Universiteit Gent
Faculteit Wetenschappen

Programmeren in Python

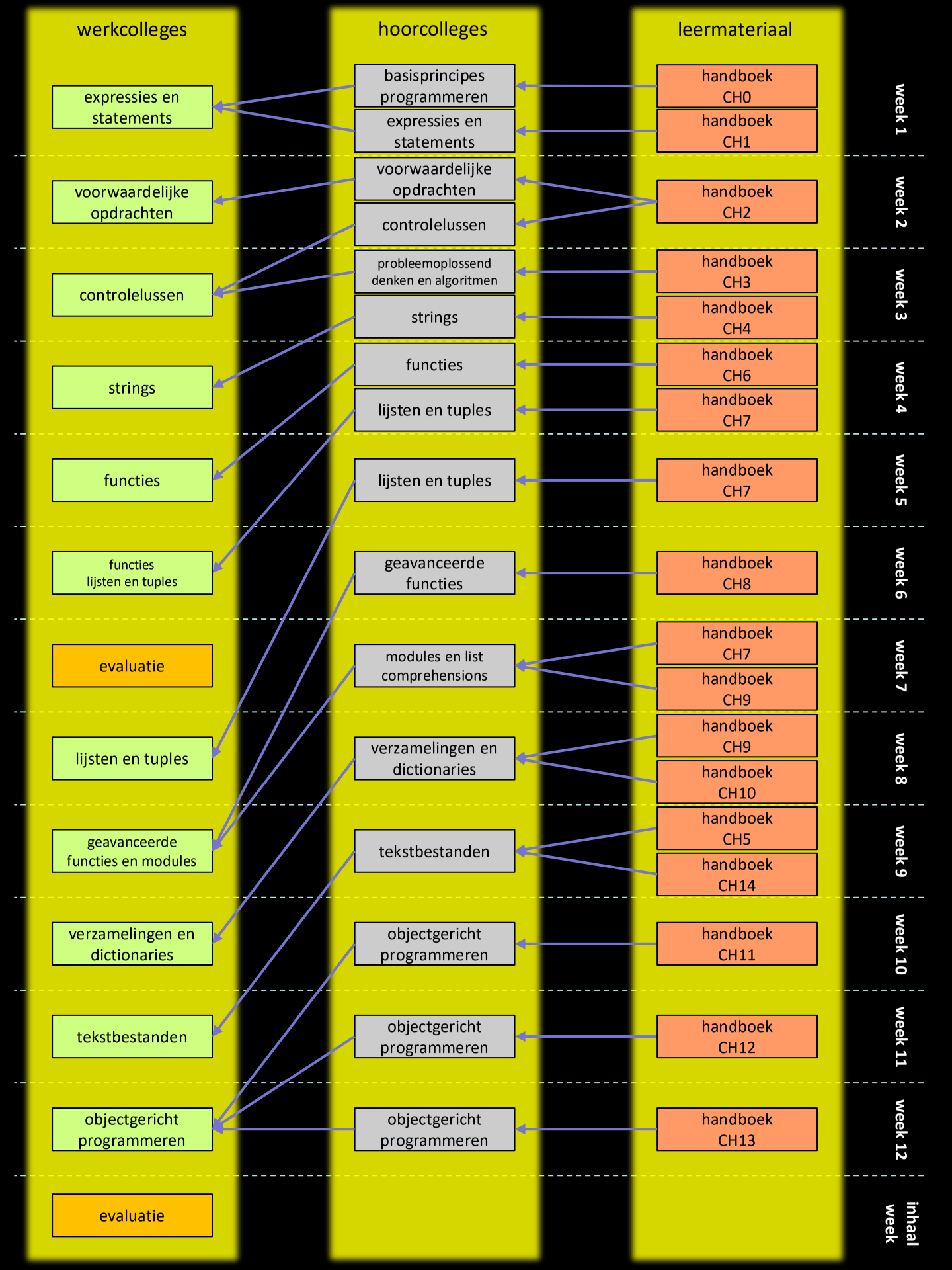
putting it all together

prof. dr. Peter Dawyndt

peter.dawyndt@ugent.be

@dawyndt

UNIVERSITEIT GENT


werkcolleges
hoorcolleges
leermateriaal
expressies en statements
voorwaardelijke opdrachten
controlelussen
strings
functies
functies lijsten en tuples
evaluatie
lijsten en tuples
geavanceerde functies en modules
verzamelingen en dictionaries
tekstbestanden
objectgericht programmeren
evaluatie
basisprincipes programmeren
expressies en statements
voorwaardelijke opdrachten
controlelussen
probleemoplossend denken en algoritmen
strings
functies
lijsten en tuples
lijsten en tuples
geavanceerde functies
modules en list comprehensions
verzamelingen en dictionaries
tekstbestanden
objectgericht programmeren
objectgericht programmeren
objectgericht programmeren
handboek CH0
handboek CH1
handboek CH2
handboek CH3
handboek CH4
handboek CH6
handboek CH7
handboek CH7
handboek CH8
handboek CH7
handboek CH9
handboek CH9
handboek CH10
handboek CH5
handboek CH14
handboek CH11
handboek CH12
handboek CH13
week 1
week 2
week 3
week 4
week 5
week 6
week 7
week 8
week 9
week 10
week 11
week 12
inhaal week


DODONA

Op welke dag ben je jarig?

socrative

socrative.com

room: 407122

formaat: DD/MM

= twee cijfers voor dag en maand

= geen geboortejaar

Voorbeeld

Ik ben geboren op 5 december 1975.

antwoord: 05/12

socrative

socrative.com

room: 407122

Wat is de kans dat in deze groep van XX studenten, minstens twee studenten op dezelfde dag jarig zijn?

