

<https://open.kattis.com/contests/jmzeca>

LICENSE TO LAUNCH

<https://open.kattis.com/problems/licensetolaunch>

- Input
 - Første linje: et tall n
 - Andre linje: en streng med n tall adskilt med mellomrom (i.e. en liste med n tall)
- Output
 - Posisjonen til (det første) tallet med lavest verdi

ODD GNOME

<https://open.kattis.com/problems/oddgnome>

- Input
 - Første linje: et tall n , antall testcaser
 - I hver testcase: en liste med tall
- Output
 - For hver testcase: posisjonen til første tall som ikke er akkurat én mer enn forrige

BUBBLE TEA

<https://open.kattis.com/problems/bubbletea>

- Input
 - En liste med priser for boble-te
 - En liste med priser for toppinger
 - For hver boble-te, en liste med hvilke toppinger som kan passe
 - Et tall, hvor stort budsjettet er
- Output
 - Et tall, hvor mange boble-te man har råd til å kjøpe (-1, siden vi drikker en selv)

LISTER vs. TUPLER

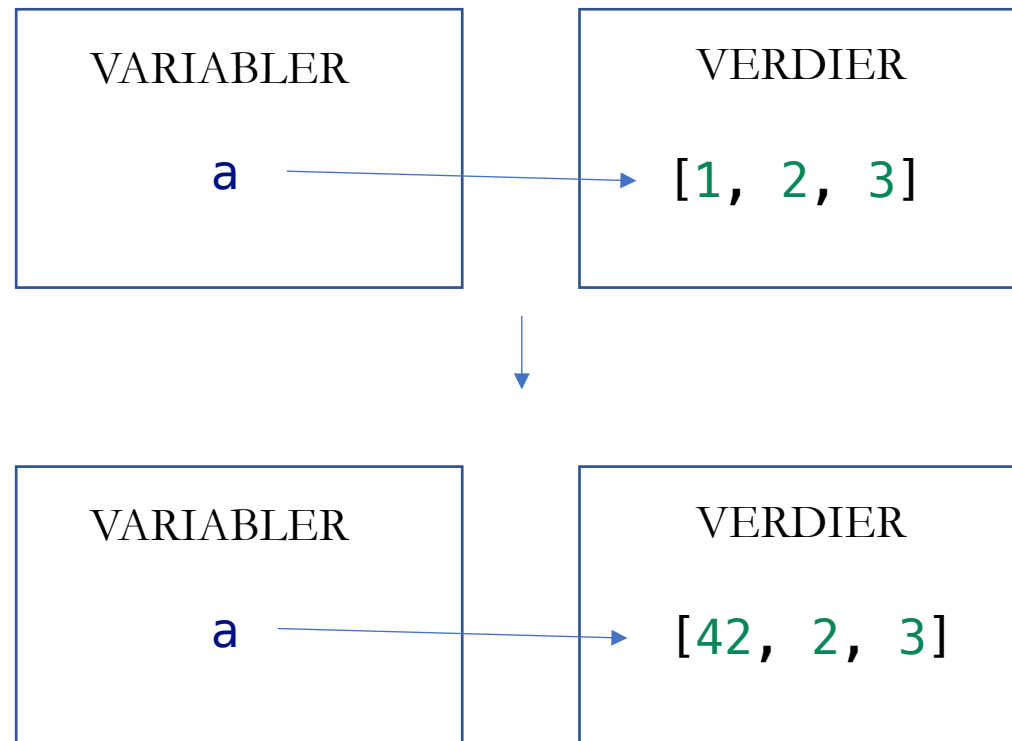
- En liste kan *muteres*
- En tupel kan *ikke* muteres

- Tidligere: *variabler* har endret seg
- Mutasjon: selve *verdien* endrer seg

MUTASJON

$a = [1, 2, 3]$

▶ $a[0] = 42$



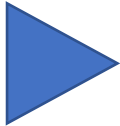
MUTASJON

```
t = (1, 2, 3)
```

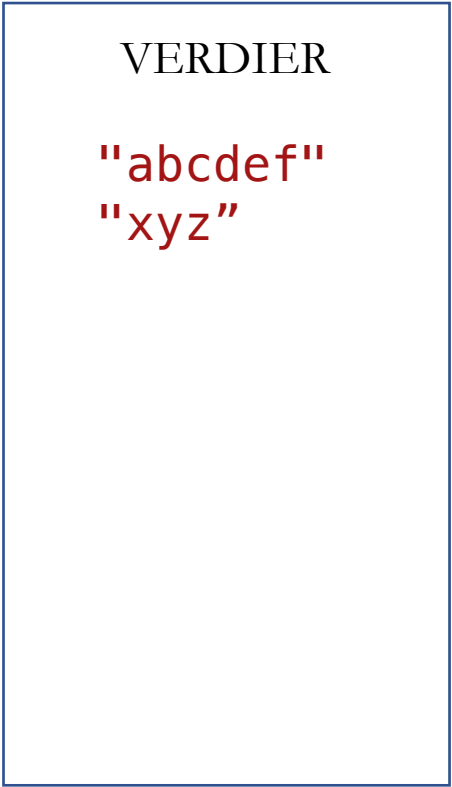
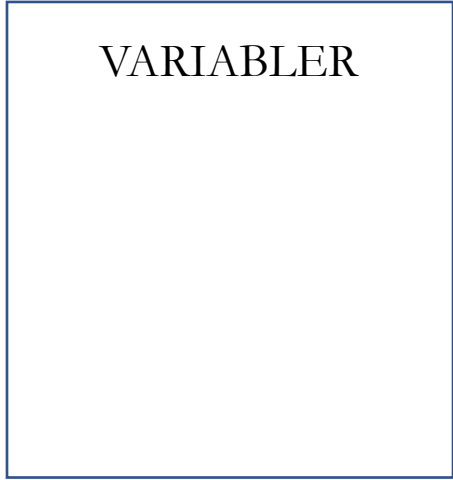
```
t[0] = 42
```

TypeError: 'tuple' object does not support item assignment

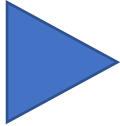
STRENG



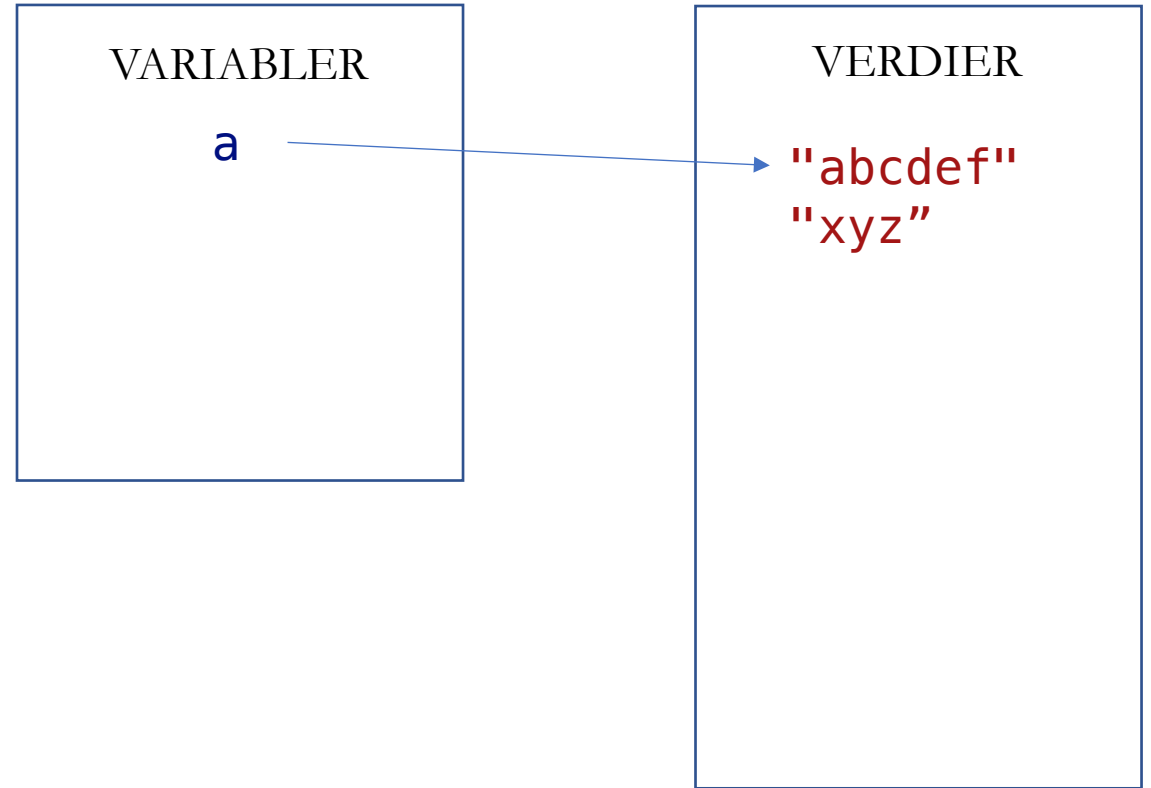
```
a = "abcdef"  
b = "xyz"  
a += b
```



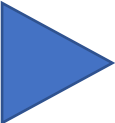
STRENG



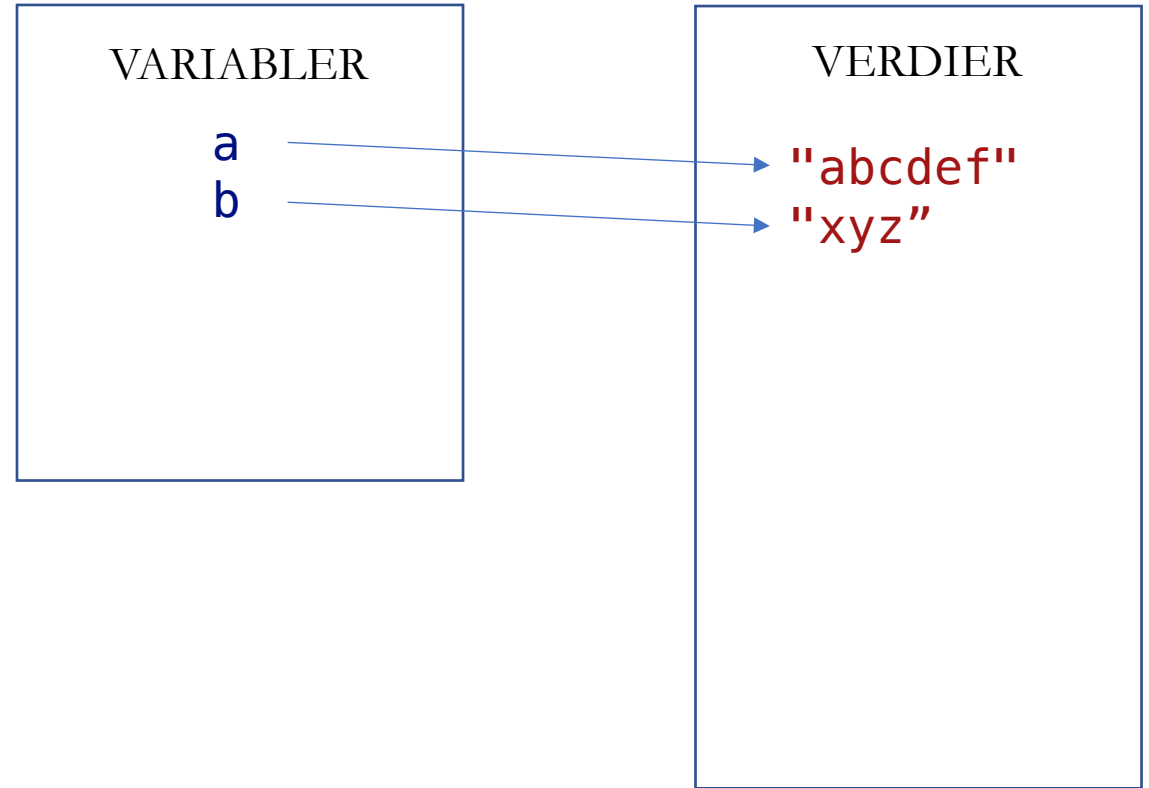
```
a = "abcdef"  
b = "xyz"  
a += b
```



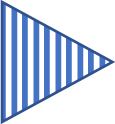
STRENG



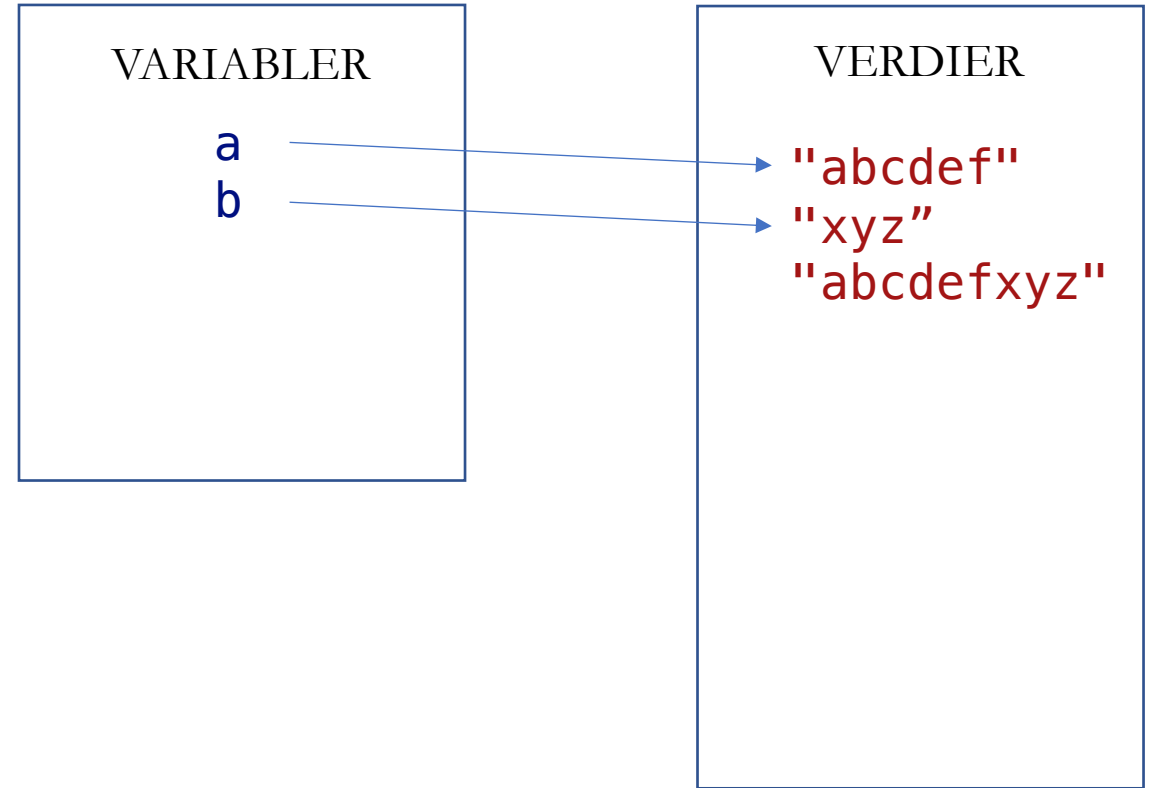
```
a = "abcdef"  
b = "xyz"  
a += b
```




