

Inleiding Programmeren

Algoritme van Euclides

Toon Calders

toon.calders@uantwerpen.be

Algoritme van Euclides (300 v. Chr.)

Probleem: Bepaal GGD van 2 getallen

Invoer: 2 getallen

Uitvoer: de GGD van deze twee getallen

1. Noem het grootste van de beide getallen A , het andere B .
2. Trek B net zo vaak van A af totdat er 0 overblijft of een getal kleiner dan B ($A \bmod B$).
3. Wanneer er 0 overblijft, is B de ggd.
4. Zo niet, herhaal dan het algoritme met B en wat er van A over is.

Algoritme van Euclides (300 v. Chr.)

- Input: Getallen 35 en 49

Algoritme van Euclides (300 v. Chr.)

- Input: Getallen 35 en 49

A 49

B 35

Algoritme van Euclides (300 v. Chr.)

- Input: Getallen 35 en 49

A 49 14

B 35 35

Algoritme van Euclides (300 v. Chr.)

- Input: Getallen 35 en 49

A	49	14	35
B	35	35	14

Algoritme van Euclides (300 v. Chr.)

- Input: Getallen 35 en 49

A	49	14	35	21
B	35	35	14	14

Algoritme van Euclides (300 v. Chr.)

- Input: Getallen 35 en 49

A	49	14	35	21	7
B	35	35	14	14	14

Algoritme van Euclides (300 v. Chr.)

- Input: Getallen 35 en 49

A	49	14	35	21	7	14
B	35	35	14	14	14	7

Algoritme van Euclides (300 v. Chr.)

- Input: Getallen 35 en 49

A	49	14	35	21	7	14	7
B	35	35	14	14	14	7	7

Algoritme van Euclides (300 v. Chr.)

- Input: Getallen 35 en 49

A	49	14	35	21	7	14	7	0
B	35	35	14	14	14	7	7	7

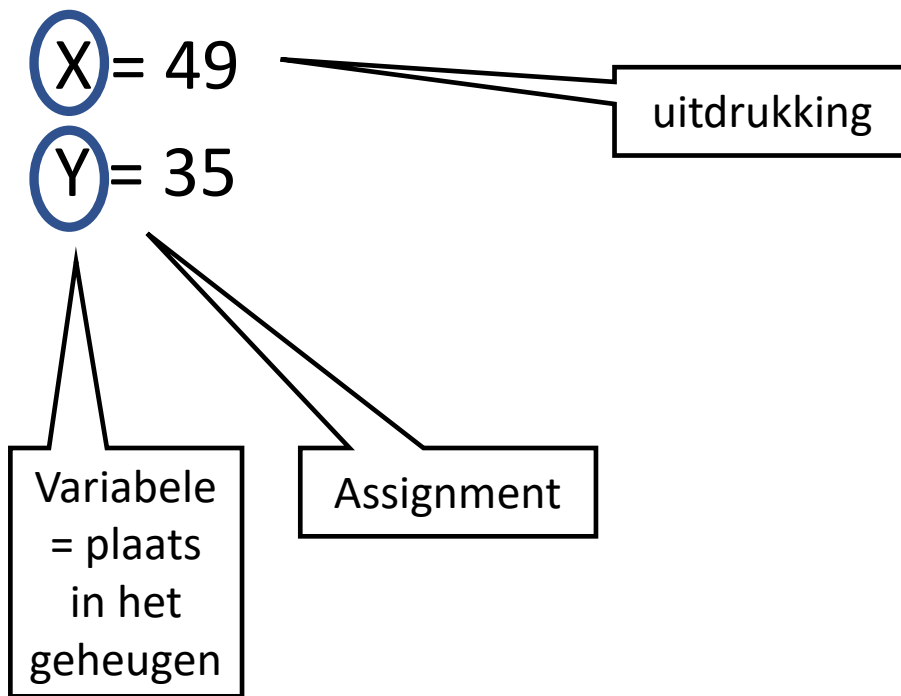
- Output: GGD = 7

Implementatie in Python

- Stap-voor-stap vertaling van dit algoritme met behulp van basisoperaties die Python begrijpt
 - Invoer
 - Bewaren van een waarde om later te gebruiken
 - Indien ... dan ... anders ...
 - Zolang ... doe ...
- We maken een implementatie in Python