A: minder dan 10%

B: 10%-30%

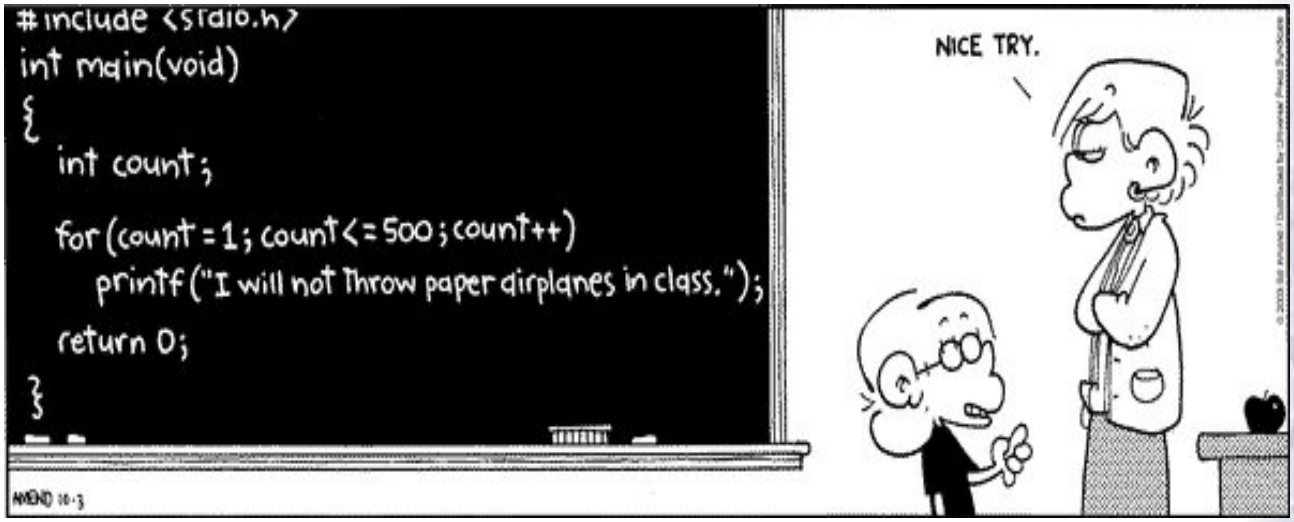
C: 30%-50%

D: 50%-75%

E: meer dan 75%

For of while ?

- for-lussen:
  - aantal iteraties ligt **vast** op ogenblik dat lus gestart wordt
  - overlopen van elementen van collectie-object (iterator)
- while-lussen:
  - elke iteratie heeft een geassocieerde **voorwaarde** die bepaalt of een nieuwe iteratie zal opgestart worden



UNIVERSITEIT GENT

the birthday paradox

opgave

Bepaal de kans dat in een groep van *n* personen minstens twee personen op dezelfde dag verjaren.

Verjaardagsparadox

$$q_n = \frac{365}{365} \cdot \frac{364}{365} \cdot \frac{363}{365} \cdot \ldots \cdot \frac{365 - n + 1}{365}$$

paradox1.py

```
for n in range(5, 76, 5):
    # kans berekenen
    qn = 1.0
    for m in range(n):
        qn *= (365 - m) / 365          # reële deling

    # kans uitschrijven
    print(n, 1 - qn)
```

UNIVERSITEIT GENT

Verjaardagsparadox

$$q_n = \frac{365}{365} \cdot \frac{364}{365} \cdot \frac{363}{365} \cdot \ldots \cdot \frac{365 - n + 1}{365}$$

paradox2.py

```
qn = 1.0
for n in range(2, 76):
    # kans berekenen
    qn *= (365 - n + 1) / 365       # reële deling

    # kans uitschrijven
    if not n % 5:
        print(n, 1 - qn)
```

UNIVERSITEIT GENT

Verjaardagsparadox

$$q_n = \frac{365}{365} \cdot \frac{364}{365} \cdot \frac{363}{365} \cdot \ldots \cdot \frac{365 - n + 1}{365}$$

paradox_grafisch.py

```
import pylab
# grafische voorstelling opbouwen
x, y, qn = [], [], 1.0
for n in range(365):
    # kans berekenen
    qn *= (365 - n) / 365          # reële deling

    # samplepunt toevoegen
    x.append(n + 1)                # x-waarde samplepunt
    y.append(1 - qn)               # y-waarde samplepunt

# grafische voorstelling weergeven
pylab.plot(x, y, 'b-')             # functie plotten
pylab.xlim([0, 365])               # bereik x-as vastleggen
pylab.show()                       # plot weergeven
```

matplotlib

UNIVERSITEIT GENT

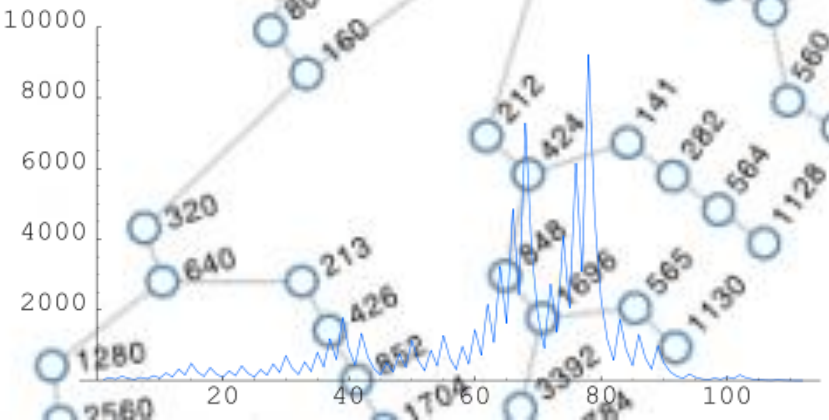
Vermoeden van Collatz

$$f(n) = \begin{cases} n/2 & \text{als } n \text{ even} \\ 3n+1 & \text{als } n \text{ oneven} \end{cases}$$

hagelsteen-reeks:

$$a_i = \begin{cases} n & \text{voor } i = 0 \\ f(a_{i-1}) & \text{voor } i > 0 \end{cases}$$