STRENG



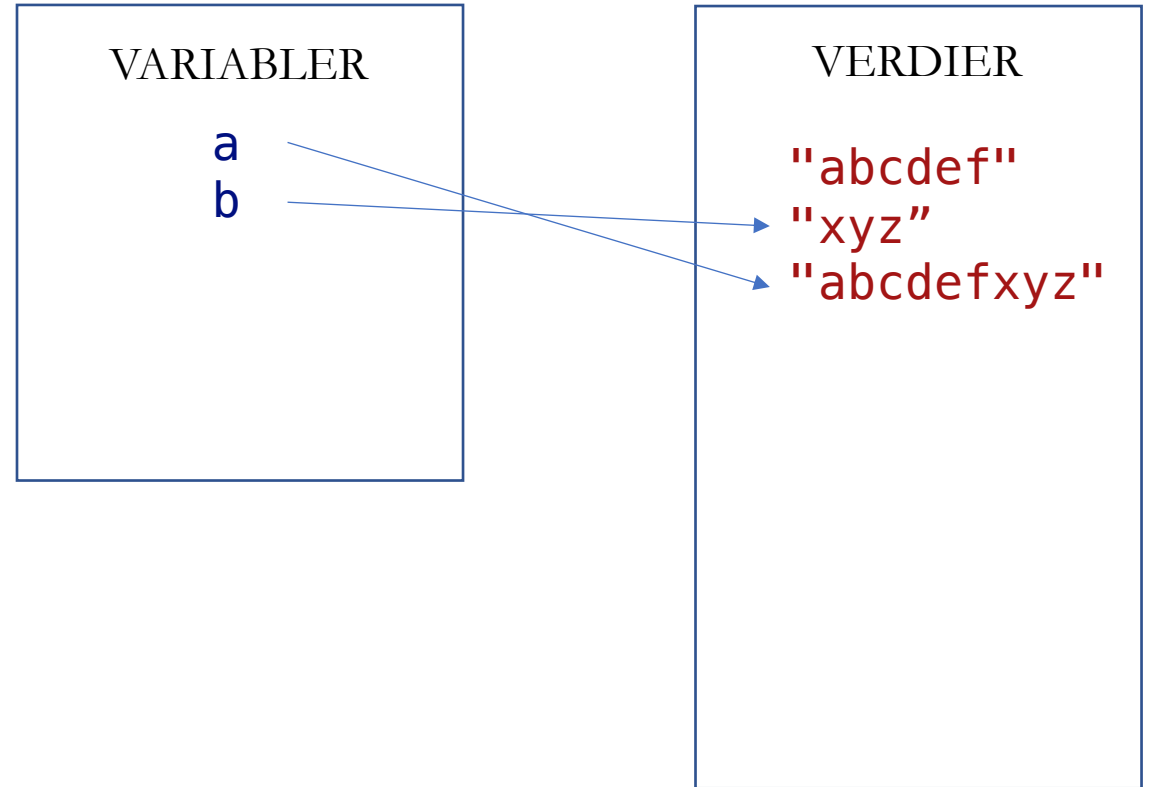
```
a = "abcdef"  
b = "xyz"  
a += b
```



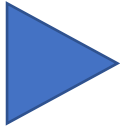
STRENG



```
a = "abcdef"  
b = "xyz"  
a += b
```



TUPLE



```
a = (2, 3, "abc")  
b = (True, 0.0)  
a += b
```

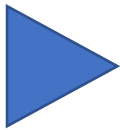
VARIABLER

VERDIER

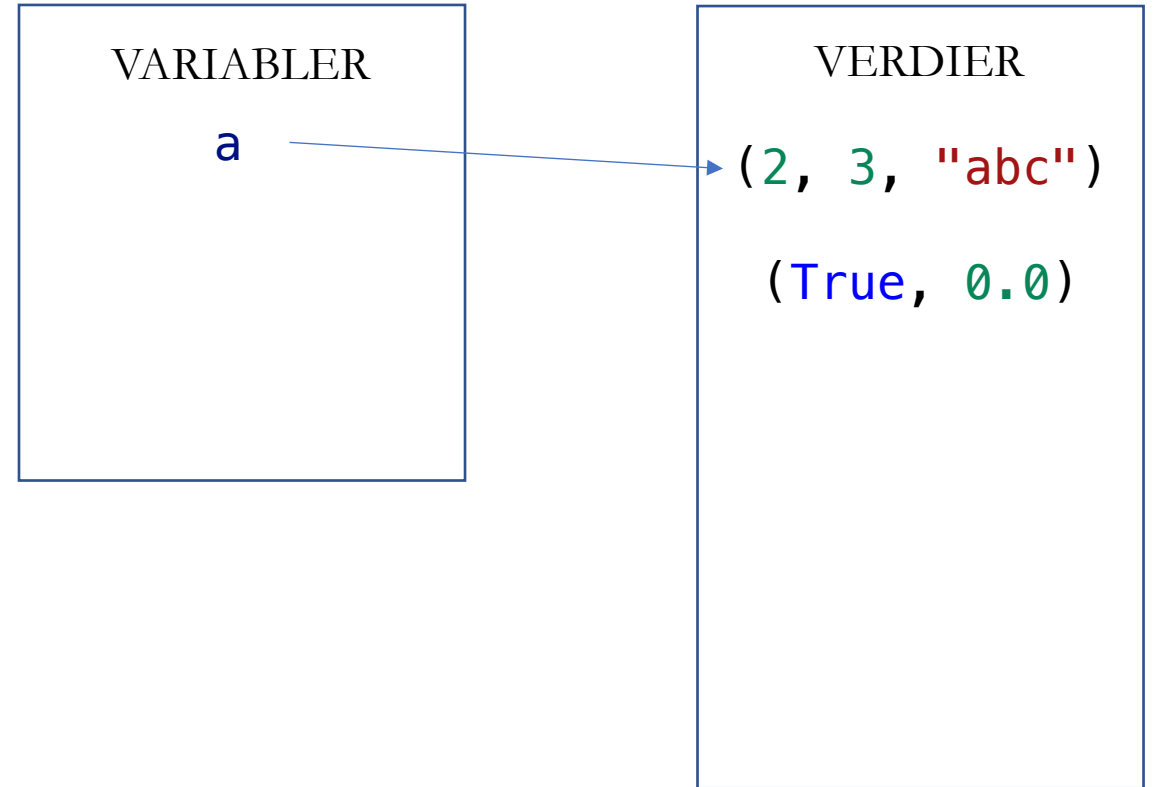
(2, 3, "abc")

(True, 0.0)


TUPLE



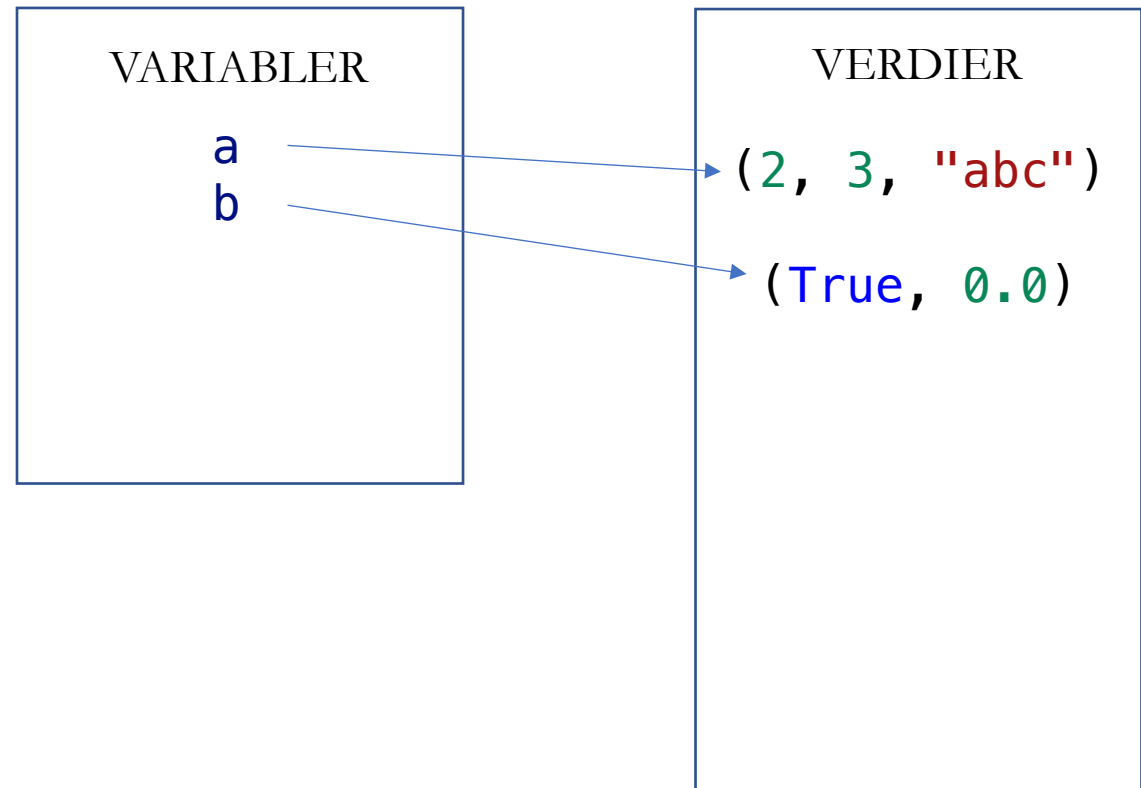
```
a = (2, 3, "abc")  
b = (True, 0.0)  
a += b
```



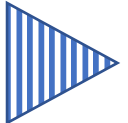
TUPLE



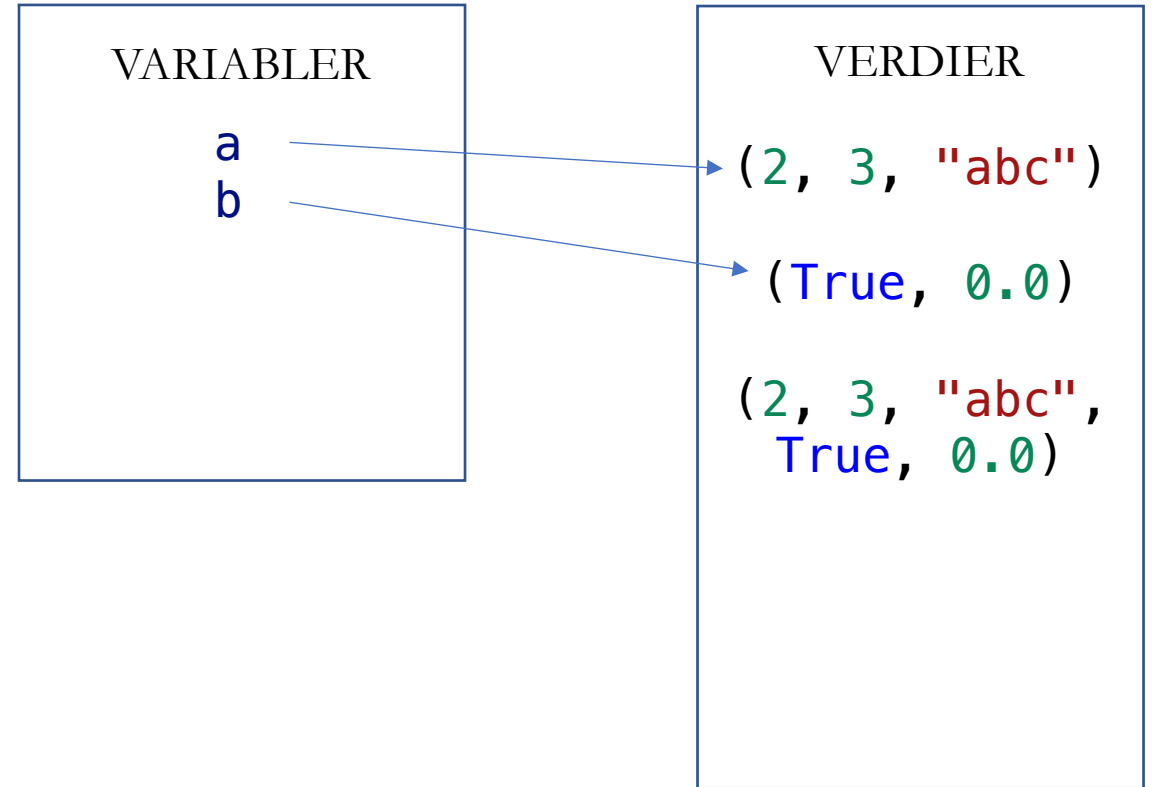
```
a = (2, 3, "abc")  
b = (True, 0.0)  
a += b
```



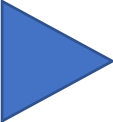
TUPLE



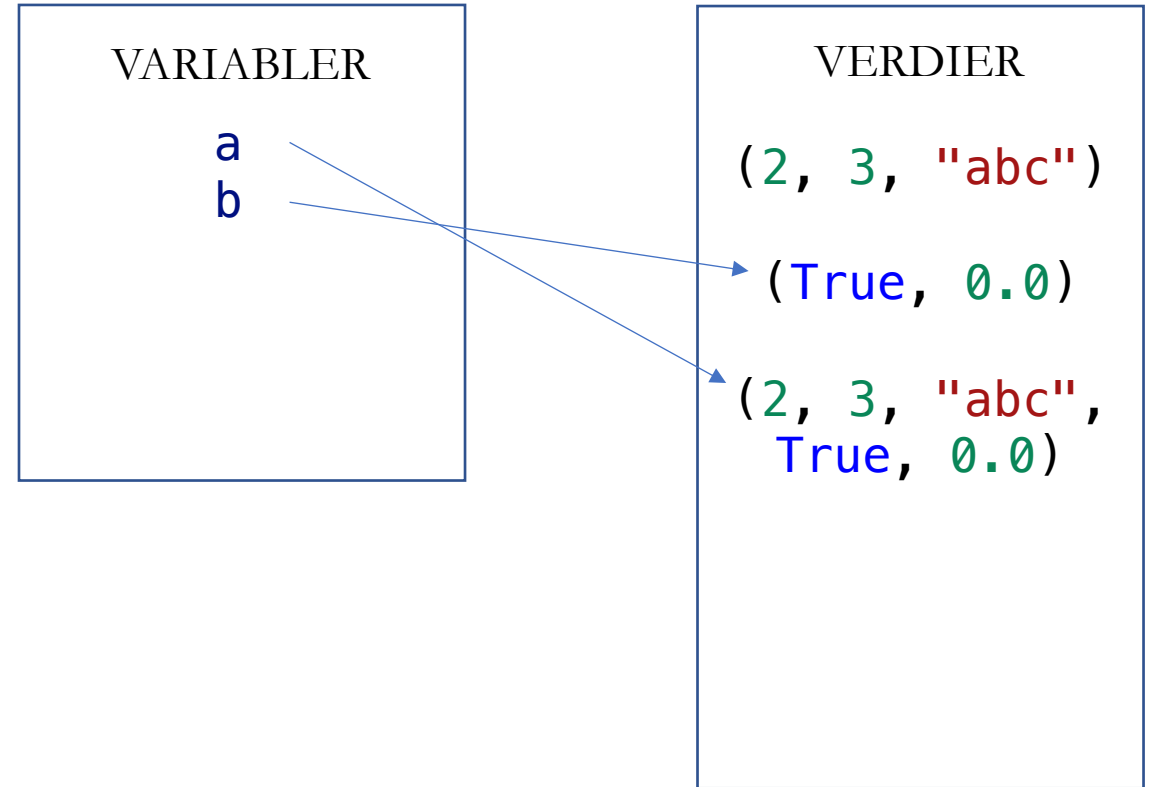
```
a = (2, 3, "abc")  
b = (True, 0.0)  
a += b
```



