ing. Jurgen Nijs

Python voor 3de graad leerkrachtenbundel

UHASSELT
KNOWLEDGE IN ACTION

Inleiding

In de cursus zijn opdrachten en oefeningen opgenomen.

Opdrachten dienen om nieuwe leerstof aan te reiken, terwijl oefeningen dienen om aangeleerde oefeningen in te oefenen.

Deze leerkrachtenhandleiding bevat alle oplossingen van de oefeningen. Van de opdrachten is de code opgenomen indien deze relevant is.

Op de ondersteunende website kan je met je stamboeknummer toegang aanvragen tot de lerarenpagina en daar alle code downloaden.

Je kan de code van individuële code downloaden door op de naam van de oefening of opdracht te klikken of een zip-bestand te downloaden met alle opdrachten en oefeningen.

De benaming van de bestanden is als volgt

<module nummer> - oefening/opdracht <nummer>.py

Zo zal de bestandsnaam van ***oefening 2.13*** zijn: **2 – oefening 13.py**

En de bestandsnaam van ***opdracht 4.5***: **4 – opdracht 5.py**

De code is één van de mogelijke oplossingen.

Soms wordt er een alternatieve code aangeboden. Dit is code waarin wat programmeurstrucjes zijn opgenomen. Ook is er hier en daar ook een uitbreiding voorzien. Die zijn niet in de cursus opgenomen, wel in de leerkrachtenbundel. Hiermee kan je differentiatie regelen voor sterke en snelle leerlingen.

Om een verschil te maken tussen code gegeven in het handboek, oplossingen van opdrachten en oplossingen van oefeningen gebruiken we volgende achtergronden

```
Op deze manier wordt code aangegeven die zo in de cursus staat
```

```
Als het een oplossing is van een opdracht, dan gebruiken we een blauwe achtergrond.
```

```
En met een gele achtergrond wordt oplossingen van oefeningen aangeduid.
```

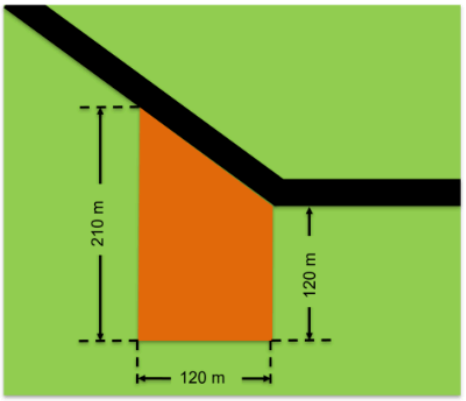
Veel succes met de cursus, feedback is steeds welkom.

UHASSELT

Module 1: Sequentie

Opdrachten

Opdracht 1.1:



Een boer heeft een akker in de vorm van een rechthoekig trapezium. De afmetingen van de akker zijn aangeduid op de bijgevoegde figuur. Hij wilt de akker omvormen tot een boomgaard. Hij leest op de website www.landleven.nl dat een appelboom een oppervlakte van 8 x 8 m nodig heeft. Hoeveel bomen moet de boer aankopen?

Neem een blad en papier en bereken op een gestructureerde manier het aantal bomen dat de boer moet aankopen.

$\frac{120+210}{2} \cdot 120 = 19\,800$

$\frac{19800}{8 \cdot 8} = 309,375 \rightarrow 310$

Opdracht 1.2:

Veronderstel dat je broer of zus een machine is die je moet programmeren om een boterham met choco te smeren.

Maak eens (in je eigen woorden) een stappenplan waarmee je broer of zus deze taak tot een goed einde kan brengen.

Bekijk daarna het filmpje: ***Leer programmeren met boterham met choco***.

Zou jou stappenplan wel tot een goed resultaat geleid hebben?

Geen oplossing gegeven

Opdracht 1.3:

Schrijf eens in je eigen woorden op welke stappen je ondernomen hebt om het aantal boompjes te berekenen dat de boer moet aankopen. Jij bent geen machine, dus je mag acties zoals pen en papier nemen overslaan.

Bereken het oppervlakte van het veld: $\frac{H+h}{2} \cdot b = \frac{210+120}{2} \cdot 120 = 19800$

Bereken het oppervlakte nodig voor 1 boom $z \cdot z = 8 \cdot 8 = 64$

Deel oppervlakte van het veld door oppervlakte boom: $\frac{19800}{64} = 309,375$

Rond af naar beneden □ 309

Opdracht 1.4:

Vul onderstaande lijst aan met de symbolen die je denkt dat overeenkomen met de bewerking.