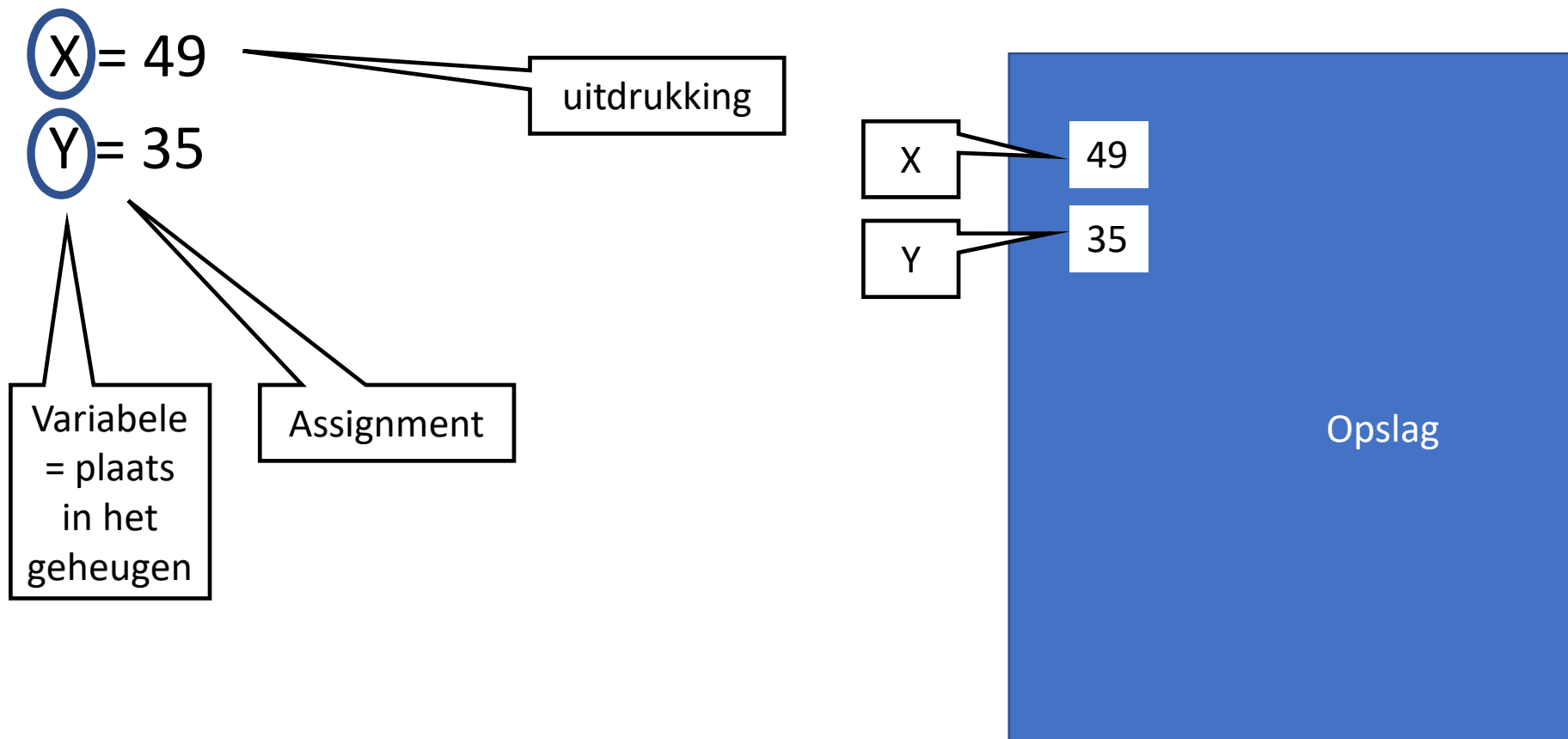
Implementatie in Python

- Invoer: 2 getallen



Implementatie in Python

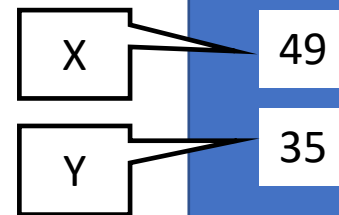
- Invoer: 2 getallen



Implementatie in Python

- Noem het grootste van de beide getallen A , het andere B = Als X groter is dan Y , maak dan A gelijk aan X en B gelijk aan Y , anders maak je A gelijk aan Y en B gelijk aan X

```
if X>Y:  
    A = X  
    B = Y  
else:  
    B = X  
    A = Y
```

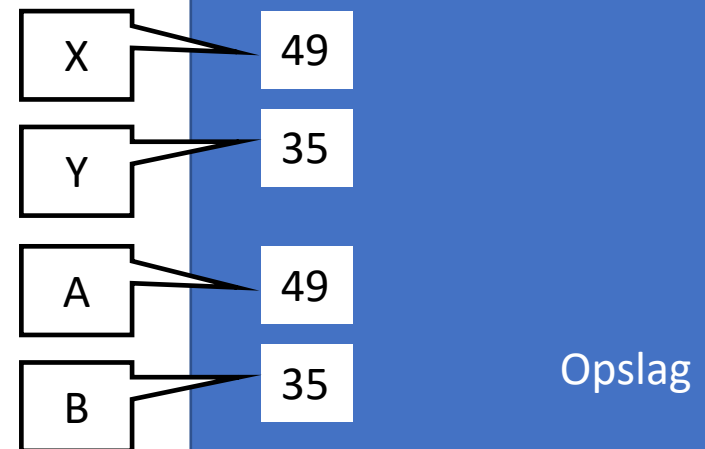


Opslag

Implementatie in Python

- Noem het grootste van de beide getallen A , het andere B = Als X groter is dan Y , maak dan A gelijk aan X en B gelijk aan Y , anders maak je A gelijk aan Y en B gelijk aan X

```
if X>Y:  
    A = X  
    B = Y  
else:  
    B = X  
    A = Y
```

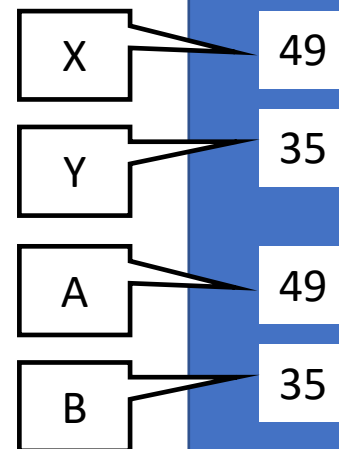


Implementatie in Python

- Trek B net zo vaak van A af totdat er 0 overblijft of een getal kleiner dan B ($A \bmod B$).