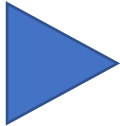
TUPLE



```
a = (2, 3, "abc")  
b = (True, 0.0)  
a += b
```



LISTE



```
a = [2, 3, "abc"]  
b = [True, 0.0]  
a += b
```

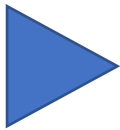
VARIABLER

VERDIER

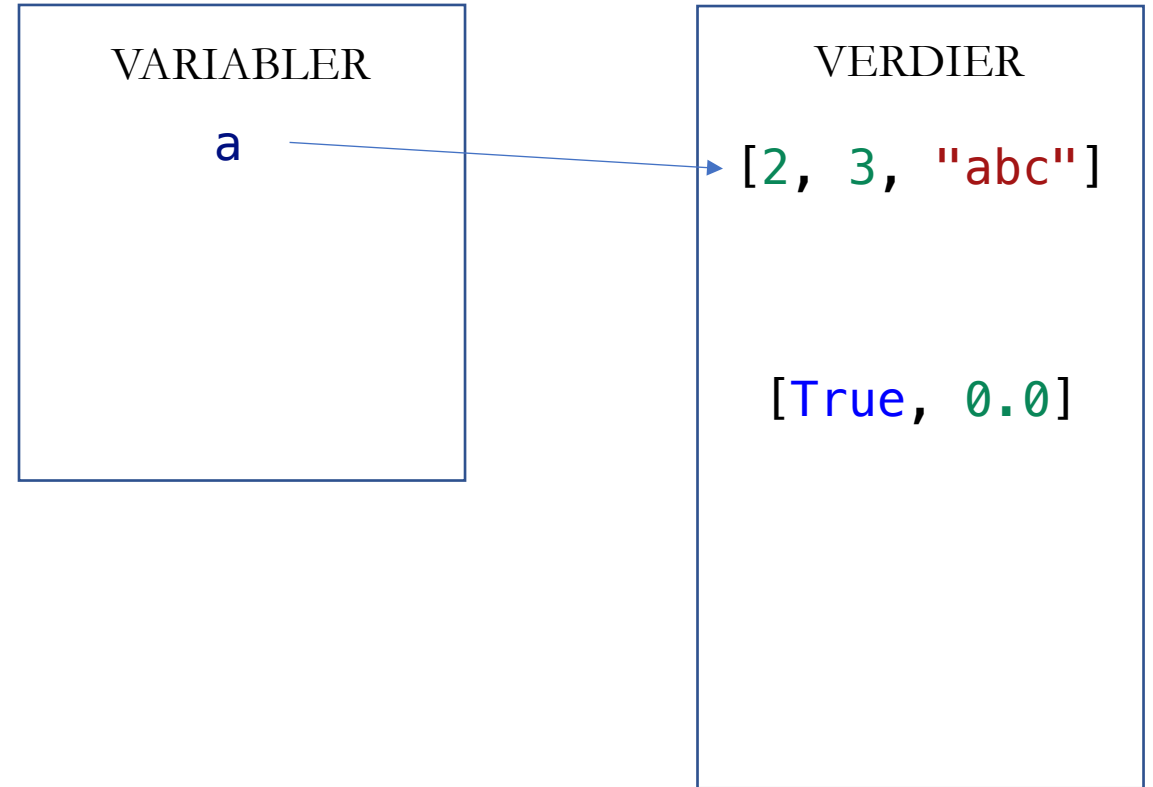
[2, 3, "abc"]

[True, 0.0]

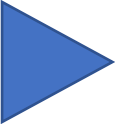
LISTE



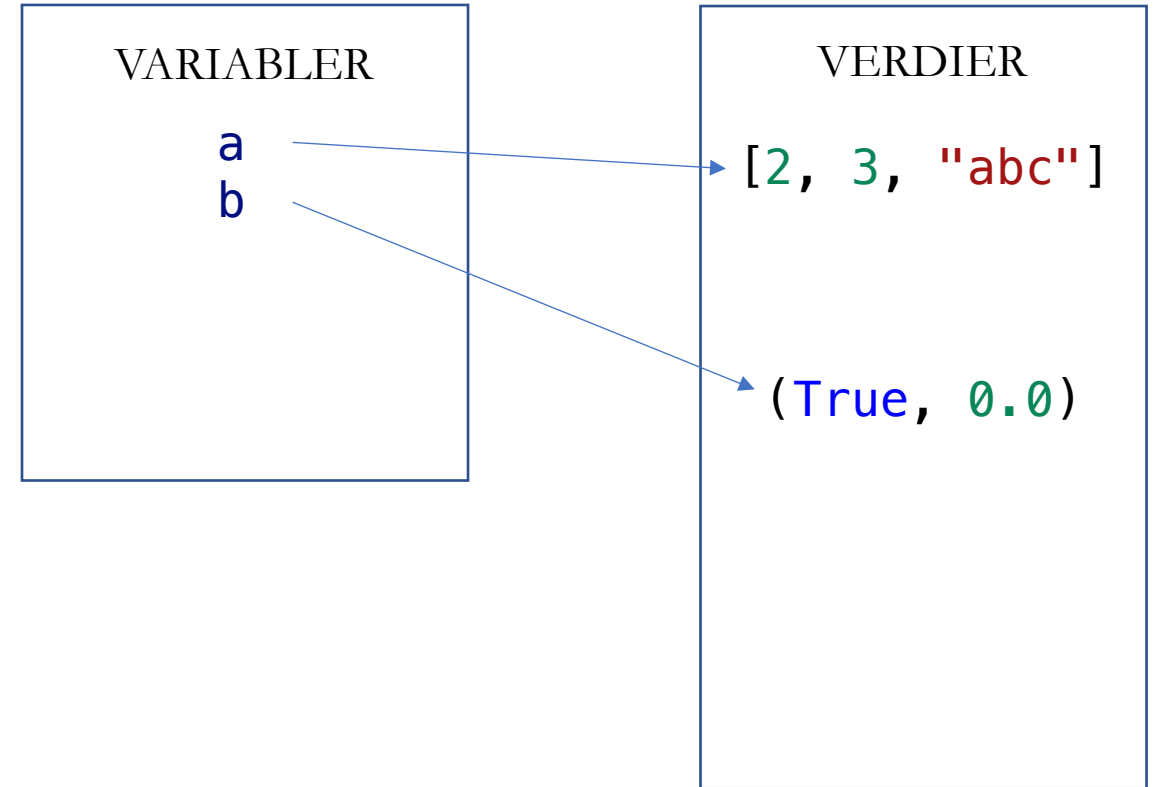
```
a = [2, 3, "abc"]  
b = [True, 0.0]  
a += b
```



LISTE

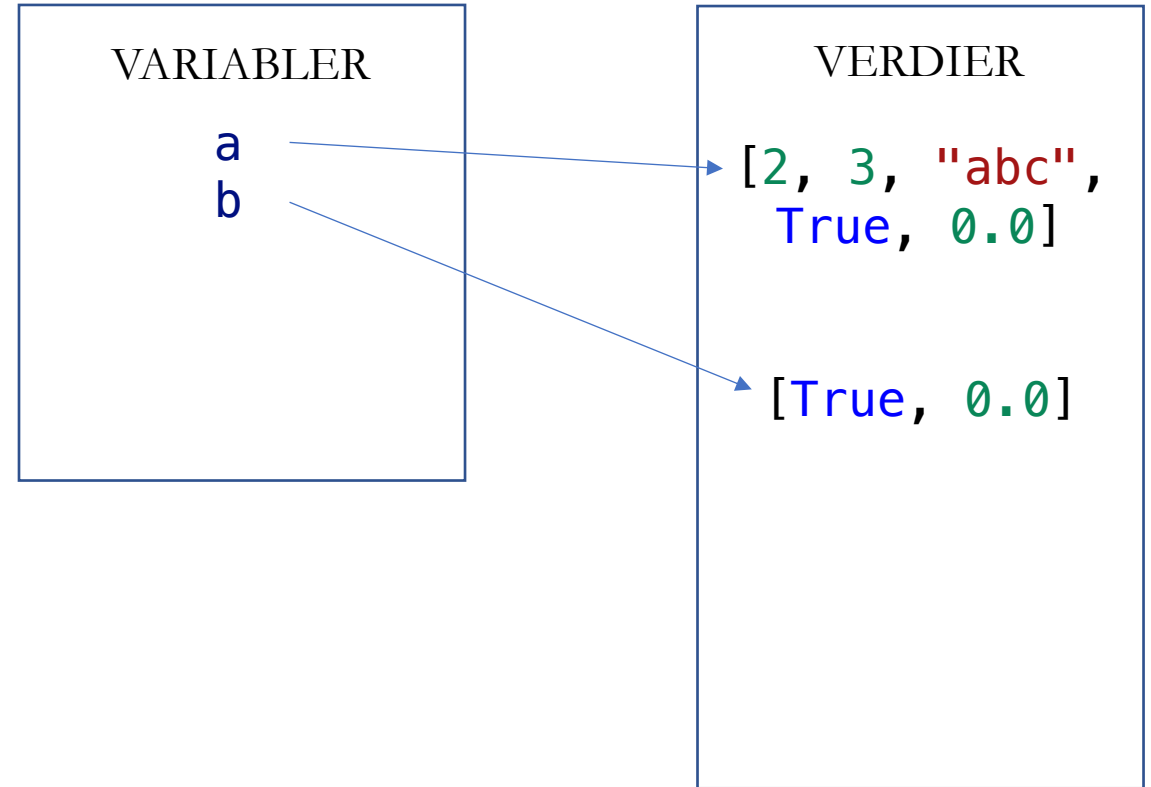


```
a = [2, 3, "abc"]  
b = [True, 0.0]  
a += b
```



LISTE

a = [2, 3, "abc"]
b = [True, 0.0]
a += b



MUTATIVE OPERASJONER PÅ LISTER

- Operasjoner som *muterer*

```
a = [1, 2, 3]
```

```
a[i] = 42
```

```
a += [4, 5]
```

```
a *= 2
```

- Operasjoner som oppretter ny verdi

```
a = [1, 2, 3]
```

```
a = a[:i] + [42] + a[i+1:]
```

```
a = a + [4, 5]
```

```
a = a * 2
```


DESTRUKTIVE FUNKSJONER

- En destruktiv funksjon har en sideeffekt: den muterer en verdi
- En ikke-destruktiv funksjon muterer ingen verdier
- Destruktive funksjoner trenger ikke returverdi
- Ikke-destruktive funksjoner benytter returverdi

FUNKSJONER PÅ LISTER

Hva?	Destruktiv funksjon	Ikke-destruktivt alternativ
Legge til en ny verdi på slutten av listen	<code>a.append(42)</code>	<code>a = a + [42]</code>
Utvid listen med flere nye elementer på en gang	<code>a.extend([3, 4])</code>	<code>a = a + [3, 4]</code>
Putte inn en verdi på gitt posisjon	<code>a.insert(3, "foo")</code>	<code>a = a[:3] + ["foo"] + a[3:]</code>
Fjerne første forekomst av gitt verdi	<code>a.remove(2)</code>	<code>i = a.index(2)</code> <code>a = a[:i] + a[i + 1:]</code>
Fjerne element på en gitt posisjon	<code>a.pop(5)</code>	<code>a = a[:5] + a[5 + 1:]</code>
Fjerne alle elementer	<code>a.clear()</code>	<code>a = []</code>
Reverser	<code>a.reverse()</code>	<code>a = a[::-1]</code>
Sorter	<code>a.sort()</code>	<code>a = sorted(a)</code>

ALIAS

+ mutasjon



```
a = [2, 3, 4]
```

```
# Oppretter et alias  
b = a
```

```
# Mutasjon av listen  
a[0] = 99  
b[1] = 42  
print(a)  
print(b)
```

VARIABLER

UTSKRIFT

VERDIER

[2, 3, 4]

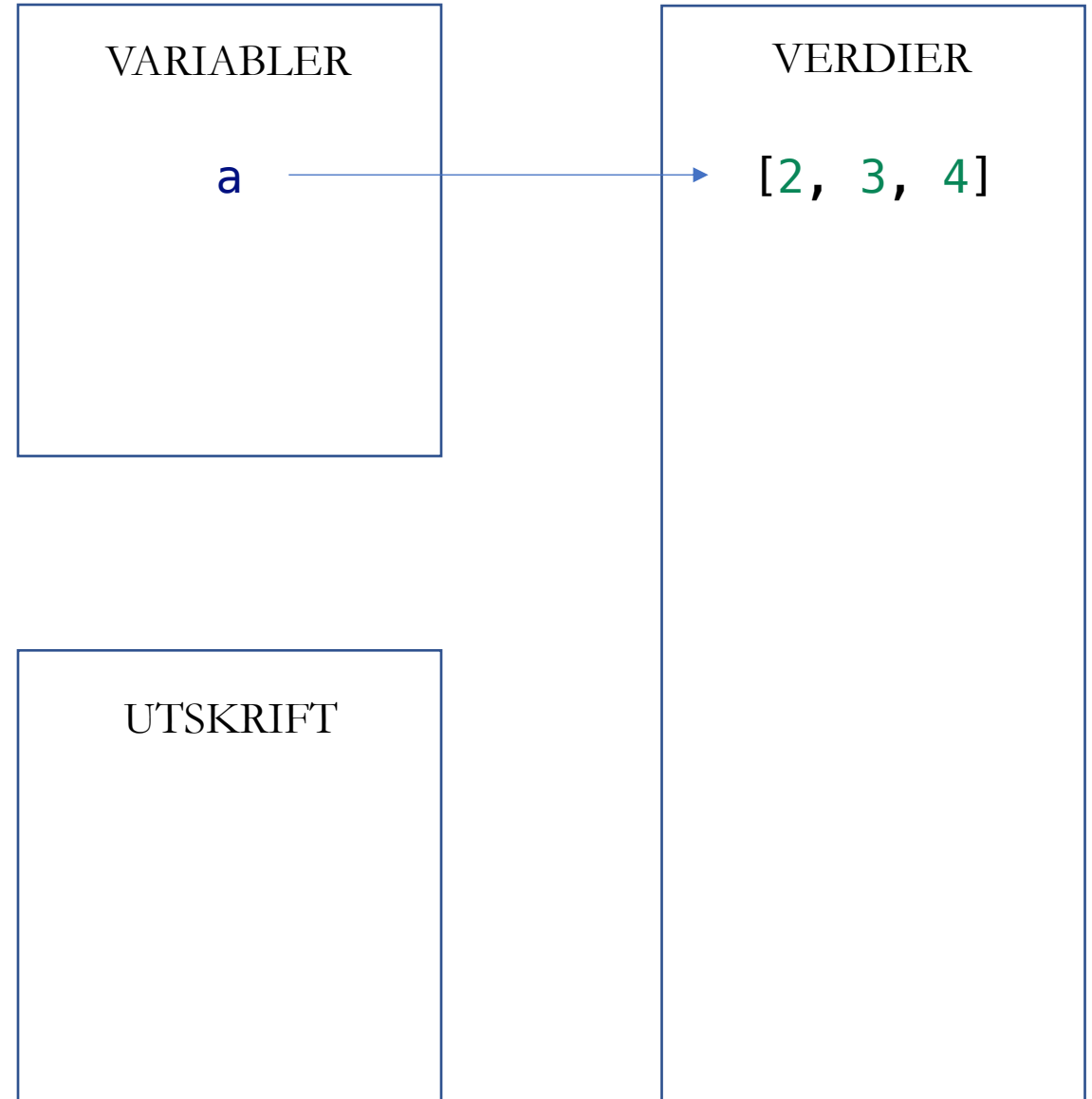
ALIAS

+ mutasjon

```
a = [2, 3, 4]
```

▶ # Oppretter et alias
b = a

```
# Mutasjon av listen  
a[0] = 99  
b[1] = 42  
print(a)  
print(b)
```