⇨ Optelling:

⇨ Aftrekking:

⇨ Vermenigvuldiging:

⇨ Deling:

⇨ Machtsverheffing:

Controleer of jouw symbolen juist zijn door onderstaande bewerkingen uit te voeren in de shell. Je typt gewoon de bewerking en in plaats van het gelijkheidsteken druk je op *ENTER*..

⇨ $8 + 2$

⇨ $8 - 2$

⇨ $8 \cdot 2$

⇨ $\frac{8}{2}$

⇨ 8^2

Optelling: +

Aftrekking: −

Vermenigvuldiging: *

Deling: /

Machtsverheffing: **

```
>>> 8+2
10
>>> 8-2
6
>>> 8*2
16
>>> 8/2
4.0
>>> 8**2
64
>>>
```

Opdracht 1.5:

Controleer je algoritme van opdracht 1.3 door het stap voor stap uit te voeren in de shell.

Je geeft één voor één de bewerkingen in. Sluit de bewerkingen af door op *ENTER* te drukken.

```
>>> (210+120)/2*120
19800.0
>>> 8*8
64
>>> 19800/64
309.375
>>>
```

Of

```
>>> (210+120)/2*120
19800.0
>>> 8*8
64
>>> 19800//64
309
>>>
```

Opdracht 1.6

Onderzoek welk type variabele (integer of float) je krijgt bij het uitvoeren van een deling (/) en een gehele deling (//). Probeer alle combinaties uit van integers en floats.

```
>>> 11//3
3
>>> 11.0//3
3.0
>>> 11//3.0
3.0
>>> 11.0//3.0
3.0
>>>
```

Je krijgt enkel als resultaat een geheel getal als zowel de deler als het deeltal een geheel getal zijn.

In alle andere gevallen krijg je een reëel getal (float)

Opdracht 1.7

Bekijk onderstaande instructies. Noteer wat je verwacht dat op de stippellijnen onder elke print-instructie op het scherm getoond gaat worden.

Controleer je antwoorden door de code in de shell in te typen.

```
>>> a = 4.8
>>> b = 4
>>> c = -2
>>> print(a/b)

>>> print(a//b)

>>> print(b/c)

>>> print(b//c)

>>> print(a/c)

>>> print(a//c)
```

```
>>> a = 4.8
>>> b = 4
>>> c = -2
>>> print(a/b)
 1.2
>>> print(a//b)
 1.0
>>> print(b/c)
 -2.0
>>> print(b//c)
 -2
>>> print(a/c)
 -2.4
>>> print(a//c)
 -3.0
>>>
```

Bij een gehele deling wordt altijd naar beneden afgerond.

Opdracht 1.8:

Python is een zeer leesbare taal. Wanneer je alle Engelse instructies zou vertalen naar het Nederlands, dan kan je al een goed idee krijgen wat het script gaat doen.

Bestudeer onderstaande scripts.

Wat denk je dat er gaat gebeuren?

Typ de code over en controleer je antwoord.

Script 1

```
getal=input("Geef getal in:")
print("Het ingegeven getal was",getal)
```

Script 2

```
getal=input("Geef getal in:")
dubbel_getal=2*getal
print("Het ingegeven getal was",getal)
print("Het dubbele van het getal is", dubbel_getal)
```

Script 1:

Er zal op het scherm gevraagd worden om een getal in te geven en dit getal wordt daarna weergegeven.

Script 2:

Het eerste gedeelte is hetzelfde gebleven, maar leerlingen verwachten dat het script *het dubbele gaat geven.*

Als ze het getal 12 ingeven, verwachten ze 24 maar input levert een string op, dus ze zien 1212

Doel is om de leerling te wijzen op het feit dat een input in Python 3 steeds een integer is.

Opdracht 1.9

Bekijk onderstaande instructies. Noteer wat je verwacht dat op de stippellijnen onder elke print-instructie op het scherm getoond gaat worden.