27, 82, 41, 124, 62, 31, 94, 47, 142, 71, 214, 107, 322, 161, 484, 242, 121, 364, 182, 91, 274, 137, 412, 206, 103, 310, 155, 466, 233, 700, 350, 175, 526, 263, 790, 395, 1186, 593, 1780, 890, 445, 1336, 668, 334, 167, 502, 251, 754, 377, 1132, 566, 283, 850, 425, 1276, 638, 319, 958, 479, 1438, 719, 2158, 1079, 3238, 1619, 4858, 2429, 7288, 3644, 1822, 911, 2734, 1367, 4102, 2051, 6154, 3077, **9232**, 4616, 2308, 1154, 577, 1732, 866, 433, 1300, 650, 325, 976, 488, 244, 122, 61, 184, 92, 46, 23, 70, 35, 106, 53, 160, 80, 40, 20, 10, 5, 16, 8, 4, 2, **1**



Vermoeden van Collatz

$$f(n) = \begin{cases} n/2 & \text{als } n \text{ even} \\ 3n+1 & \text{als } n \text{ oneven} \end{cases}$$

collatz.py

```
gevallen = int(input())
for geval in range(gevallen):
    # startwaarde inlezen en cyclelengte initialiseren op 1
    n, cyclelengte = int(input()), 1
    # lengte hagelsteenreeks bepalen
    while n != 1:
        if n % 2:                # n oneven
            n = 3 * n + 1
        else:                    # n even
            n //= 2
        cyclelengte += 1
    # lengte hagelsteenreeks uitschrijven
    print(cyclelengte)
```

hagelsteenreeks:

$$a_i = \begin{cases} n & \text{voor } i = 0 \\ f(a_{i-1}) & \text{voor } i > 0 \end{cases}$$

UNIVERSITEIT GENT

Vermoeden van Collatz

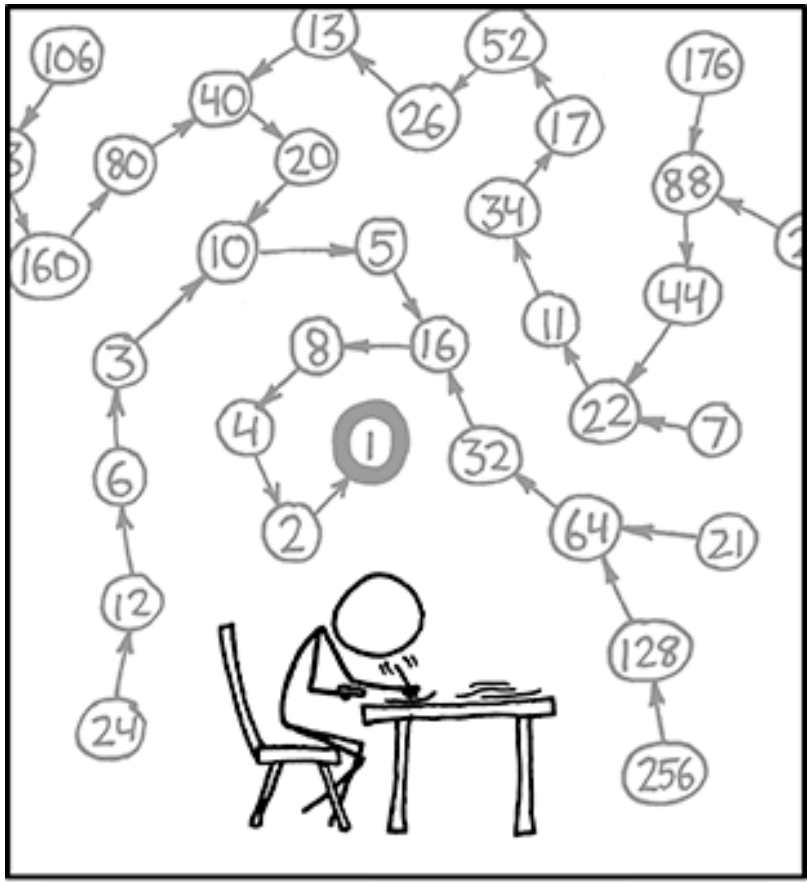
collatz_grafisch.py

```
m = int(input('Geef een waarde voor n: '))

# hagelsteenreeks bepalen
n = m
hagelsteenreeks = [n]
while n != 1:
    if n % 2:                # n oneven
        n = 3 * n + 1
    else:                    # n even
        n //= 2
    hagelsteenreeks.append(n)

# hagelsteenreeks grafisch weergeven
import pylab
pylab.plot(range(1, len(hagelsteenreeks) + 1), hagelsteenreeks)
pylab.title(f'hagelsteenreeks voor $n = {m}$')
pylab.xlim(1, len(hagelsteenreeks))
pylab.xlabel('$i$')
pylab.ylabel('$a_i$')
pylab.show()
```

matplotlib

UNIVERSITEIT GENT

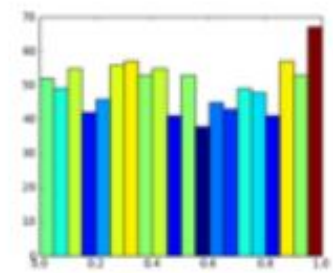
What's the use ?