ALIAS

+ mutasjon

```
a = [2, 3, 4]
```

```
# Oppretter et alias
```

```
b = a
```

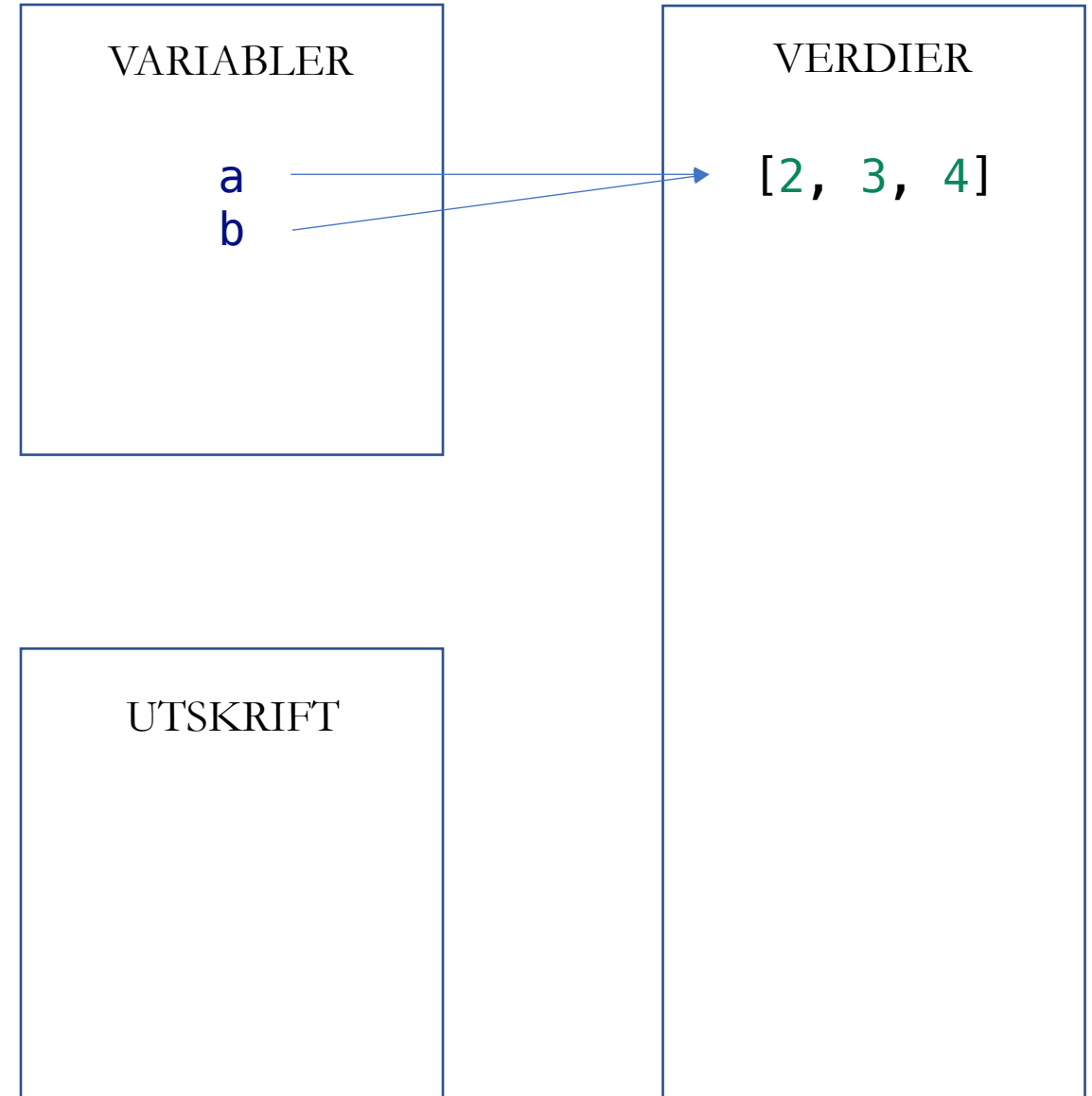
```
# Mutasjon av listen
```

```
a[0] = 99
```

```
b[1] = 42
```

```
print(a)
```

```
print(b)
```



ALIAS

+ mutasjon

```
a = [2, 3, 4]
```

```
# Oppretter et alias
```

```
b = a
```

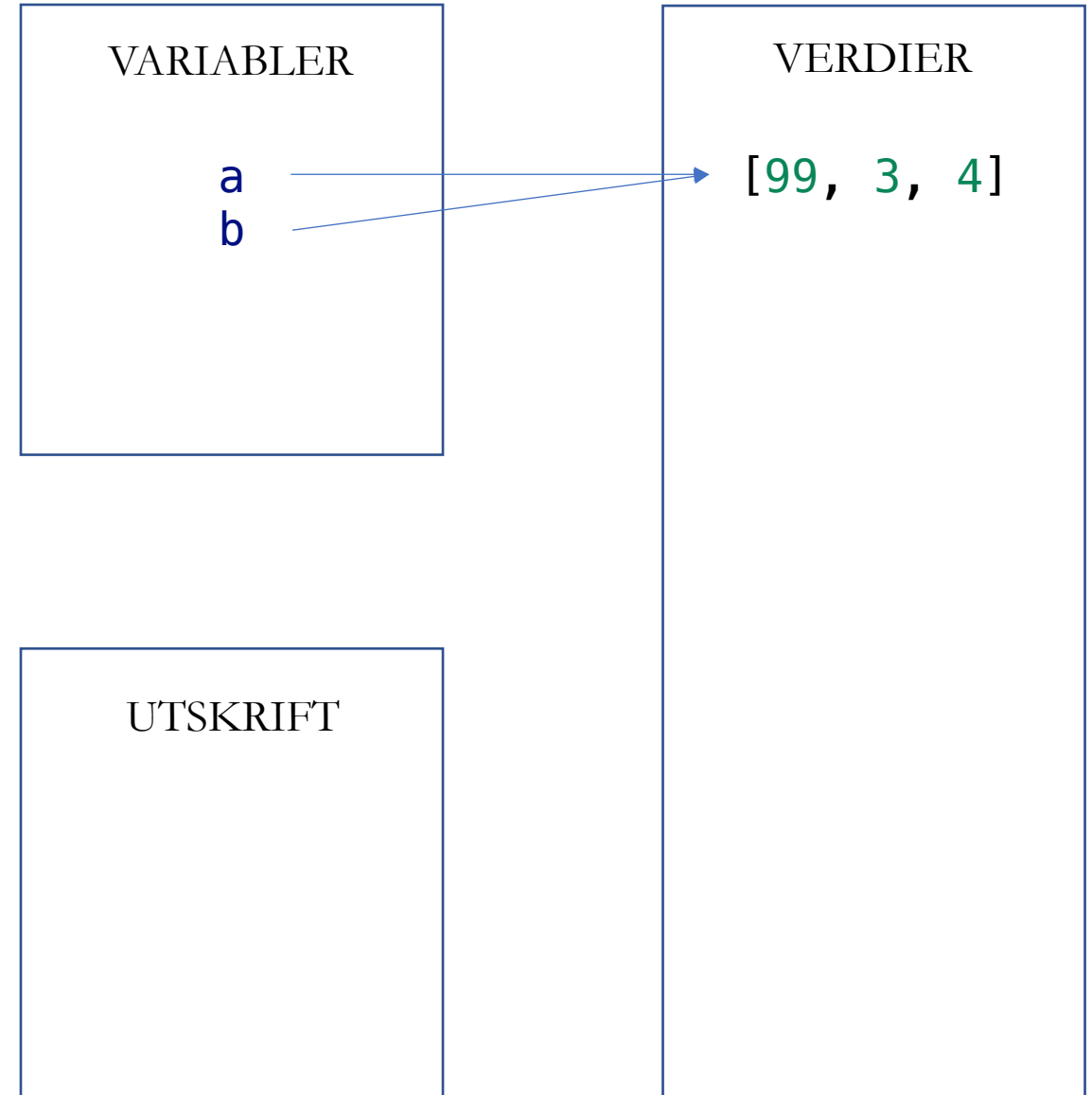
```
# Mutasjon av listen
```

```
a[0] = 99
```

```
b[1] = 42
```

```
print(a)
```

```
print(b)
```



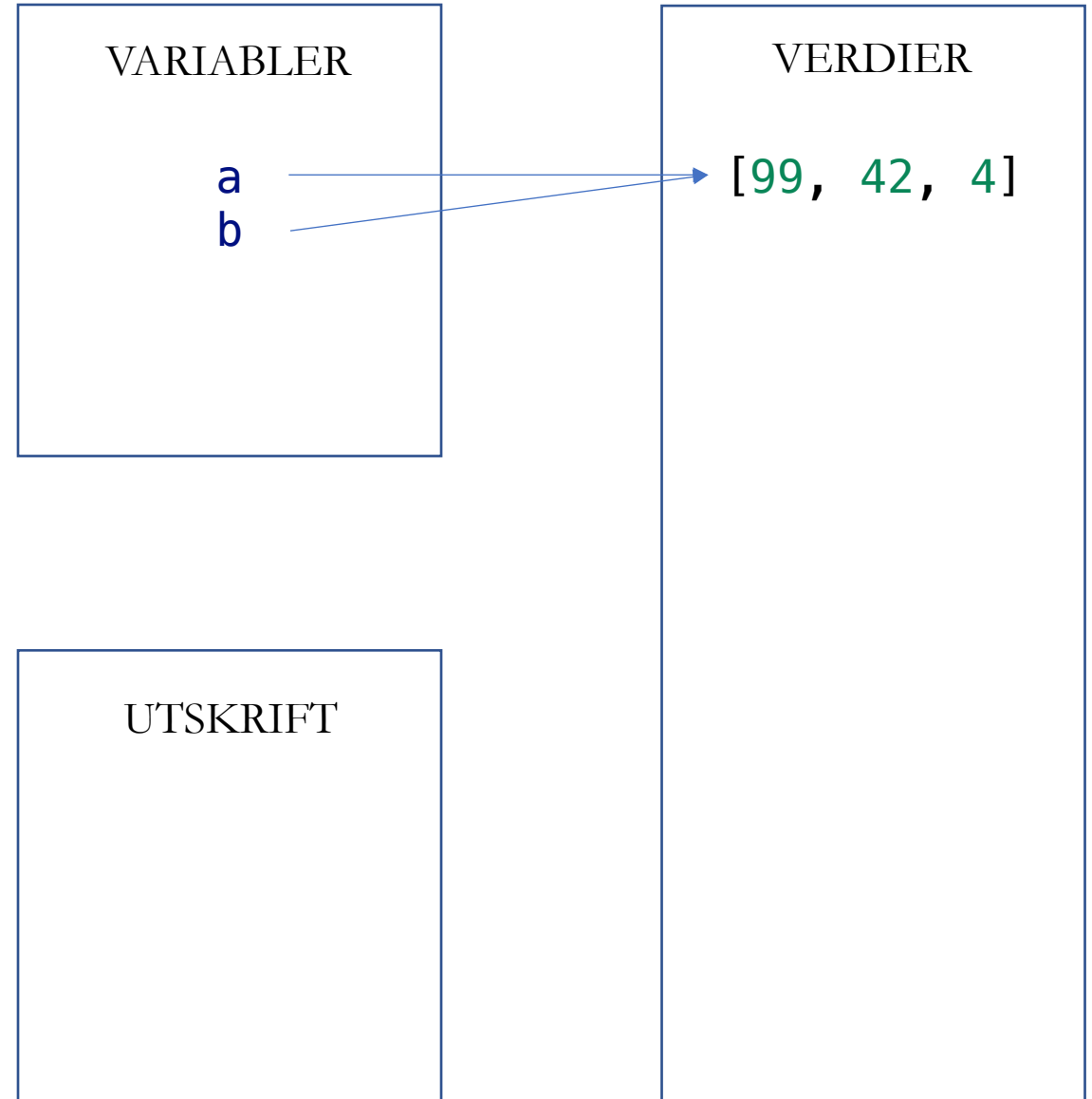
ALIAS

+ mutasjon

```
a = [2, 3, 4]
```

```
# Oppretter et alias  
b = a
```

```
# Mutasjon av listen  
a[0] = 99  
b[1] = 42  
print(a)  
print(b)
```