Controleer je antwoorden door de code in de shell in te typen.

```
>>> a = 9
>>> b = 2
>>> c = -2
>>> print(a/b)
.....
>>> print(a//b)
.....
>>> print(int(a/b))
.....
>>> print(a/c)
.....
>>> print(a//c)
.....
>>> print(int(a/c))
.....
>>>
```

```
>>> a = 9
>>> b = 2
>>> c = -2
>>> print(a/b)
 4.5
>>> print(a//b)
 4
>>> print(int(a/b))
 4
>>> print(a/c)
 -4.5
>>> print(a//c)
 -5
>>> print(int(a/c))
 -4
>>>
```

Doel van deze oefening is aantonen dat als het resultaat een positief getal is, er geen verschil is tussen de gehele deling en **int( )**. Als het resultaat negatief is, wel. De gehele deling rond af naar beneden, en **int( )** kapt af aan het decimale punt.

Opdracht 1.10

Pas onderstaande code aan zodat het script het dubbele van het getal berekent en dit als output geeft.

```
getal=input("Geef getal in:")
dubbel_getal=2*getal
print("Het ingegeven getal was",getal)
print("Het dubbele van het getal is", dubbel_getal)
```

Een voorbeeld van een mogelijke output wordt hieronder getoond.

```
>>> %Run -c $EDITOR_CONTENT
  Geef getal in:13
  Het ingegeven getal was 13
  Het dubbele van het getal is 26
>>>
```

Oplossing

```
getal=input("Geef getal in:")
getal = int(getal)
dubbel_getal=2*getal
print("Het ingegeven getal was",getal)
print("Het dubbele van het getal is", dubbel_getal)
```

of

```
getal=int(input("Geef getal in:"))
dubbel_getal=2*getal
print("Het ingegeven getal was",getal)
print("Het dubbele van het getal is", dubbel_getal)
```

Opdracht 1.11:

Onderstaande code is een mogelijke oplossing voor opdracht 1.1.
Pas deze, of je eigen code, aan zodat de gebruiker verschillende zijden kan ingeven zonder er iets aan de code moet veranderd worden.

```
# Aantal bomen berekenen

# Gegevens
lange_zijde = 210
korte_zijde = 120
breedte = 120

oppervlakte_boom = 8*8

# Oppervlakte veld berekenen
oppervlakte_veld = (lange_zijde+korte_zijde)/2*breedte

# Aantal bomen berekenen
aantal_bomen = oppervlakte_veld/oppervlakte_boom

# Output
print("Gegevens:")
print("Lange zijde:",lange_zijde)
print("Korte zijde:",korte_zijde)
print("Breedte:",breedte)
print()
print("Resultaat:")
print("Aantal bomen aan te kopen:",aantal_bomen)
```

Oplossing:

```
""" Opdracht 1.11
    Aantal bomen berekenen
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                        """

# Gebruikers interface
print("Geef de afmetingen van het veld: ")

# Gegevens veld
lange_zijde = int(input("  Lengte lange zijde: "))
korte_zijde = int(input("  Lengte korte zijde: "))
breedte = int(input("  Breedte: "))
print()

# Gegeven boom
print("Geef zijde op van het oppervlakte dat 1 boom nodig heeft")
zijde = int(input("   zijde oppervlakte boom: "))
oppervlakte_boom = zijde*zijde

# Oppervlakte veld berekenen
oppervlakte_veld = (lange_zijde+korte_zijde)/2*breedte

# Aantal bomen berekenen
aantal_bomen = oppervlakte_veld/oppervlakte_boom
```

```
# Output
print("Gegevens:")
print("Lange zijde:",lange_zijde)
print("Korte zijde:",korte_zijde)
print("Breedte:",breedte)
print()
print("Resultaat:")
print("Aantal bomen aan te kopen:",aantal_bomen)
```

Alternatief kan je ook rechtstreeks het oppervlakte berekenen door de input te kwadrateren

```
# Gegeven boom
print("Geef zijde op van het oppervlakte dat 1 boom nodig heeft")
oppervlakte_boom = int(input("   zijde oppervlakte boom: "))**2

# Oppervlakte veld berekenen
```