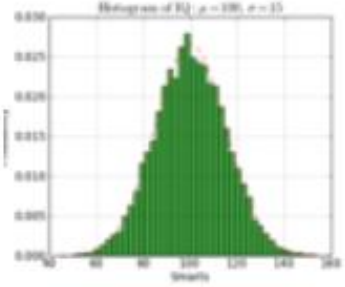


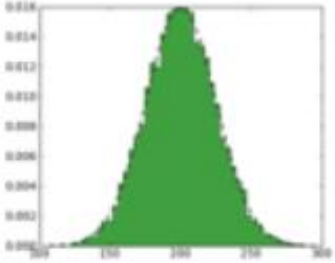
THE COLLATZ CONJECTURE STATES THAT IF YOU PICK A NUMBER, AND IF IT'S EVEN DIVIDE IT BY TWO AND IF IT'S ODD MULTIPLY IT BY THREE AND ADD ONE, AND YOU REPEAT THIS PROCEDURE LONG ENOUGH, EVENTUALLY YOUR FRIENDS WILL STOP CALLING TO SEE IF YOU WANT TO HANG OUT.

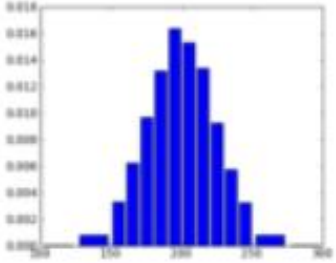
UNIVERSITEIT GENT

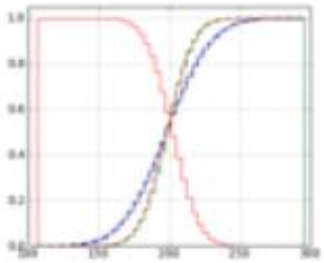
Matplotlib gallery

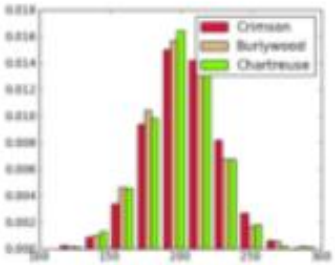