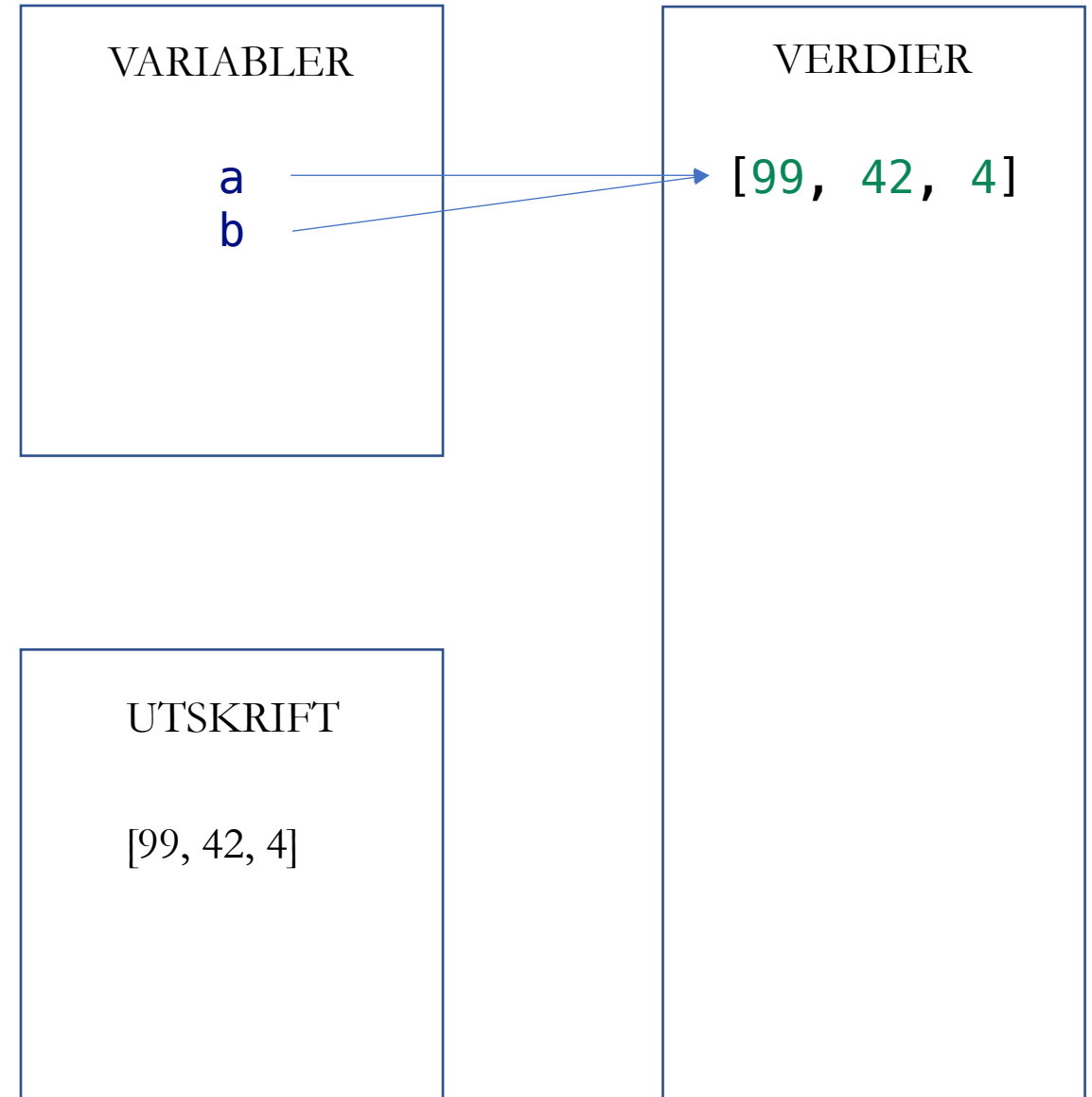
ALIAS

+ mutasjon

```
a = [2, 3, 4]
```

```
# Oppretter et alias  
b = a
```

```
# Mutasjon av listen  
a[0] = 99  
b[1] = 42  
print(a)  
print(b)
```



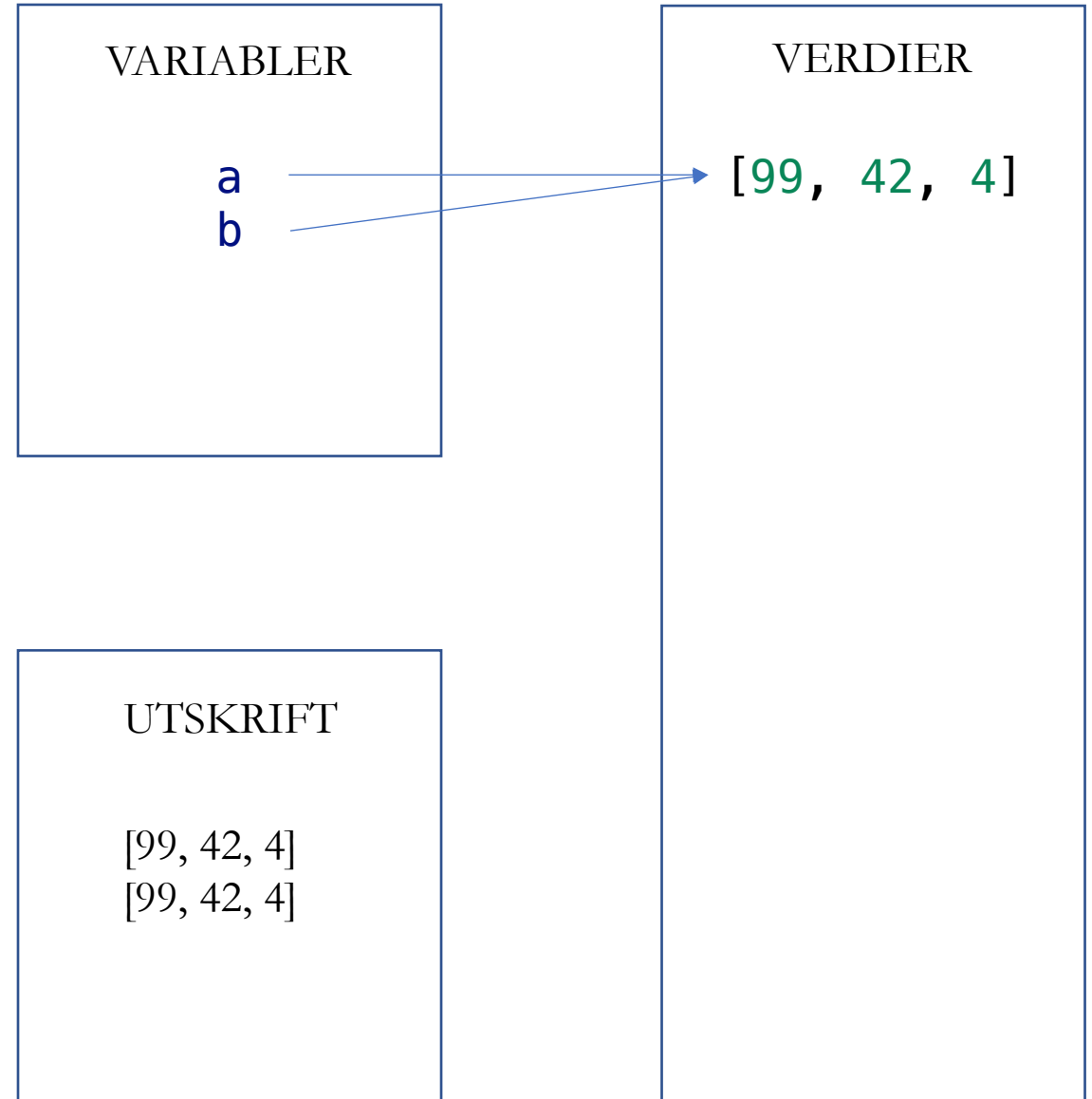
ALIAS

+ mutasjon

```
a = [2, 3, 4]
```

```
# Oppretter et alias  
b = a
```

```
# Mutasjon av listen  
a[0] = 99  
b[1] = 42  
print(a)  
print(b)
```



ALIAS

+ ikke-destruktive endringer



```
a = [2, 3, 4]
```

```
# Oppretter et alias
```

```
b = a
```

```
# Ikke-destruktiv endring
```

```
a = [99] + a[1:]
```

```
b = b[:1] + [42] + b[2:]
```

```
print(a)
```

```
print(b)
```

VARIABLER

VERDIER

[2, 3, 4]

UTSKRIFT

ALIAS

+ ikke-destruktive endringer

```
a = [2, 3, 4]
```

```
# Oppretter et alias
```

```
b = a
```

```
# Ikke-destruktiv endring
```

```
a = [99] + a[1:]
```

```
b = b[:1] + [42] + b[2:]
```

```
print(a)
```

```
print(b)
```

VARIABLER

a

VERDIER

[2, 3, 4]

UTSKRIFT

ALIAS

+ ikke-destruktive endringer

```
a = [2, 3, 4]
```

```
# Oppretter et alias
```

```
b = a
```

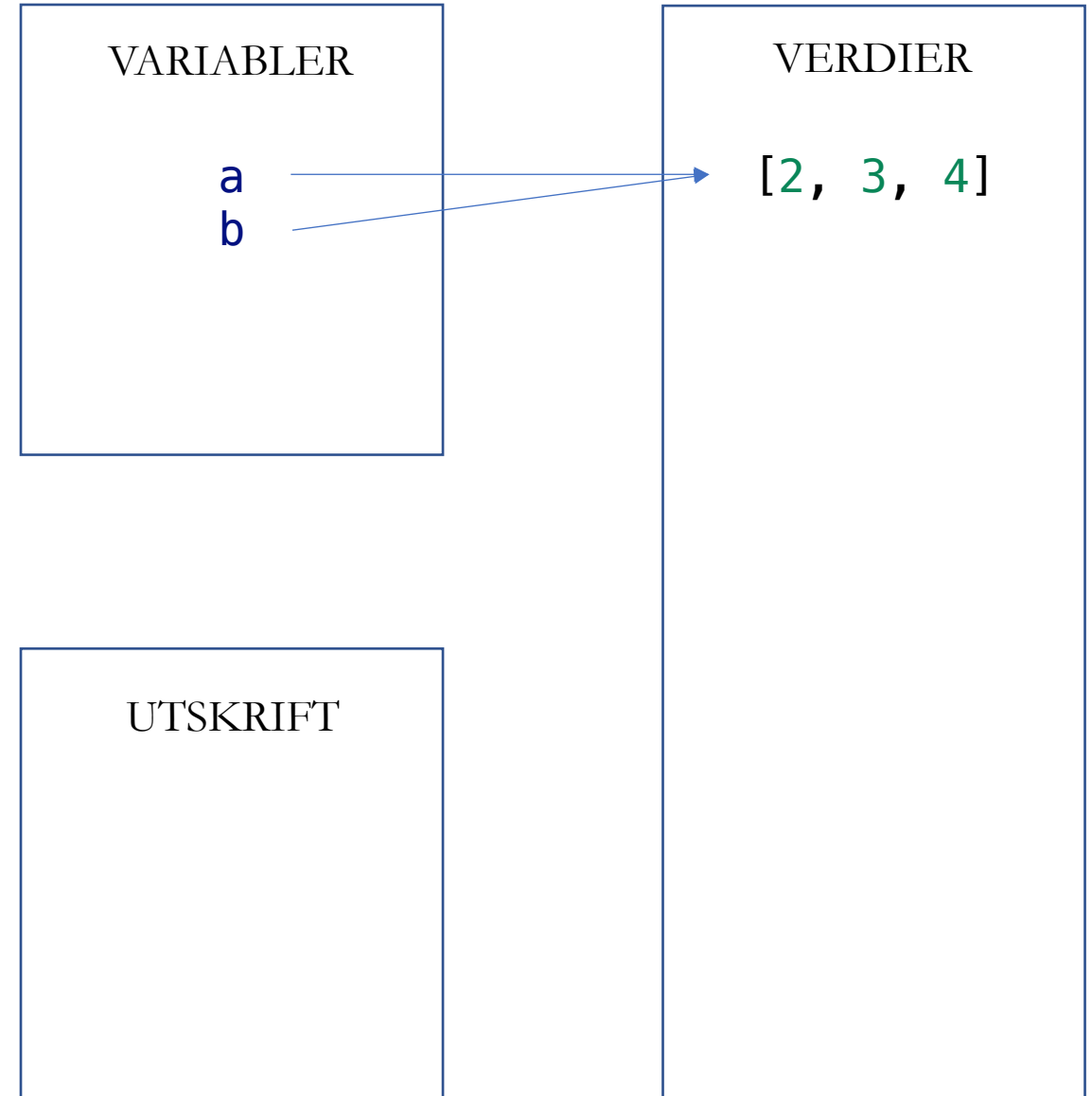
```
# Ikke-destruktiv endring
```

```
a = [99] + a[1:]
```

```
b = b[:1] + [42] + b[2:]
```

```
print(a)
```

```
print(b)
```



ALIAS

+ ikke-destruktive endringer

```
a = [2, 3, 4]
```

```
# Oppretter et alias
```

```
b = a
```

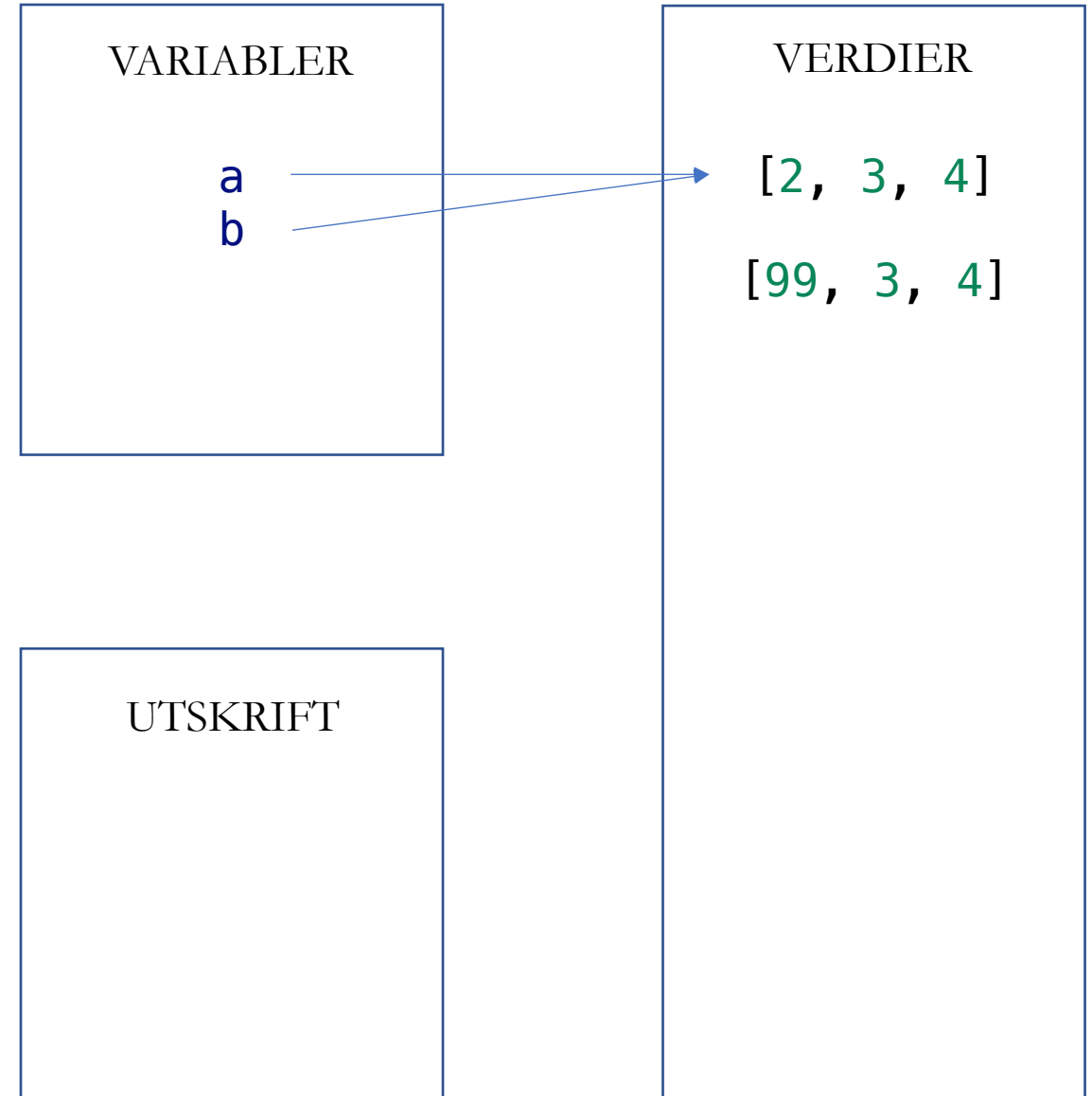
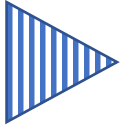
```
# Ikke-destruktiv endring
```

```
a = [99] + a[1:]
```

```
b = b[:1] + [42] + b[2:]
```

```
print(a)
```

```
print(b)
```