Opdracht 1.12

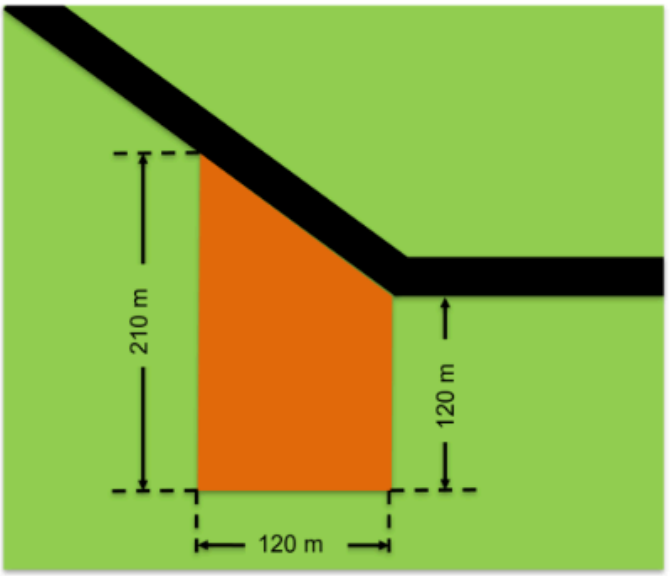
Onze boer is heel gelukkig met zijn boomgaard, en hij begint al aardig wat appelen op te leveren. Spijtig genoeg ligt zijn boomgaard langs een toeristisch fietspad. Zeer veel fietsers stoppen voor een appel, en gaan zelfs met volle fietstassen naar huis.

Hij besluit daarom langs de weg een omheining te plaatsen.

In de omheining moet een opening zijn van 4 m voor een poort, en de omheining wordt gemaakt van metalen draadhekken zoals op de foto is aangegeven. De lengte van 1 draadhek is 2,03 m.

Maak een algoritme en zet dit om naar een script dat flexibel kan ingezet worden om het nodige aantal draadhekken en palen te berekenen.

> Tip:
>
> Je script van opdracht 1.11 kan je gebruiken als start-punt.



Oplossing:

```
""" Opdracht 1.12
```

```
    Aantal hekken en palen berekenen
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

# Titel op scherm (wat doet het script)

print("Aantal palen en hekken berekenen")
print()

# Gegevens ingeven

lange_zijde = float(input("Lengte lange zijde: "))
korte_zijde = float(input("Lengte korte zijde: "))
breedte_akker = float(input("Breedte van de akker: "))

breedte_poort = float(input("Breedte van de poort: "))
lengte_element = float(input("Lengte van 1 draadhek: "))


# Berekenen van de lengte langs de weg met Phytagoras

lengte_verschil = lange_zijde-korte_zijde

lengte_weg = (lengte_verschil**2+breedte_akker**2)**.5

# Berekenen aantal hekken

aantal_hekken = int(lengte_weg//lengte_element + 1)
#int om een geheel getal te krijgen
# +1 extra element nodig dat je moet inkorten

# Berekenen aantal palen

aantal_palen = aantal_hekken + 2

# Links en recht van de poort een hek. Telkens 1 extra paal nodig

#uitvoer

print("Aan te kopen:")
print("  palen: ",aantal_palen)
print("  hekken:",aantal_hekken)
```

Opdracht 1.13

Schrijf een script dat aan de gebruiker vraagt om het volume van een kubus in te geven. Het script berekent dan de zijde.

```
Bereken zijde van een kubus uit het volume
Volume van de kubus: 144
De zijde is 5.241482788417793

>>>
```

Oplossing:

```
""" Opdracht 1.13
    Bereken zijde kubus als volume gegeven is
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

# Titel op scherm en input

print("Bereken de zijde van de kubus uit het volume:")
volume = float(input("Volume van de kubus: "))

# Berekening

zijde = volume ** (1/3) # Tot de macht 1/3 is de derdemachtswortel

print("De zijde is",zijde)
```

Oefeningen

Oefening 1.1

Schrijf een script dat aan de gebruiker vraagt om 3 gehele getallen in te geven. Het script berekent dan de som van de getallen en geeft dit als output op het scherm.

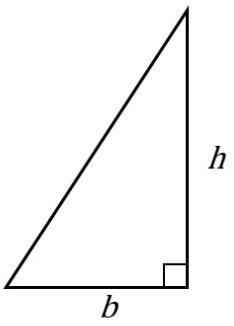
Voorbeeld:

```
>>> %Run -c $EDITOR_CONTENT
 Het eerste getal is: 12
 Het tweede getal is: 6
 Het derde getal is: -3
 De som van de drie getallen is: 15.0
>>>
```

```
""" Opdracht 1.13
    Bereken zijde kubus als volume gegeven is
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

getal1 = float(input("Het eerste getal is: "))
getal2 = float(input("Het tweede getal is: "))
getal3 = float(input("Het derde getal is: "))

# Berekenen: som
som = getal1 + getal2 + getal3

# Output
print("De som van de drie getallen is:",som)
```