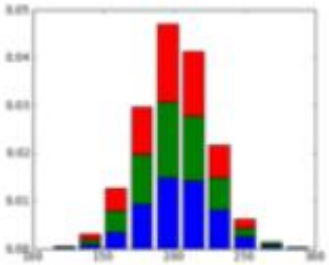


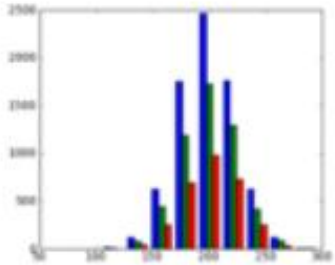


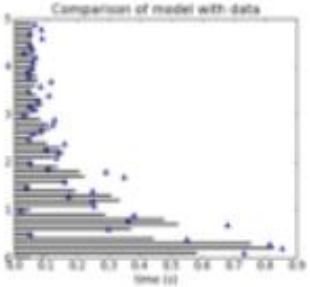


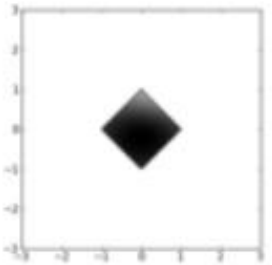


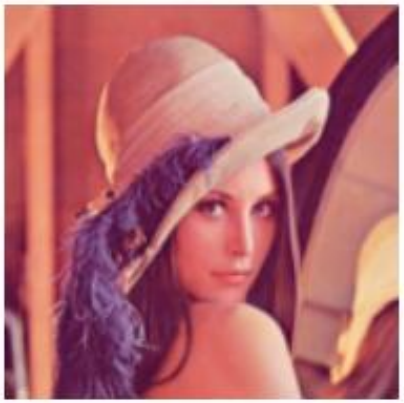


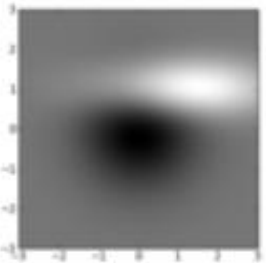


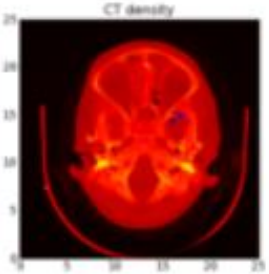


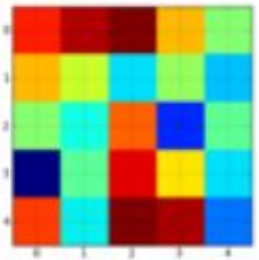


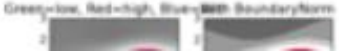


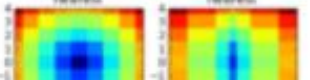




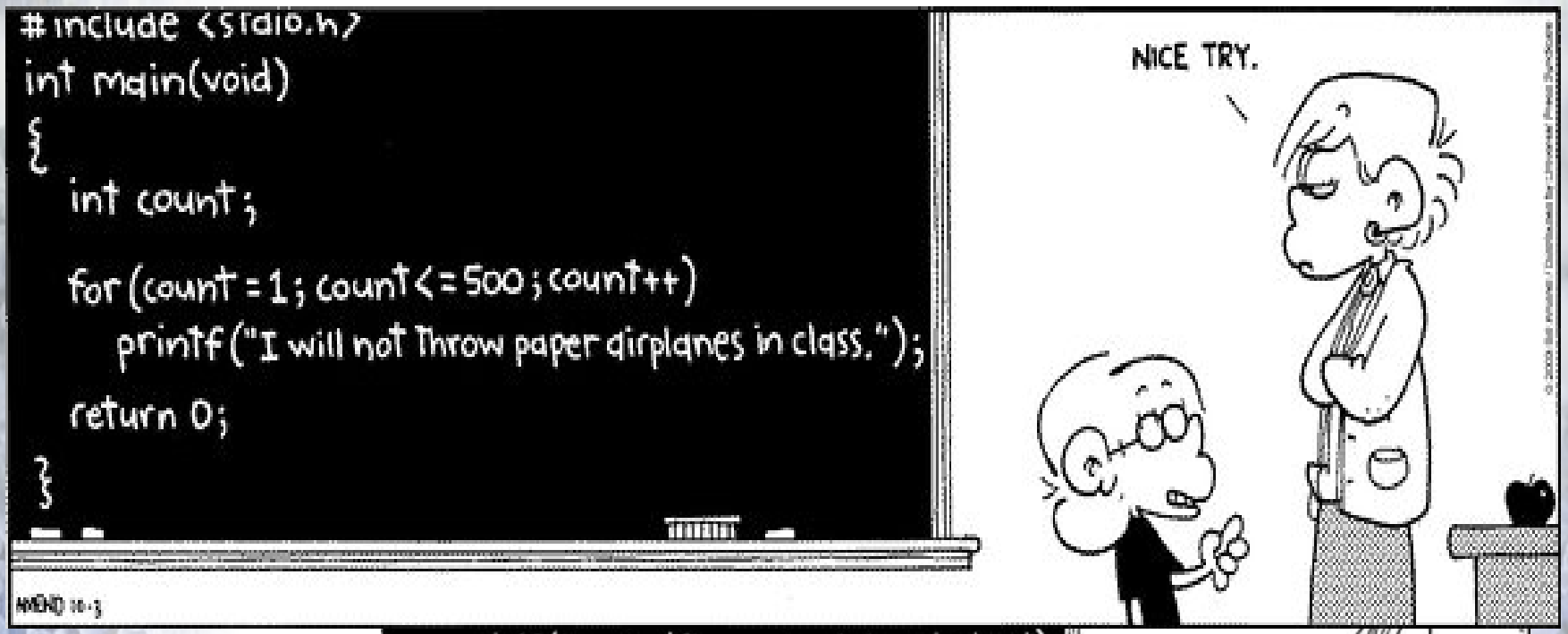


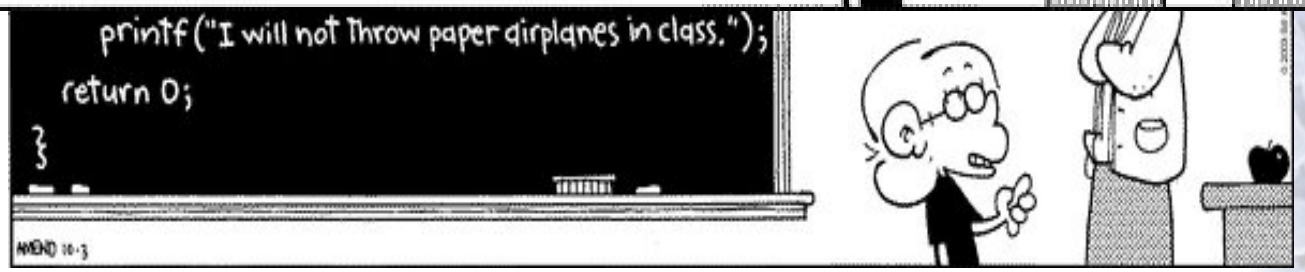






For of while ?