→ while $A \geq B$:

$A = A - B$



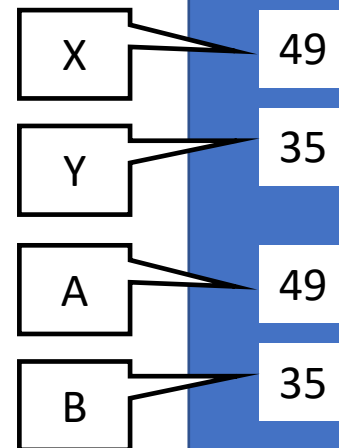
Opslag

Implementatie in Python

- Trek B net zo vaak van A af totdat er 0 overblijft of een getal kleiner dan B ($A \bmod B$).

→ `while A >= B:`

`A = A - B`



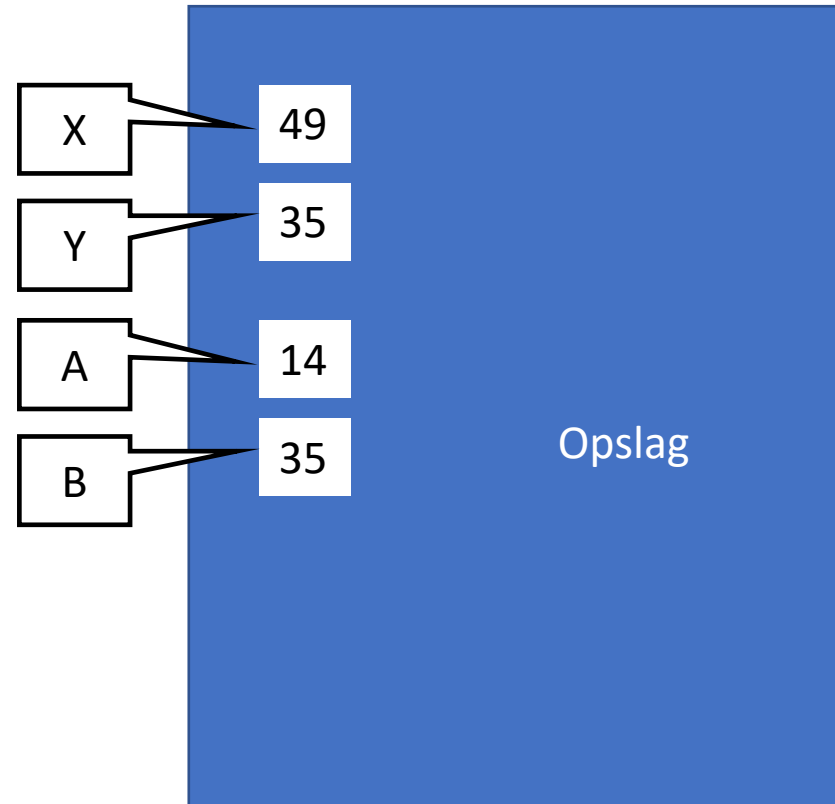
Opslag

Implementatie in Python

- Trek B net zo vaak van A af totdat er 0 overblijft of een getal kleiner dan B ($A \bmod B$).

while $A \geq B$:

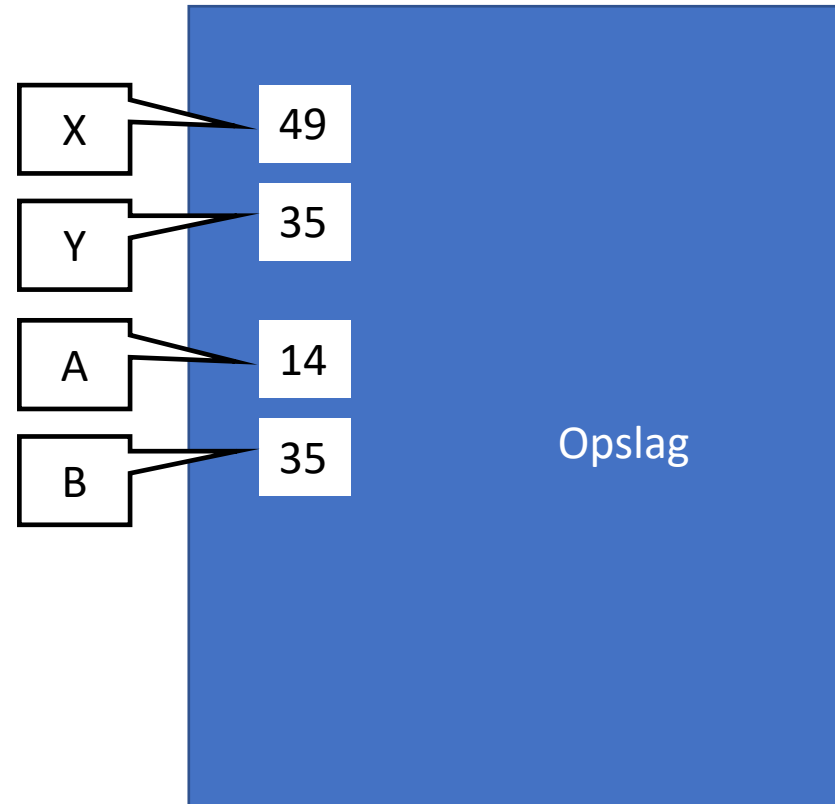
$A = A - B$



Implementatie in Python

- Trek B net zo vaak van A af totdat er 0 overblijft of een getal kleiner dan B ($A \bmod B$).