Oefening 1.2

Bereken de oppervlakte van een rechthoekige driehoek. De gebruiker geeft de twee rechthoekszijden in en geeft als output de oppervlakte van de driehoek.



$$A = \frac{1}{2}bh$$

```
  Oppervlakte berekening rechthoekige driehoek
  --------------------------------------------
  Zijde: 4
  Hoogte: 2.5
  Oppervlakte A = 5.0

>>>
```

Voorbeeld:

```
""" Oefening 1.2
    Oppervlakte rechthoekige driehoek
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """
# Input
print("Oppervlakte berekenen rechthoekige driehoek")
print("-------------------------------------------")
zijde = float(input("Zijde: "))
hoogte = float(input("Hoogte: "))

# Berekeningen
opp =  zijde*hoogte/2

# Output
print("Oppervlakte A =",opp)
```

Oefening 1.3

Gegeven twee tijdstippen op dezelfde dag
(bv. 12:34:56 en 21:43:07).

Het tweede tijdstip valt later dan het eerste tijdstip (laatste tijdstip 23:59:59)

Schrijf een script dat berekent hoeveel seconden er tussen beide tijdstippen zitten en geeft het resultaat op het scherm.

Voorbeeld:

```
  Tijdsduur tussen 2 tijdstippen
  ------------------------------
  Tijdstip 1:
     uur: 8
     minuten: 12
     seconden: 16
  Tijdstip 2:
     uur: 12
     minuten: 9
     seconden: 51
  Tijdsduur in seconden = 14255

>>>
```

```
""" Oefening 1.3
    Seconden tussen 2 tijdstippen
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """
# Input
print("Tijdsduur tussen 2 tijdstippen")
```

```
print("------------------------------")
print("Tijdstip 1:")
uur1 = int(input("   uur: "))
minuut1 = int(input("   minuten: "))
second1 = int(input("   seconden: "))

print("Tijdstip 2:")
uur2 = int(input("   uur: "))
minuut2 = int(input("   minuten: "))
second2 = int(input("   seconden: "))

# Berekeningen
# Omzetten beide tijden omzetten naar seconden
# Beide tijden van elkaar aftrekken

tijd1 = uur1*3600 + minuut1*60 + second1
tijd2 = uur2*3600 + minuut2*60 + second2

tijdsduur = tijd2-tijd1


# Output
print("Tijdsduur in seconden =",tijdsduur)
```

Oefening 1.4

Bereken de schuine zijde van een rechthoekige driehoek als de twee rechthoekszijden gegeven zijn.

Voorbeeld:

```
Schuine zijde rechthoekige driehoek
Rechthoekszijde 1: 3
Rechthoekszijde 2: 4
De schuine zijde is 5.0

>>>
```

```
""" Oefening 1.4
    Schuine zijde rechthoekige driehoek
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """
# Input
print("Schuine zijde rechthoekige driehoek")
zijde1 = float(input("Rechthoekszijde 1:"))
zijde2 = float(input("Rechthoekszijde 1:"))

# Berekeningen
# Pythagoras

hypothenusa = (zijde1**2+zijde2**2)**.5


# Output
print("De schuine zij is",hypothenusa)
```

Oefening 1.5: Hoekpunten van een driehoek

Schrijf een script waarin de gebruiker de coördinaten ingeeft van de hoekpunten van een driehoek waarvan geen enkele zijde verticaal (evenwijdig met de y-as) ligt.

Het script berekent dan de oppervlakte van de driehoek

```
""" Oefening 1.5
    Oppervlakte driehoek - geen enkele zijde verticaal
    coördinaten van punten gegeven
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """
# Input
print("Oppervlakte van driehoek")
print("------------------------")
print("Geef de coördinaten van een hoekpunten van de driehoek")
print("Geen enkele zijde mag verticaal liggen")
print("Je mag dus niet tweemaal dezelfde y-coördinaat hebben")
print();
print("Hoekpunt 1:")
x1 = float(input(" x: "))
y1 = float(input(" y: "))
print("Hoekpunt 2:")
x2 = float(input(" x: "))
y2 = float(input(" y: "))
print("Hoekpunt 3:")
x3 = float(input(" x: "))
y3 = float(input(" y: "))


"""vergelijking rechte tussen punt 2 en punt 3
y = mx + q --> ax + by + c =0
y - y2 = m(x-x2)
mx - y -mx2+y2 = 0
a = m
b = -1
c = -mx2+y2 = q
"""
m = (y3-y2)/(x3-x2)
q = - m*x2+y2

# Afstand van punt tot rechte
hoogte = abs(m*x1-y1+q)/(m**2+1)**.5
# Afstand tussen punt 2 en punt 3
basis = ((x3-x2)**2+(y3-y2)**2)**.5

opp = hoogte*basis/2

# Output
print("De hoogte van de driehoek is",hoogte)
print("De bais van de driehoek is",basis)
print("Dit geeft een oppervlakte van",opp)
```