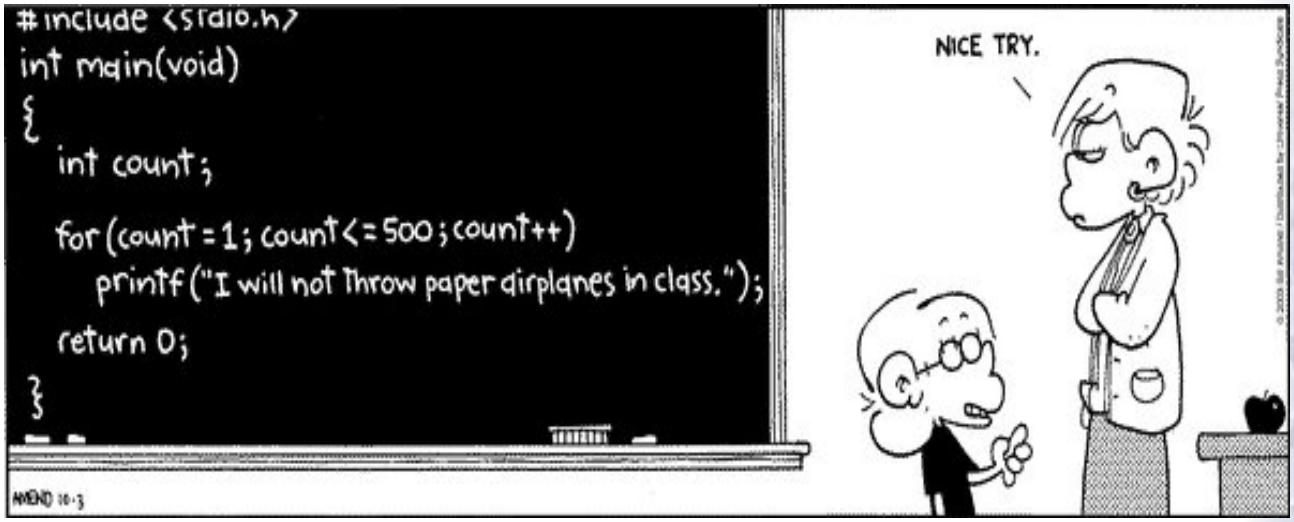




UNIVERSITEIT GENT

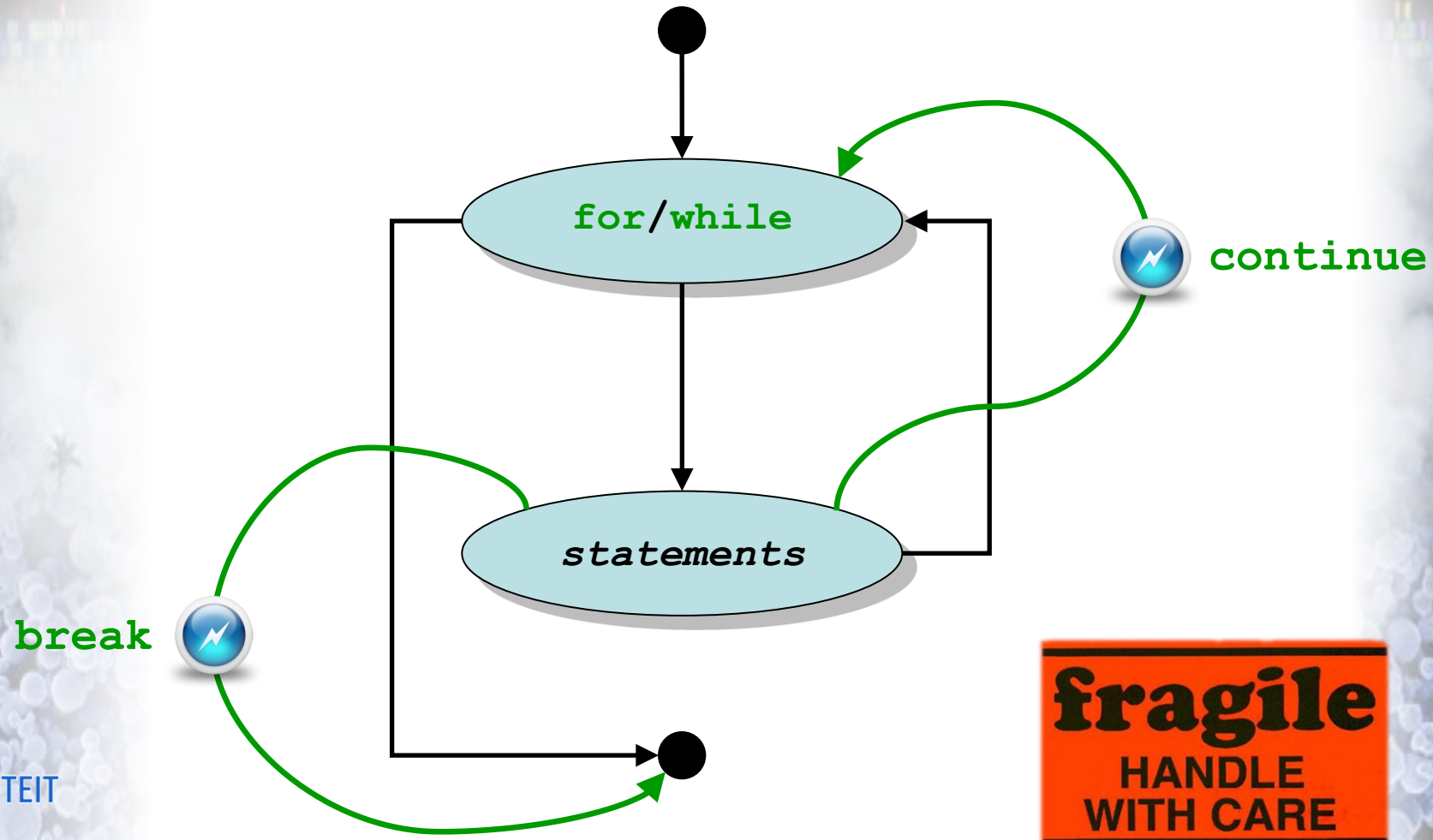
For of while ?