→ while $A \geq B$:
 $A = A - B$

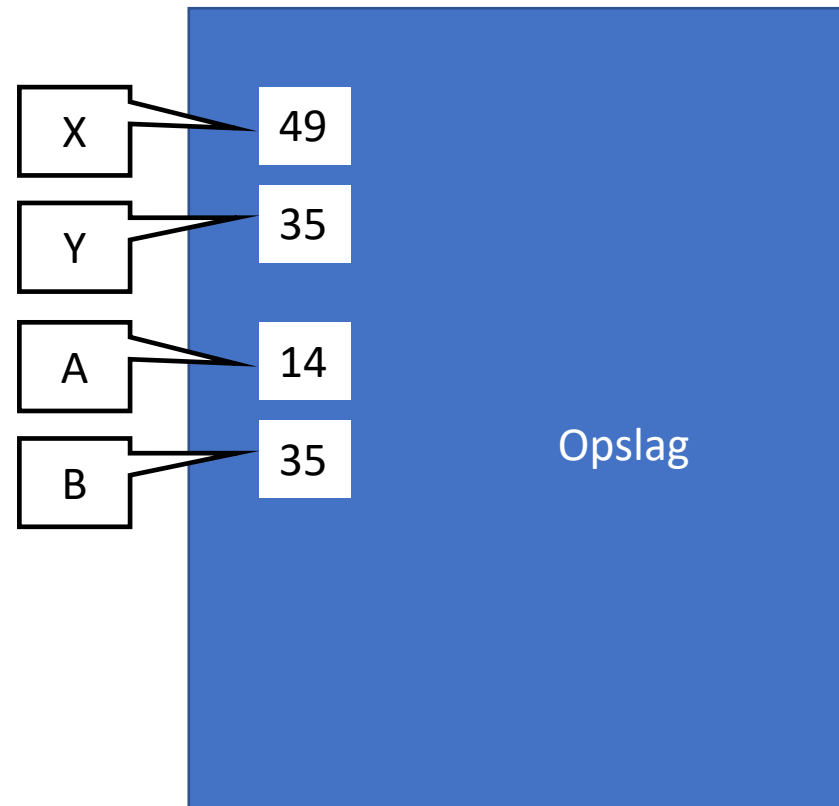


Implementatie in Python

- Trek B net zo vaak van A af totdat er 0 overblijft of een getal kleiner dan B ($A \bmod B$).

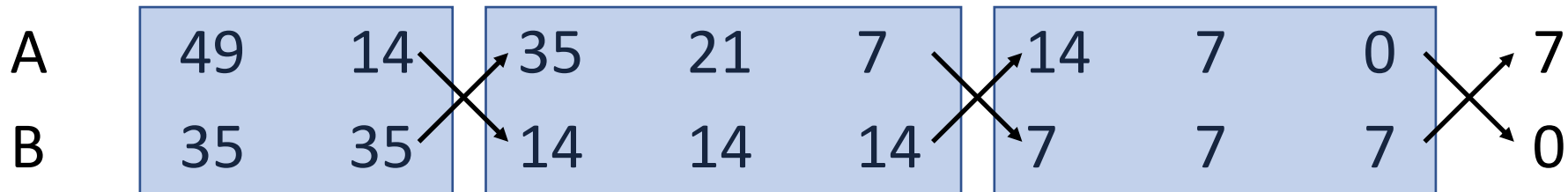
while $A \geq B$:

$A = A - B$



Implementatie in Python

3. Wanneer er 0 overblijft, is B de ggd.
4. Anders, herhaal dan het algoritme met B en wat er van A over is



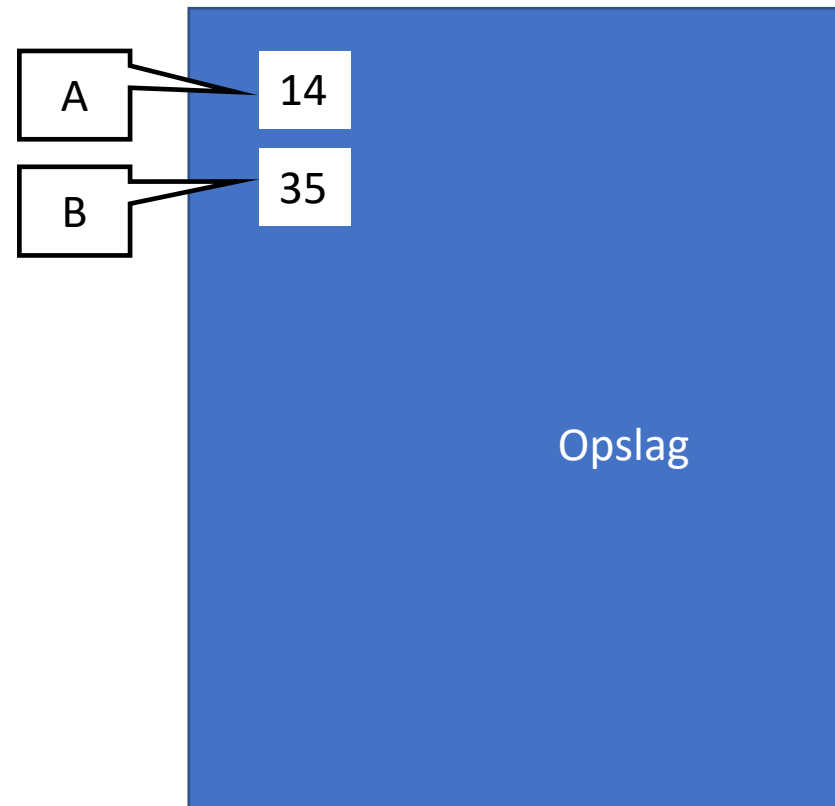
Wissel A en B om

- Volgende code werkt NIET:



A=B

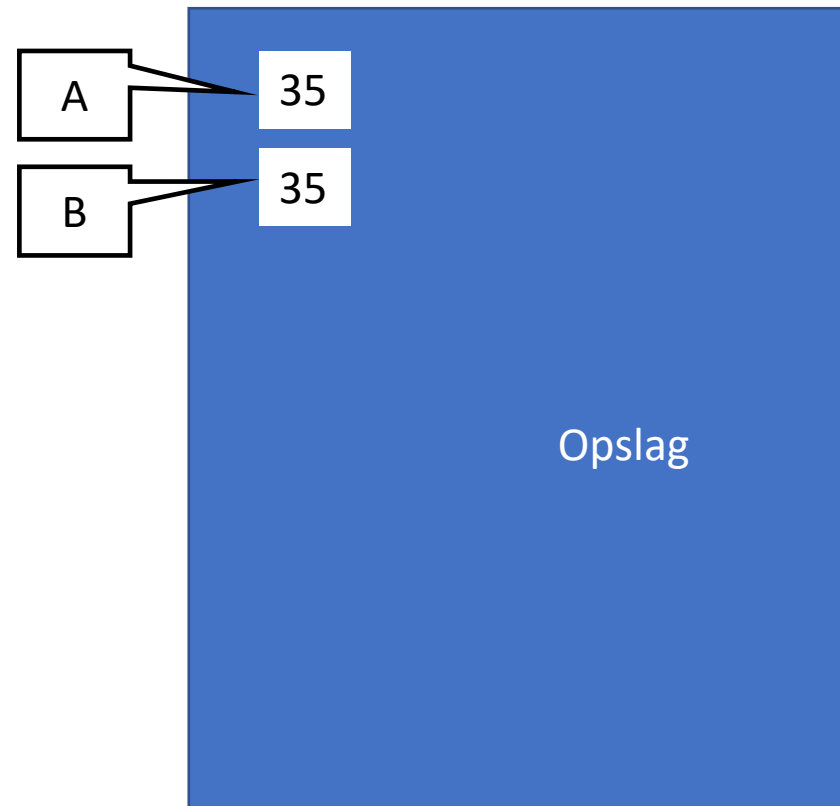
B=A



Wissel A en B om

- Volgende code werkt NIET:

→
A=B
B=A

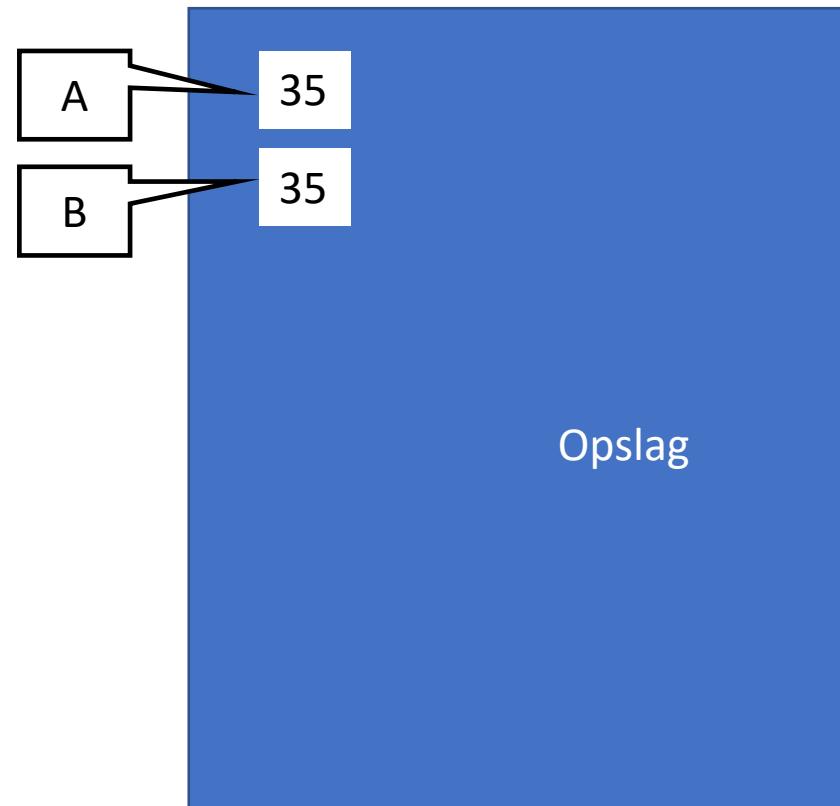


Wissel A en B om

- Volgende code werkt NIET:

A=B

B=A



Wissel A en B om

- Volgende code werkt NIET:

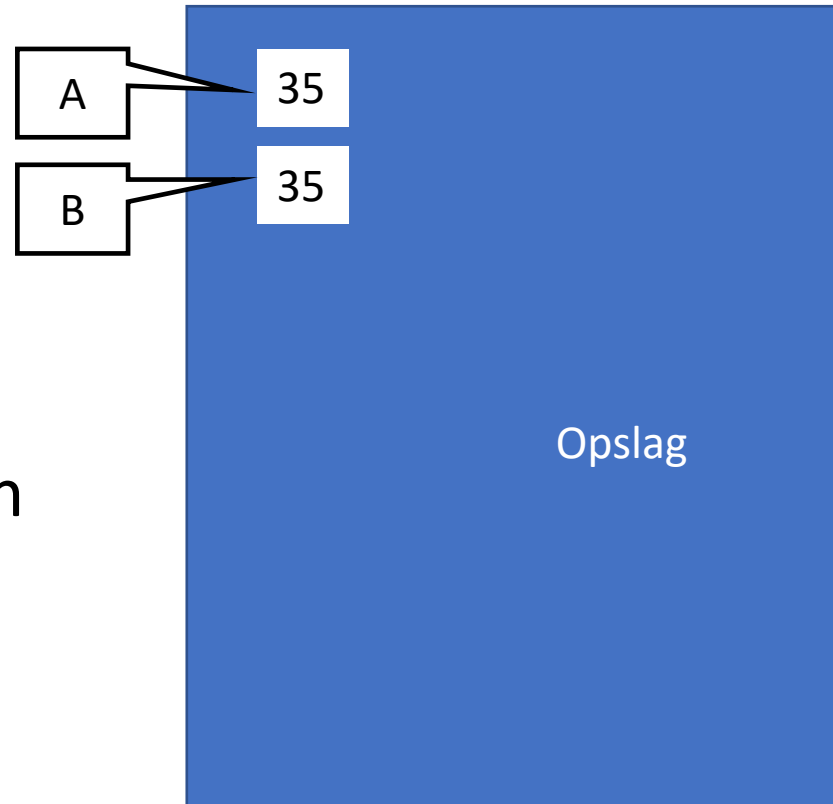
A=B

B=A



Waarde van A werd
overschreven

➔ We moeten tijdelijk de
waarde van A onthouden



Wissel A en B om

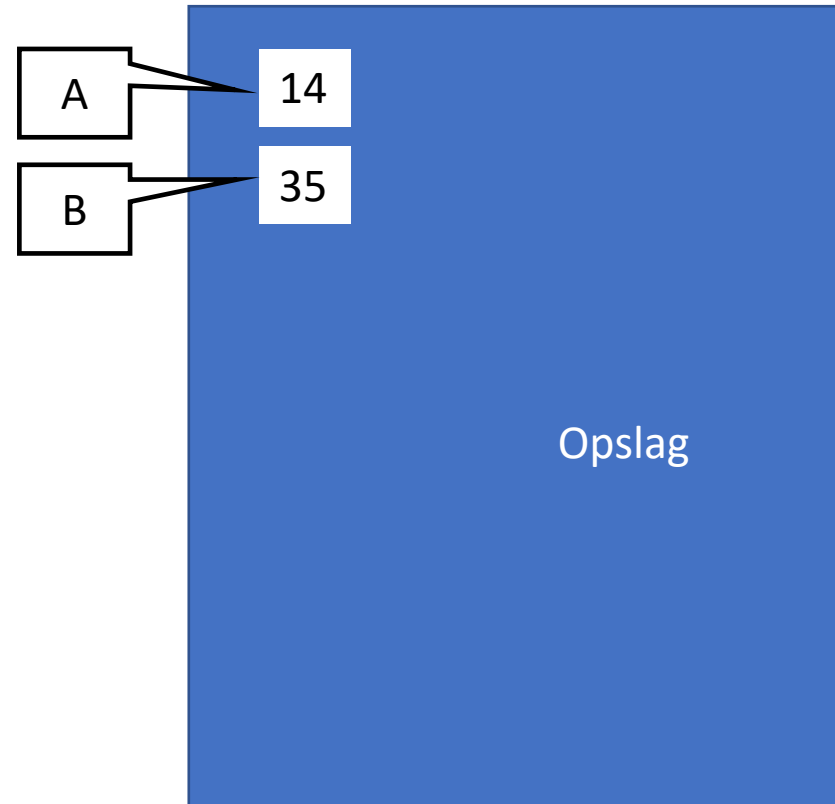
- Volgende code werkt WEL:



temp = A

A=B

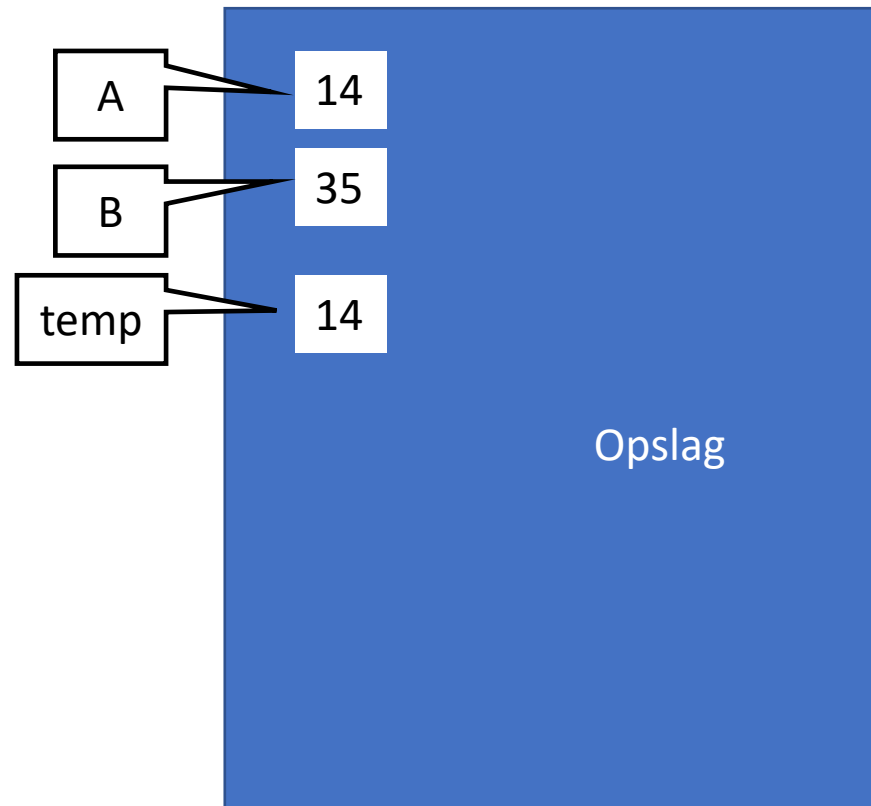
B=temp



Wissel A en B om

- Volgende code werkt WEL:

→ temp = A
A=B
B=temp



Wissel A en B om

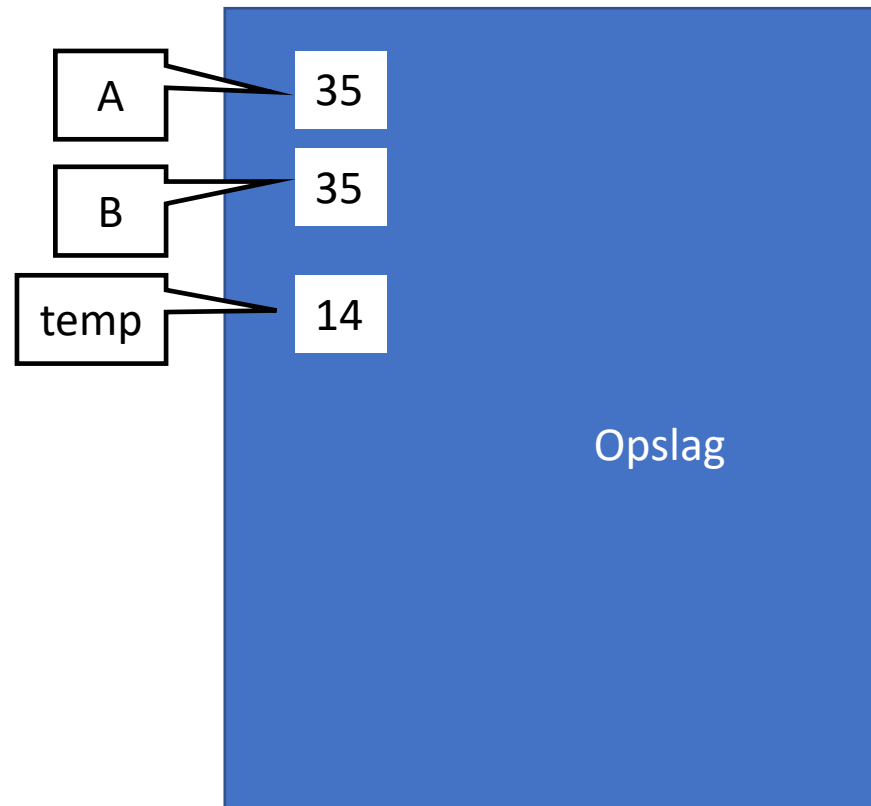
- Volgende code werkt WEL:

temp = A



A=B

B=temp



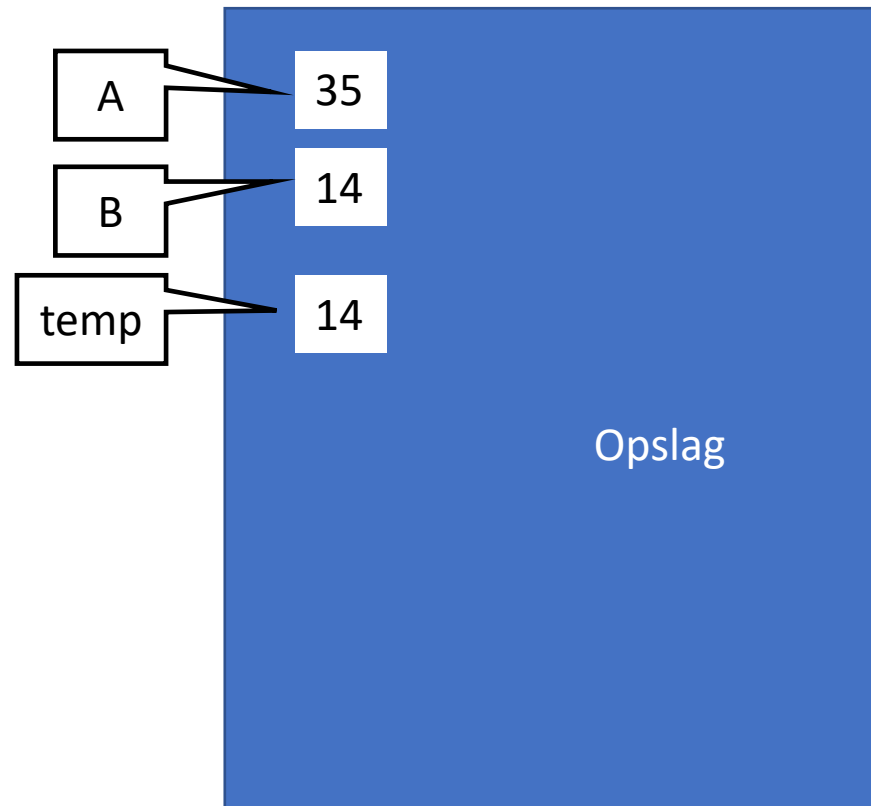
Wissel A en B om

- Volgende code werkt WEL:

temp = A

A=B

B=temp



Implementatie in Python

```
X=int(input("Geef het eerste getal: "))
Y=int(input("Geef het tweede getal: "))

if X>Y:
    A = X
    B = Y
else:
    B = X
    A = Y

while A!=0:
    while A>=B:
        A = A - B
    if A!=0:
        temp=A
        A=B
        B=temp

print(B)
```

Implementatie in Python

```
X=int(input("Geef het eerste getal: "))
Y=int(input("Geef het tweede getal: "))

if X>Y:
    A = X
    B = Y
else:
    B = X
    A = Y

while A!=0:
    while A>=B:
        A = A - B
    if A!=0:
        temp=A
        A=B
        B=temp

print(B)
```


Implementatie in Python

```
A=int(input("Geef het eerste getal: "))
B=int(input("Geef het tweede getal: "))

while A!=0:
    while A>=B:
        A = A - B
    if A!=0:
        temp=A
        A=B
        B=temp

print(B)
```

Wat moet ik onthouden?