ALIAS

+ ikke-destruktive endringer

```
a = [2, 3, 4]
```

```
# Oppretter et alias
```

```
b = a
```

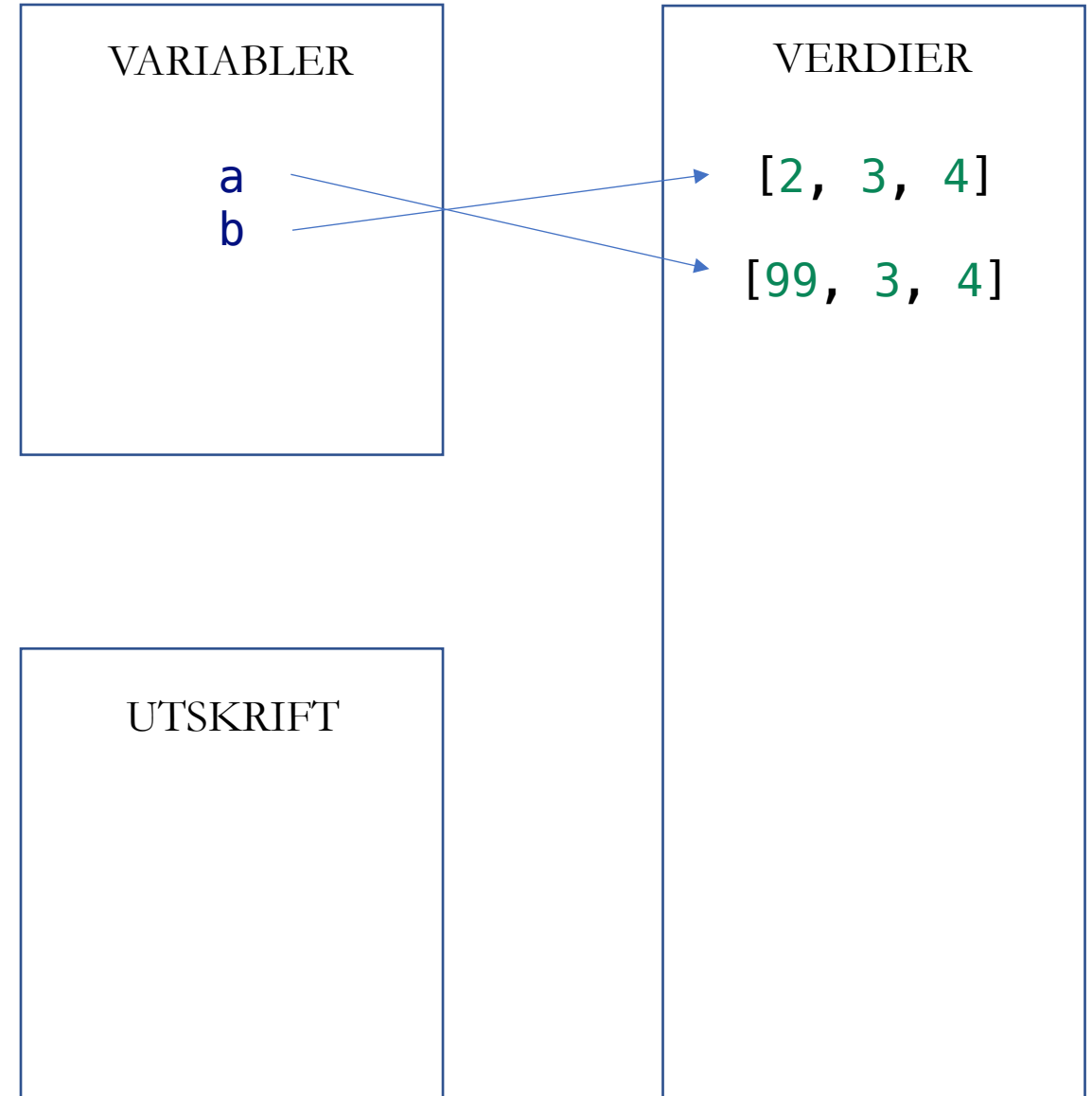
```
# Ikke-destruktiv endring
```

```
a = [99] + a[1:]
```

```
b = b[:1] + [42] + b[2:]
```

```
print(a)
```

```
print(b)
```



ALIAS

+ ikke-destruktive endringer

```
a = [2, 3, 4]
```

```
# Oppretter et alias
```

```
b = a
```

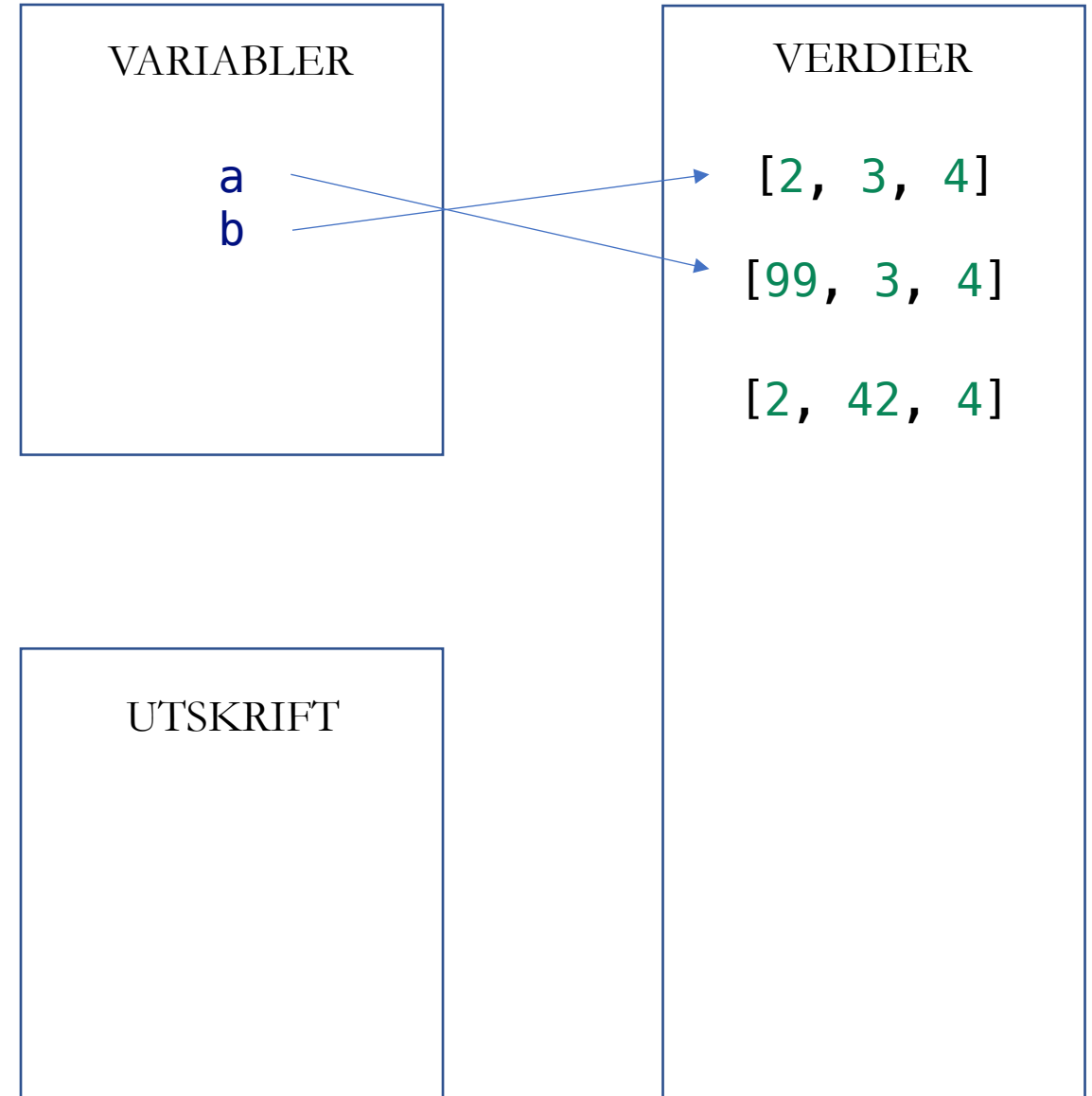
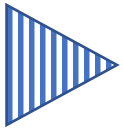
```
# Ikke-destruktiv endring
```

```
a = [99] + a[1:]
```

```
b = b[:1] + [42] + b[2:]
```

```
print(a)
```

```
print(b)
```



ALIAS

+ ikke-destruktive endringer

```
a = [2, 3, 4]
```

```
# Oppretter et alias
```

```
b = a
```

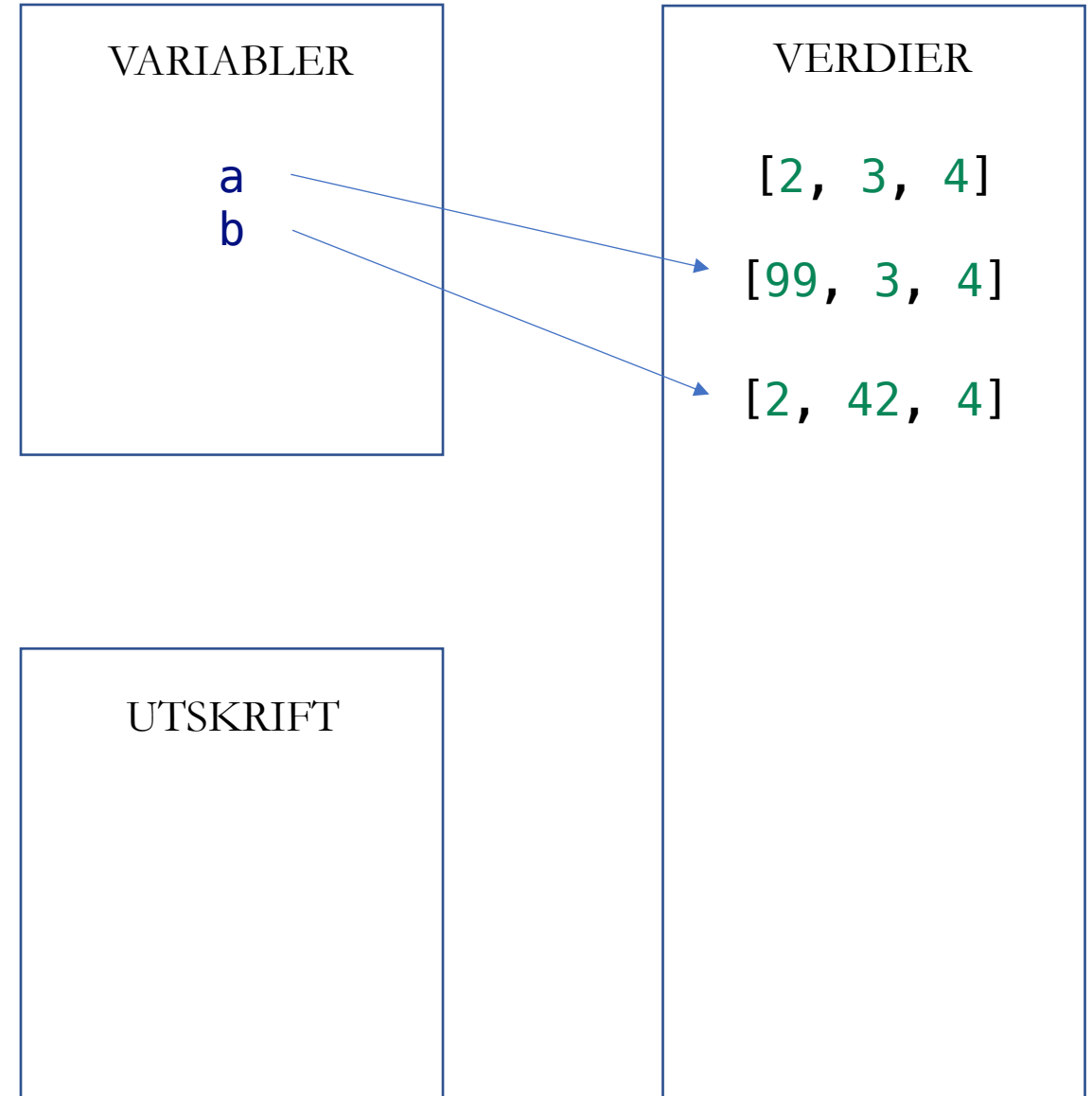
```
# Ikke-destruktiv endring
```

```
a = [99] + a[1:]
```

```
b = b[:1] + [42] + b[2:]
```

```
print(a)
```

```
print(b)
```



ALIAS

+ ikke-destruktive endringer

```
a = [2, 3, 4]
```

```
# Oppretter et alias
```

```
b = a
```