- for-lussen:
  - aantal iteraties ligt **vast** op ogenblik dat lus gestart wordt
  - overlopen van elementen van collectie-object (iterator)
- while-lussen:
  - elke iteratie heeft een geassocieerde **voorwaarde** die bepaalt of een nieuwe iteratie zal opgestart worden



UNIVERSITEIT GENT

Kortsluiting van lussen



UNIVERSITEIT GENT

Getallen raden

raden1.py

```
# willekeurig getal kiezen tussen 1 en 10 (grenzen inbegrepen)
import random
getal = random.randint(1, 10)

# geef speler drie beurten om getal te raden
geraden = False
for beurten in range(3, 0, -1):

    # vraag speler om een gok te doen
    gok = int(input(f'Doe een gok (nog {beurten} beurten): '))

    # bepaal of speler het correcte getal geraden heeft
    if gok == getal:
        print(f'Getal geraden in {3 - beurten + 1} beurten!')
        geraden = True
        break
    else:
        print('Niet correct!')

# correct getal uitschrijven indien speler het niet geraden heeft
if not geraden:
    print(f'Het getal was {getal}.')
```

UNIVERSITEIT GENT

Getallen raden

raden1.py

```
# willekeurig getal kiezen tussen 1 en 10 (grenzen inbegrepen)
import random
getal = random.randint(1, 10)

# geef speler drie beurten om getal te raden
geraden = False
for beurten in range(3, 0, -1):

    # vraag speler om een gok te doen
    gok = int(input(f'Doe een gok (nog {beurten} beurten): '))

    # bepaal of speler het correcte getal geraden heeft
    if gok == getal:
        print(f'Getal geraden in {3 - beurten + 1} beurten!')
        geraden = True
        break
    else:
        print('Niet correct!')

# correct getal uitschrijven indien speler het niet geraden heeft
if not geraden:
    print(f'Het getal was {getal}.')
```

UNIVERSITEIT GENT

Getallen raden

raden2.py

```
# willekeurig getal kiezen tussen 1 en 10 (grenzen inbegrepen)
import random
getal = random.randint(1, 10)

# geef speler drie beurten om getal te raden
beurten, geraden = 3, False
while beurten and not geraden:

    # vraag speler om een gok te doen
    gok = int(input(f'Doe een gok (nog {beurten} beurten): '))
    beurten -= 1

    # bepaal of speler het correcte getal geraden heeft
    if gok == getal:
        print(f'Getal geraden in {3 - beurten} beurten!')
        geraden = True
    else:
        print('Niet correct!')

# correct getal uitschrijven indien speler het niet geraden heeft
if not geraden:
    print(f'Het getal was {getal}.')
```

UNIVERSITEIT GENT

Getallen raden

raden3.py

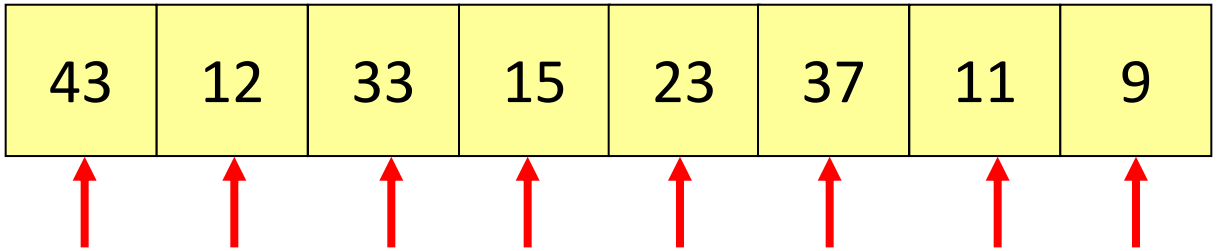
```
# tel aantal pogingen nodig om getal tussen 1 en 100 te raden

# willekeurig getal kiezen tussen 1 en 100 (grenzen inbegrepen)
import random
getal = random.randint(1, 100)

# laat speler raden tot het getal gevonden is
beurten, gok = 0, 0
while gok != getal:

    # vraag speler om nog een gok te wagen
    beurten += 1
    gok = int(input('Wat is je volgende gok? '))

    # geef hint aan gebruiker over gezochte getal
    if gok < getal:
        print('Hoger!!')
    elif gok > getal:
        print('Lager!!')

# aantal benodigde beurten uitschrijven
print(f'Getal geraden na {beurten} beurten.')
```

UNIVERSITEIT GENT

Reeks doorlopen

Schrijf een programma dat het minimum en het gemiddelde van een reeks met *n* getallen bepaalt.

```
$ python gemiddelde.py < invoer.txt
minimum: 9
gemiddelde: 22.875
$
```



min = 43 min = 12 min = 12 min = 12 min = 12 min = 11 min = 9

som = 43 som = 55 som = 88 som = 103 som = 126 som = 163 som = 174 = 183

Reeks doorlopen

gemiddelde.py

```
# bepaal minimum en gemiddelde van reeks getallen

# lengte van reeks opvragen
n = int(input('Geef het aantal getallen in de reeks: '))

# lopende variabelen initialiseren
min = int(input('Geef het eerste getal: '))
som = min

# reeks doorlopen
for i in range(n - 1):
    getal = int(input('Geef het volgende getal: '))
    som += getal
    if getal < min:
        min = getal

# minimum en gemiddelde uitschrijven
print('minimum:', min)
print('gemiddelde:', float(som) / n)
```

UNIVERSITEIT GENT

Reekssom bepalen

Schrijf een programma dat de som van de kwadraten van de eerste *n* natuurlijke getallen bepaalt.

$$\sum_{i=1}^{n} i^2$$

1	4	9	16	25	36	49	64

UNIVERSITEIT GENT

Reekssom bepalen

Schrijf een programma dat de som van de kwadraten van de eerste *n* natuurlijke getallen bepaalt.

$$\sum_{i=1}^{n} i^2$$

gesloten_reekssom.py

```
# aantal getallen opvragen
n = int(input('Geef een strikt positief geheel getal: '))

# som van kwadraten bepalen
som = 0
for i in range(1, n + 1):
    som += i ** 2

# som uitschrijven
print(f'De som van de eerste {n} kwadraten is {som}.')
```

UNIVERSITEIT GENT

Reekssom bepalen

Schrijf een programma dat de som bepaalt van een reeks getallen die wordt afgesloten met lege invoer.