- Probleem → algoritme → implementatie
- Programmeertaal is een formele taal
- Enkele basisconstructies:
 - Variabelen: manier om een plaats in het geheugen te adresseren
 - If – else: conditionele uitvoering van code
 - While ... : herhaling / looping

Variabelen (Python)

- Referentie naar een plaats in het geheugen waar een waarde staat
 - Wordt aangemaakt bij eerste “assignment”
 - Heeft een type

```
I = 5                # I is een integer
B = True             # B is een Boolean
F = 0.53             # F is een floating point getal
                     # ("reeel getal")
S = "Ab cd 3"        # S is een string
                     # Gebruik enkele of dubbele
                     # aanhalingstekens
```

Voorbeelden Expressies (Python)

Expression	Result	Type
1+5	6	int
1/5	0.2 *	float *
10 // 3	3	int
10.0 // 3	3.0	float
10%3	1	int
1 + (3*5)//2.0	8.0	float
2**4==16.0	True	bool
(2!=3) or not (2!=5)	True	bool
"ab"+"bc"	"abbc"	string

* Uitkomst kan verschillen voor Python versies < 3.0

Type cast

- Soms willen we een variabele *converteren*
 - int $\rightarrow$ float
 - string $\rightarrow$ int
- Hiervoor gebruiken we volgende *functies* :

Doel type	Expressie die variabele X converteert
string	str(X)
int	int(x)
float	float(X)
bool	bool(X)

- Niet mogelijk? **Foutmelding!**

Enkele nuttige *functies*

- Een *functie* is een stukje bestaande code dat we kunnen *aanroepen*.
 - Voeren een opdracht uit en/of
 - Genereren nieuwe waarden
- Standaard in- en uitvoer: `print(expr, expr, ...)` en `input(string)`
- Conversies: `str(expr)`, `int(expr)`, `bool(expr)`, `float(expr)`
- Geef type: `type(expr)`

Types

- Een variabele is een referentie naar een plaats in het geheugen waar een waarde staat
 - Wordt aangemaakt bij eerste “assignment”
 - Heeft een **type**

```
I = 5          # I is een integer
B = True       # B is een Boolean
F = 0.53       # F is een floating point getal
               # ("reeel getal")
S = "Ab cd 3"  # S is een string
               # Gebruik enkele of dubbele
               # aanhalingstekens
```

Type van een expressie (Python)

Expression	Result	Type
1+5	6	int
1/5	0.2 *	float *
10 // 3	3	int
10.0 // 3	3.0	float
10%3	1	int
1 + (3*5)//2.0	8.0	float
2**4==16.0	True	bool
(2!=3) or not (2!=5)	True	bool
"ab"+"bc"	"abbc"	string

* Uitkomst kan verschillen voor Python versies < 3.0

Type cast

- Soms willen we een variabele converteren
 - int $\rightarrow$ float
 - string $\rightarrow$ int
- Hiervoor gebruiken we volgende *functies* :

Doel type	Expressie die variabele X transformeert
string	str(X)
int	int(x)
float	float(X)
bool	bool(X)

- Niet mogelijk? **Foutmelding!**

Voorbeelden Expressies

```
print("Dit is waarom je conversies nodig hebt:")  
S=input("geef een getal:")  
I=int(S)
```

```
print("S+S=", S+S)  
print(str(I) + "+" + str(I) + "=", I+I)  
print(type(S+S), type(I+I))
```

Dit is waarom je conversies nodig hebt:

geef een getal: 12

S+S= 1212

12+12= 24

<class 'str'> <class 'int'>

Voorbeeld expressies

Dit is waarom je conversies nodig hebt:

geef een getal: blabla

```
Traceback (most recent call last):
```

```
  File "C:/Users/toon/Google Drive/courses/2016  
/Inleiding programmeren/lectures/01/types.py",  
line 42, in <module>
```

```
    X=int(input("Geef een nummer:"))
```

```
ValueError: invalid literal for int() with base  
10: 'blabla'
```

Snelle vraag

- Wat is het type van volgende uitdrukkingen?

$(a > 3) \text{ or } (b - 8 \neq 7)$

$7/3 + 5$

- | | |
|------------------|--------------------|
| (a) bool en int | (b) bool en float |
| (c) int en float | (d) float en float |

Snelle vraag

- Wat is het type van volgende uitdrukkingen?

$(a > 3) \text{ or } (b - 8 \neq 7)$

$7/3 + 5$

(a) bool en int

(b) bool en float

(c) int en float

(d) float en float

Opdracht

- Schrijf een Python programmaatje dat:
 - Aan de gebruiker een getal vraagt
 - Berekent of het gegeven getal een priemgetal is
 - De uitkomst van de berekening afprint

```
i = int(input("getal?"))    # lees input, zet om naar int,  
                           # sla op in i
```

```
i%2 : rest bij deling door 2
```

```
if i%2==0:  
    print(i,"is een even getal")
```