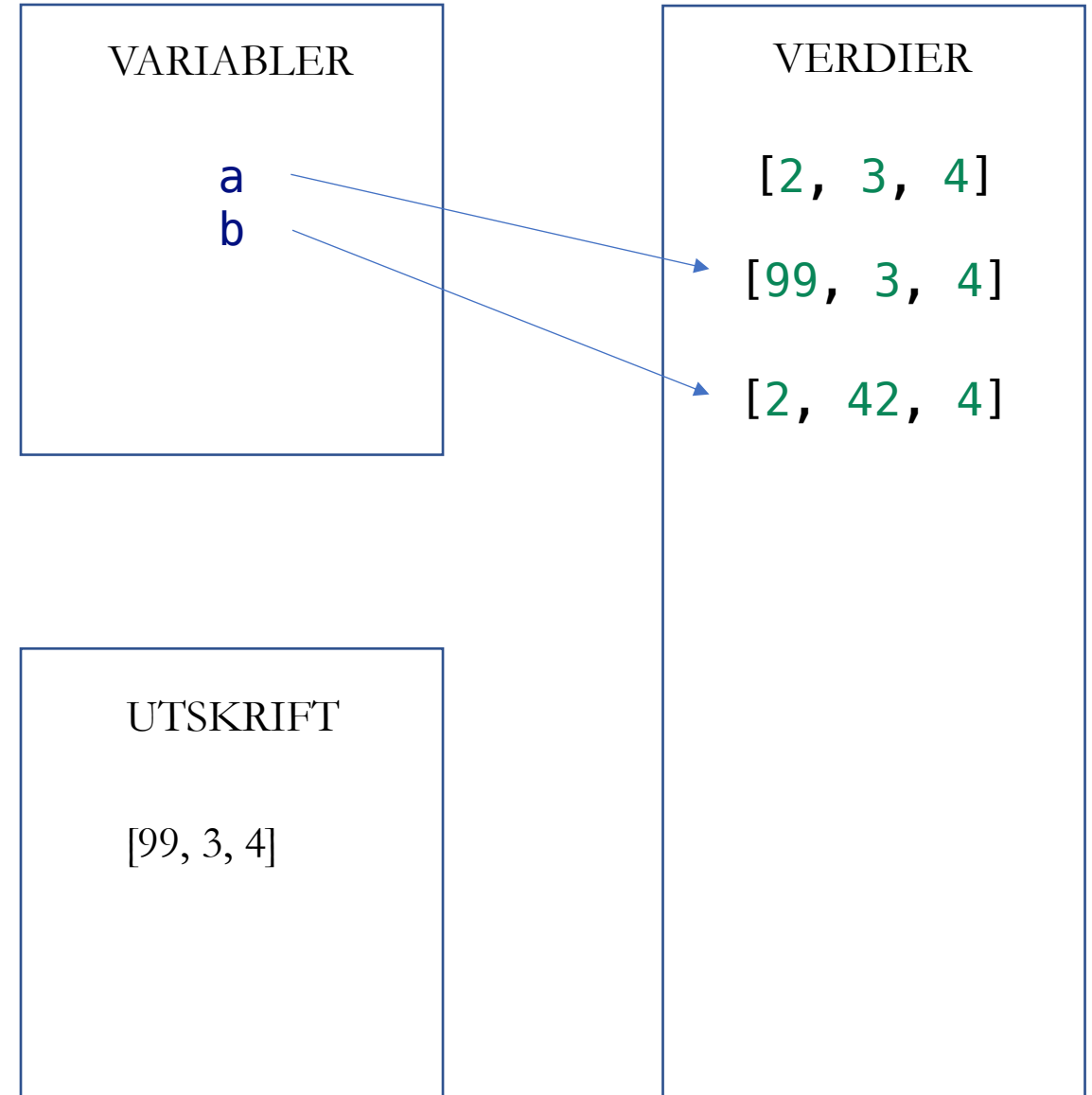
```
# Ikke-destruktiv endring
```

```
a = [99] + a[1:]
```

```
b = b[:1] + [42] + b[2:]
```

```
print(a)
```

```
print(b)
```



ALIAS

+ ikke-destruktive endringer

```
a = [2, 3, 4]
```

```
# Oppretter et alias
```

```
b = a
```

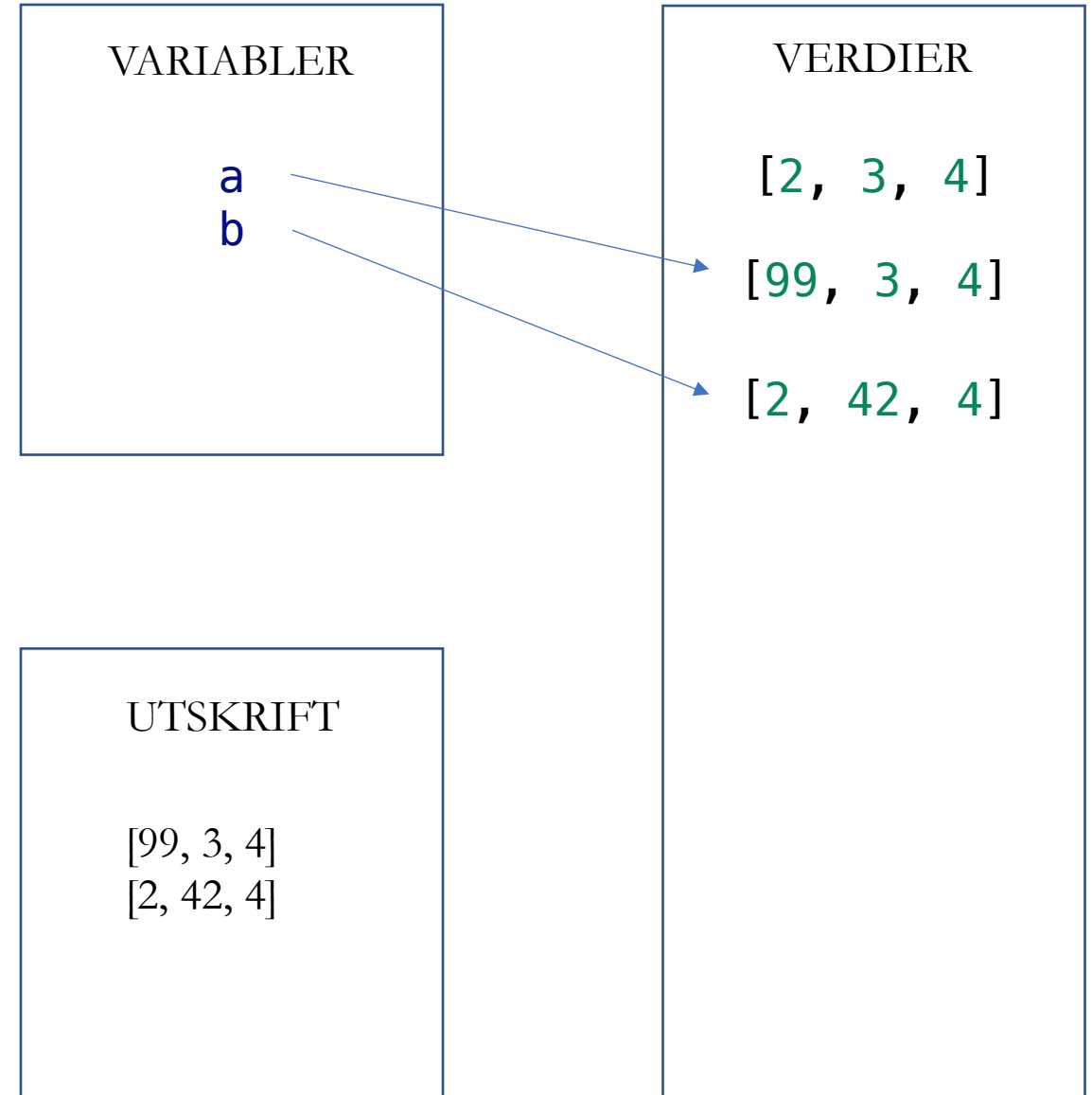
```
# Ikke-destruktiv endring
```

```
a = [99] + a[1:]
```

```
b = b[:1] + [42] + b[2:]
```

```
print(a)
```

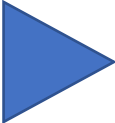
```
print(b)
```



ALIAS

+ destruktiv funksjon

```
def foo(bar):  
    bar[2] = 42
```



```
a = ["p", "q", "r", "s"]  
foo(a)  
print(a)
```

VARIABLER

VERDIER

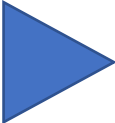
```
["p", "q", "r", "s"]
```

UTSKRIFT

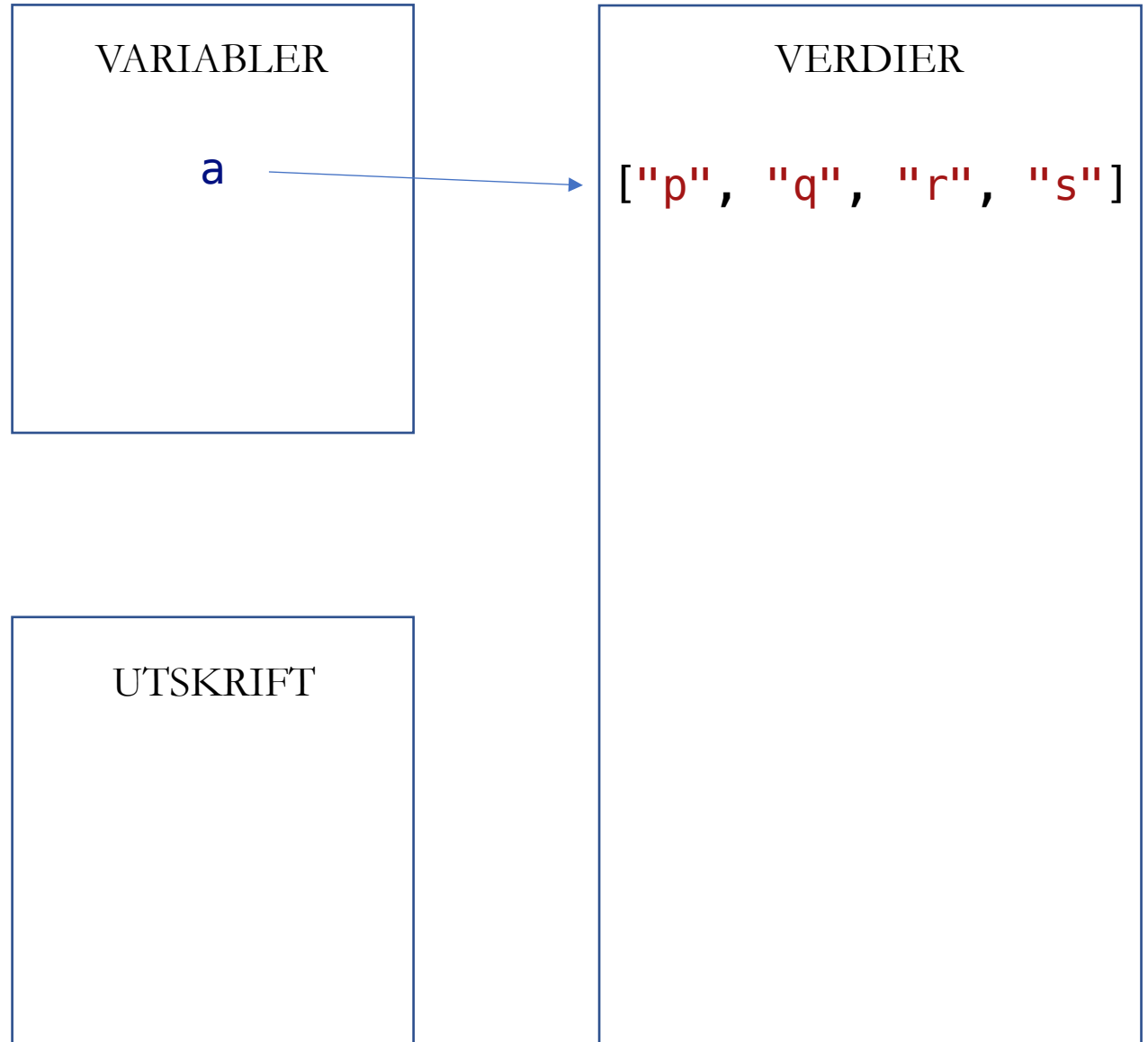
ALIAS

+ destruktiv funksjon

```
def foo(bar):  
    bar[2] = 42
```



```
a = ["p", "q", "r", "s"]  
foo(a)  
print(a)
```

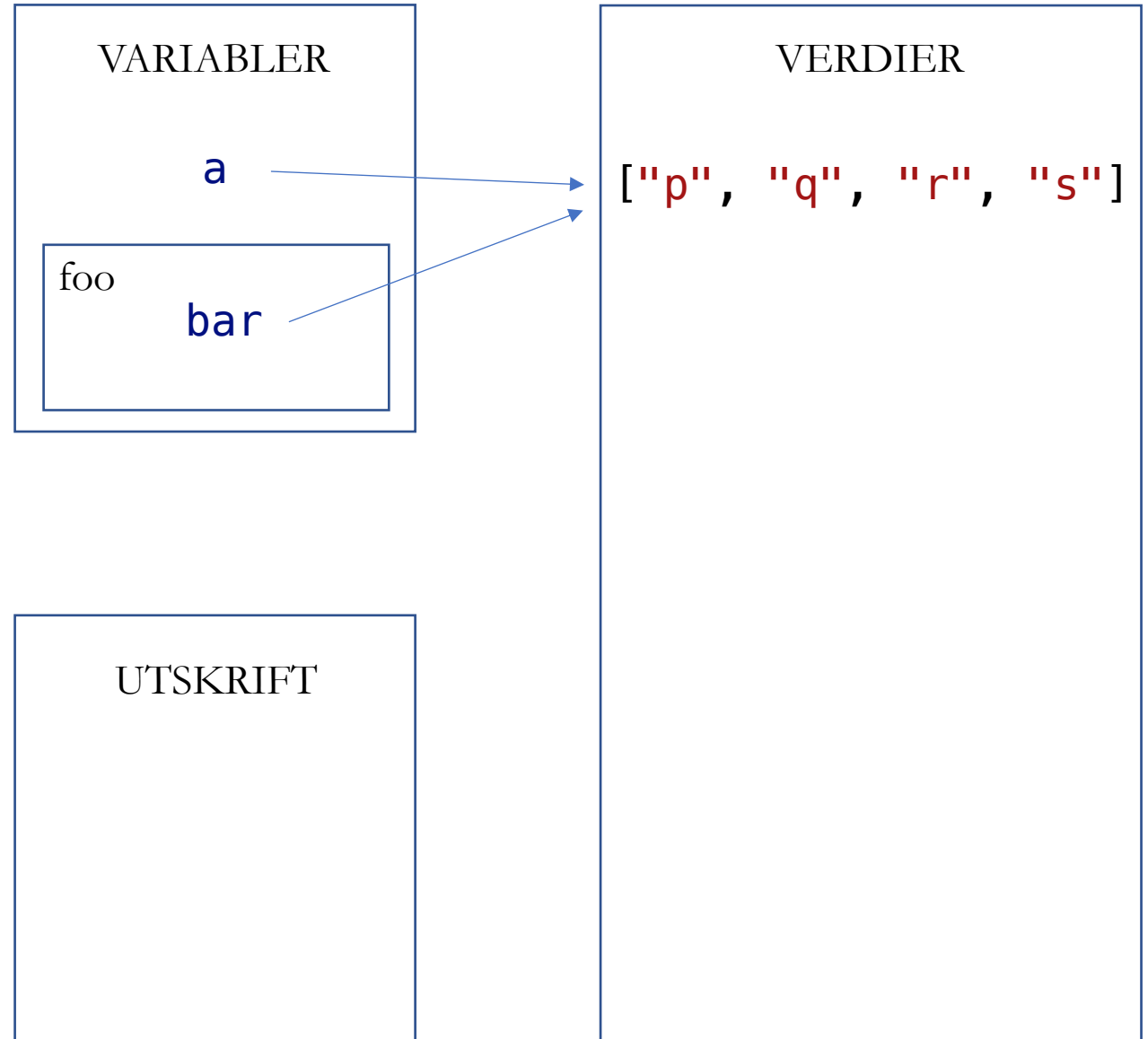


ALIAS

+ destruktiv funksjon

```
def foo(bar):  
    bar[2] = 42
```

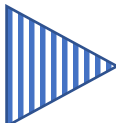
```
a = ["p", "q", "r", "s"]  
foo(a)  
print(a)
```