117	44	12	87	48	94	38	

UNIVERSITEIT GENT

Reekssom bepalen

Schrijf een programma dat de som bepaalt van een reeks getallen die wordt afgesloten met lege invoer.

open_reekssom.py

```
# reekssom bepalen
som = 0
invoer = input('Geef volgend getal ([enter] om te stoppen): ')
while invoer:
    som += int(invoer)
    invoer = input('Geef volgend getal ([enter] om te stoppen): ')

# reekssom uitschrijven
print(f'De som van de getallen is {som}.')
```

UNIVERSITEIT GENT

Geneste lussen

Schrijf een programma dat de uitkomst berekent van volgende dubbele reekssom.

$$\sum_{i=1}^{m} \sum_{j=1}^{n} \frac{1}{i^2 + j^2}$$

UNIVERSITEIT GENT

Geneste lussen

Schrijf een programma dat de uitkomst berekent van volgende dubbele reekssom.

dubbele_reekssom.py

```
# limieten opvragen
m = int(input('Geef de waarde van m: '))
n = int(input('Geef de waarde van n: '))

# reekssom bepalen
som = 0.0
for i in range(1, m + 1):
    for j in range(1, n + 1):
        som += 1 / (i ** 2 + j ** 2)

# reekssom uitschrijven
print(f'De reekssom is {som:.6f}.')
```

UNIVERSITEIT GENT

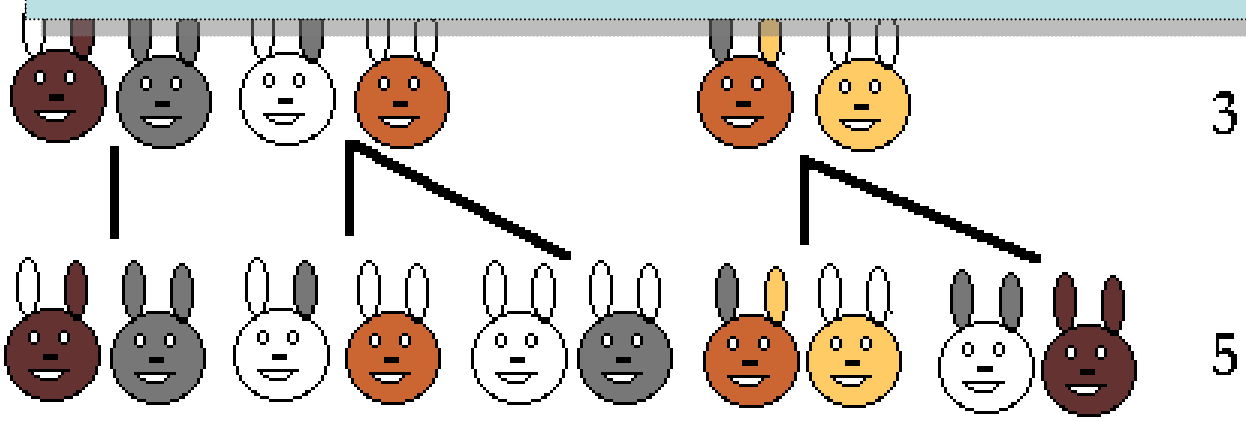
Fibonacci

 # paren

Veronderstel dat een pasgeboren paar konijnen, één mannetje en één vrouwtje, in een veld geplaatst worden. Konijnen zijn in staat om te paren op de leeftijd van één maand, zodat op het einde van de tweede maand een vrouwtje een ander paar konijnen kan werpen.

Veronderstel ook dat onze konijnen nooit doodgaan, en dat elk vrouwtje **elke maand** telkens een nieuw paar konijnen ter wereld brengt vanaf de tweede maand. De vraag die Fibonacci hierbij lanceerde was…

Hoeveel paar konijnen zal er na één jaar in het veld zitten?



UNIVERSITEIT GENT

Fibonacci

Schrijf een programma dat de eerste *n* termen uit de rij van Fibonacci uitschrijft

$$\begin{cases} F_0 = 0 \\ F_1 = 1 \\ F_n = F_{n-1} + F_{n-2} \quad (n > 1) \end{cases}$$

fibonacci.py

```
# waarde voor n opvragen
n = int(input('Geef de waarde voor n: '))

# eerste termen initialiseren en uitschrijven
f0, f1 = 0, 1
print(f1)

# volgende termen berekenen en uitschrijven
for i in range(n - 1):
    f0, f1 = f1, f0 + f1
    print(f1)
```

UNIVERSITEIT GENT

Fibonacci in de natuur

zonnebloemen hebben bloemblaadjes in spiralen van 34 en 55 rond de buitenkant

UNIVERSITEIT GENT

Huiswerk (volgende les)

- lees handboek
  - hoofdstuk 4 (strings)
- lees omschrijving van voorbeeldopgaven (reeks 4)
  - Caesarrotatie
  - Lettersoep
  - Tandenstokers

Huiswerk (werkcolleges)

- afwerken opgelegde oefeningen reeks 03 (deadline dinsdag 14 oktober 2025, 22:00)
  - ISBN (demo)
  - Chaos
  - Duitse tanks
  - Apen en kokosnoten
  - Pythagorese drietallen
  - Lifters

Vragen of opmerkingen?

UNIVERSITEIT GENT

The sky is the limit …

"If you don't know where you are going,
any road will get you there."

— Lewis Carroll

UNIVERSITEIT GENT