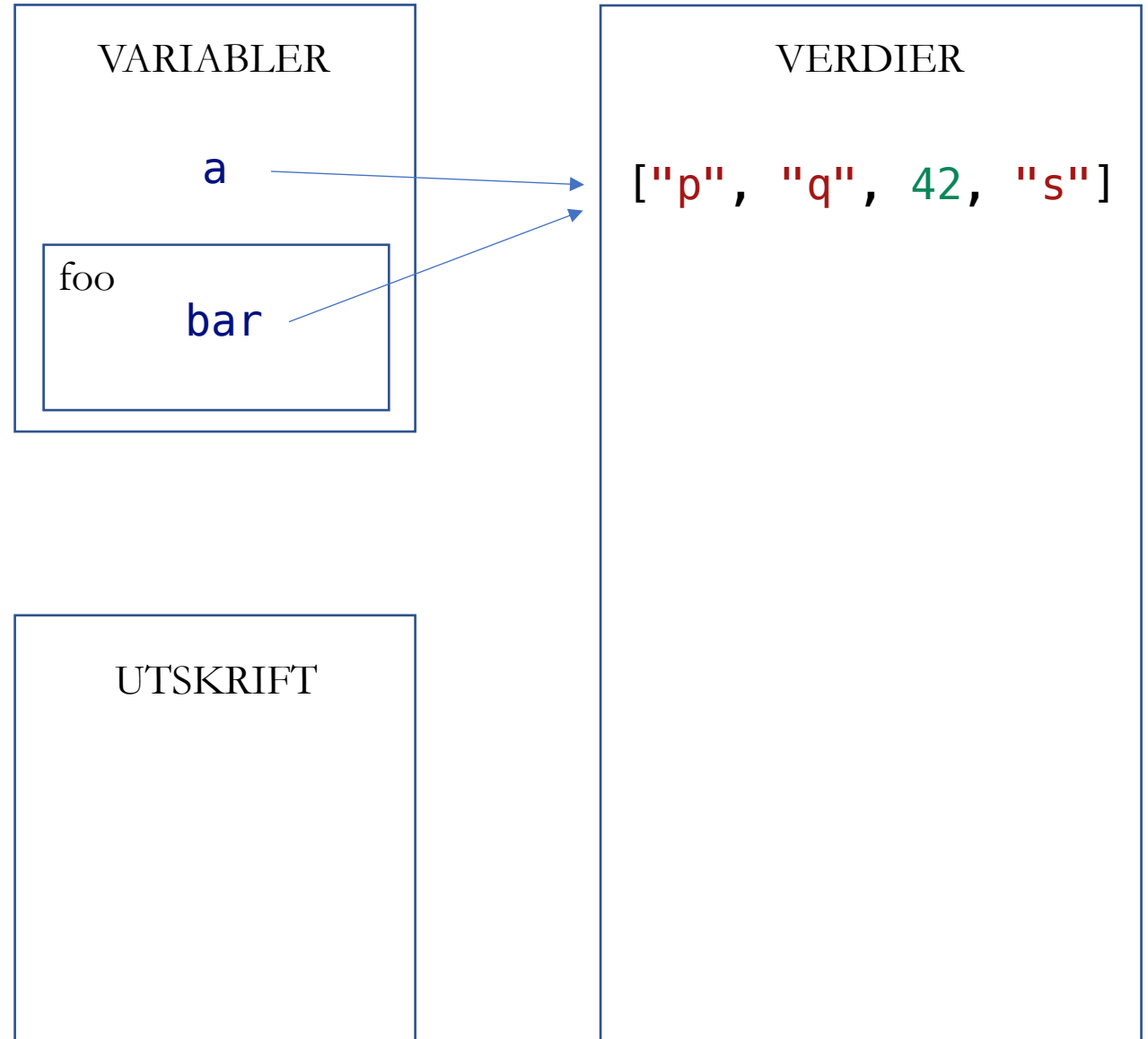
ALIAS

+ destruktiv funksjon

```
def foo(bar):  
    bar[2] = 42
```



```
a = ["p", "q", "r", "s"]  
foo(a)  
print(a)
```

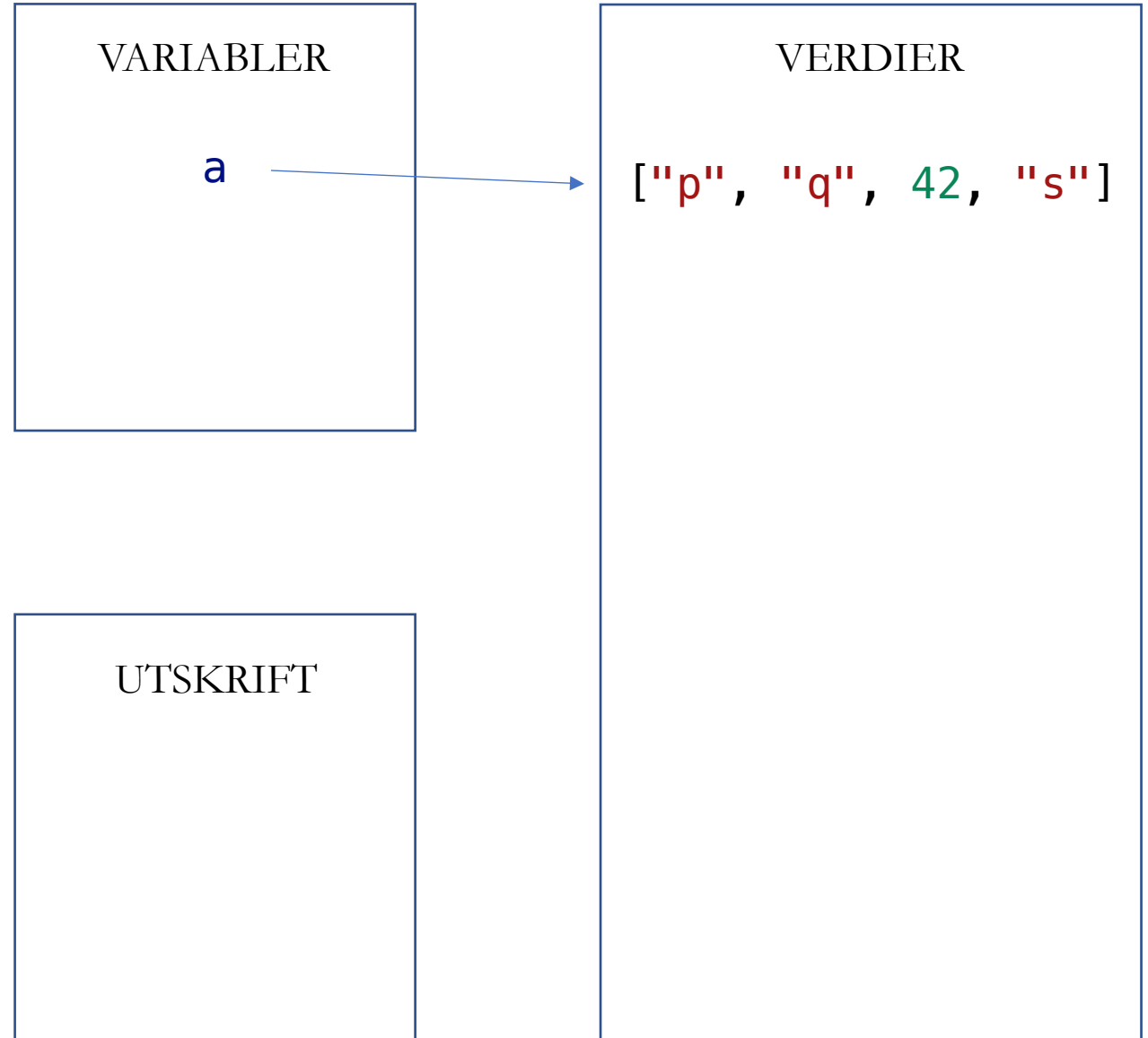


ALIAS

+ destruktiv funksjon

```
def foo(bar):  
    bar[2] = 42
```

```
a = ["p", "q", "r", "s"]  
foo(a)  
print(a)
```

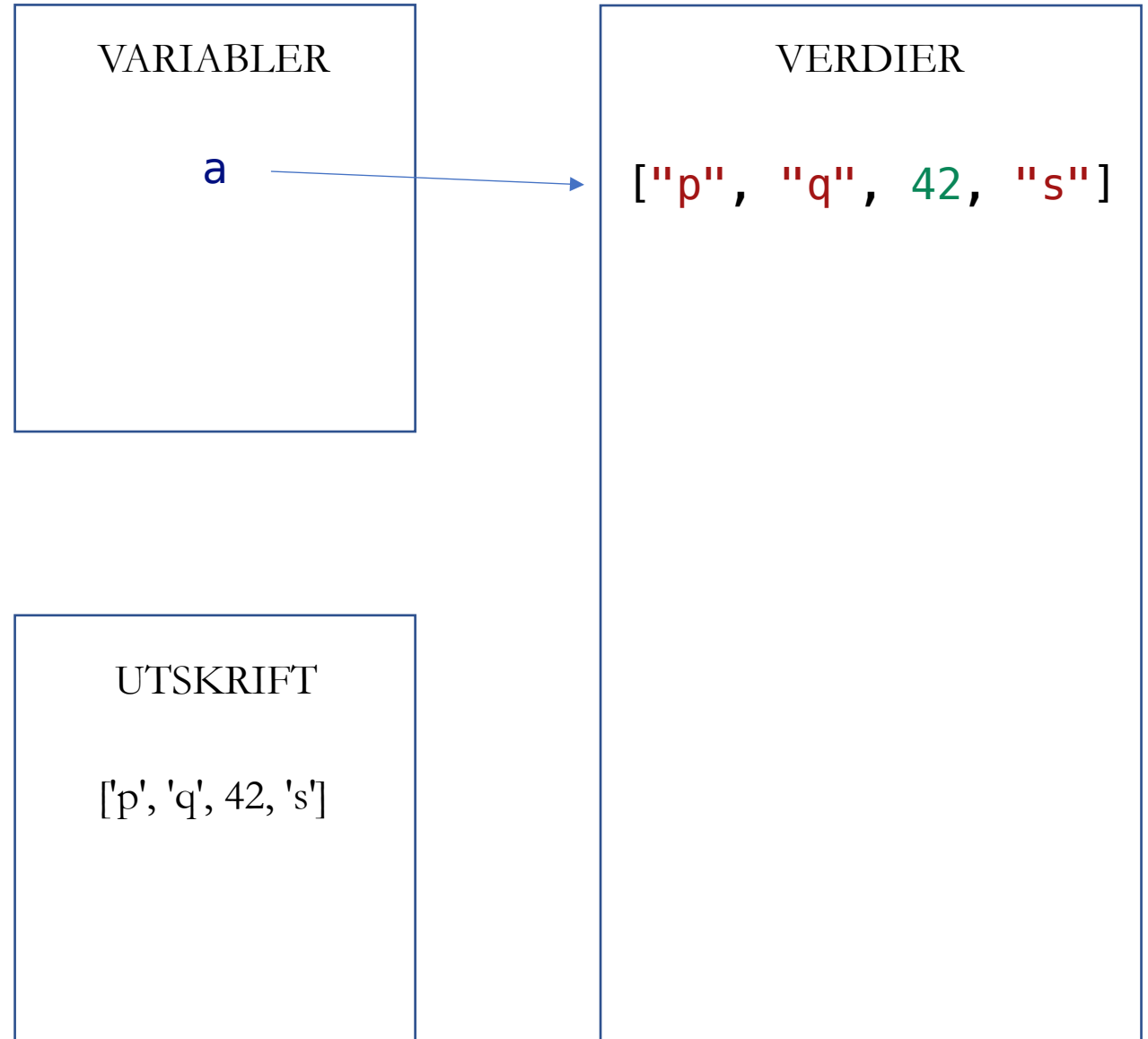


ALIAS

+ destruktiv funksjon

```
def foo(bar):  
    bar[2] = 42
```

```
a = ["p", "q", "r", "s"]  
foo(a)  
print(a)
```



KOPIERING

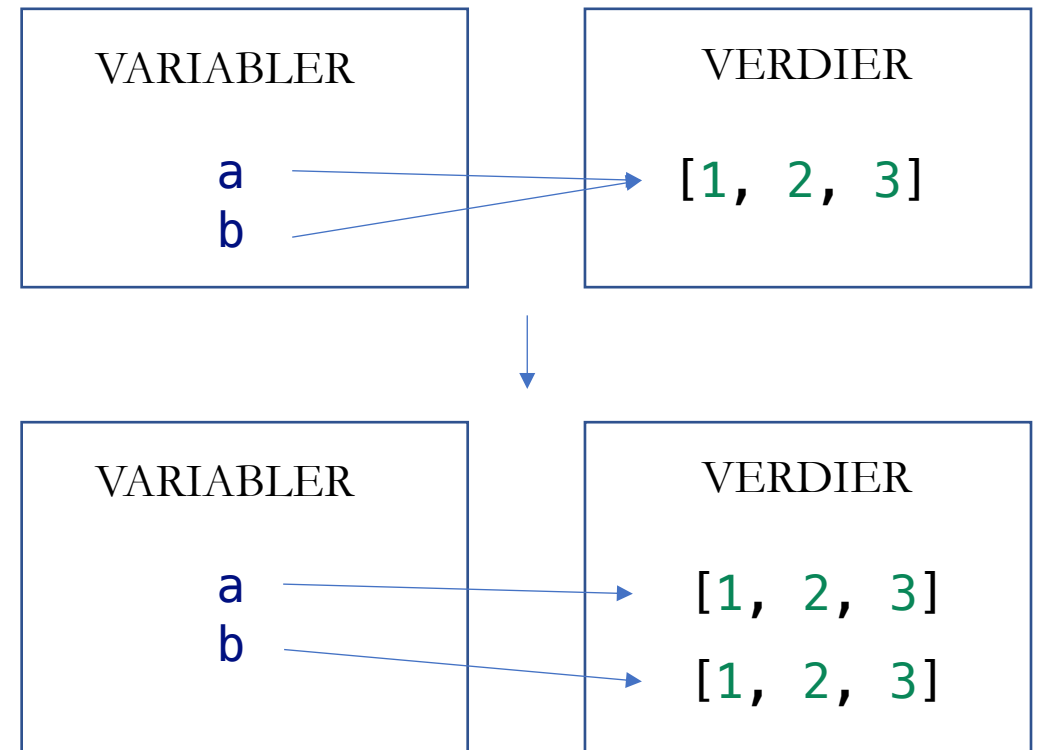
- Vi kan bryte et alias ved å lage en kopi

```
a = [1, 2, 3]
```

```
b = a
```

```
b = b.copy()
```

- Destruktive operasjoner på kopien er uten sideeffekter
- Bruker mer tid og minne





UNIVERSITETET I BERGEN

GRAFIKK

INF100

HØST 2022

Torstein Strømme

<https://inf100.i.uib.no/notat/grafikk1/>