of deze alternatieve oplossing. Hierbij mogen de zijden wel verticaal liggen.

```
""" Oefening 1.5 (Alternatief)
```

```
    Oppervlakte driehoek - geen enkele zijde verticaal
    coördinaten van punten gegeven
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """
# Input
print("Oppervlakte van driehoek")
print("------------------------")
print("Geef de coördinaten van een hoekpunten van de driehoek")
print();
print("Hoekpunt 1:")
x1 = float(input(" x: "))
y1 = float(input(" y: "))
print("Hoekpunt 2:")
x2 = float(input(" x: "))
y2 = float(input(" y: "))
print("Hoekpunt 3:")
x3 = float(input(" x: "))
y3 = float(input(" y: "))

opp = 0.5 * abs(x1 * (y2 - y3) + x2 * (y3 - y1) + x3 * (y1 - y2))

# Output
print("De driehoek heeft een oppervlakte van",opp)
```

Oefening 1.6: Zijde van een kubus met afronding van resultaat

Schrijf een script dat aan de gebruiker vraagt om het volume van een kubus in te geven. Het script berekent dan de zijde en rond het resultaat af tot op 2 cijfers na de komma.

Je mag hier geen gebruik maken van de functie **round( )**.

Dit is een uitbreiding van **opdracht 1.13**

Voorbeeld:

```
  Bereken zijde van een kubus uit het volume
  Volume van de kubus: 142
  De zijde is 5.21

>>>
```

```
""" Oefening 1.6
    Zijde van kubus uit volume - afronden op 2 cijfers
    Uitbreiding opdracht 1.13
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

# Titel op scherm en input

print("Bereken de zijde van de kubus uit het volume:")
volume = float(input("Volume van de kubus: "))

# Berekening

zijde = volume ** (1/3) # Tot de macht 1/3 is de derdemachtswortel

""" Uitbreiding """
```

```
zijde = zijde * 100 + .5 # plus 0.5 om wiskundig te kunnen afronden
zijde = int(zijde)
zijde = zijde / 100

""" of in één formule

zijde = int(zijde * 100 + .5)/100

"""

print("De zijde is",zijde)
```

Oefening 1.7: Zijde van een kubus met afronding van resultaat

Zelfde oefening als oefening 1.6 maar vraag nu ook aan de gebruiker op hoeveel decimalen er moet afgerond worden.

Voorbeeld:

```
Bereken zijde van een kubus uit het volume
Volume van de kubus: 144
Op hoeveel tekens na de komma (decimale punt) afronden? 2
De zijde is 5.24

>>>
```

```
""" Oefening 1.7
    Zijde van kubus uit volume - afronden op gewenst aantal cijfers
    Uitbreiding oefening 1.6
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

# Titel op scherm en input

print("Bereken de zijde van de kubus uit het volume:")
volume = float(input("Volume van de kubus: "))
afronding =int(input("Op hoeveel tekens na de komma (decimale punt) afronden? "))

# Berekening

zijde = volume ** (1/3) # Tot de macht 1/3 is de derdemachtswortel

""" Uitbreiding """

zijde = zijde * 10**afronding + .5 # plus 0.5 om wiskundig te kunnen afronden
zijde = int(zijde)
zijde = zijde / 10**afronding

""" of in één formule

zijde = int(zijde * 10**afronding + .5)/10**afronding

"""

print("De zijde is",zijde)
```

Oefening 1.8: Tijdsduur tussen 2 tijdstippen

Herhaal oefening 1.3 maar geef je resultaat in uren, minuten en seconden.

Voorbeeld:

```
Tijdsduur tussen 2 tijdstippen
------------------------------
Tijdstip 1:
   uur: 12
   minuten: 0
   seconden: 0
Tijdstip 2:
   uur: 13
   minuten: 10
   seconden: 12
Tijdsduur tussen de twee tijdstippen: 1:10:12

>>>
```

Module 2: Conditie

Opdrachten

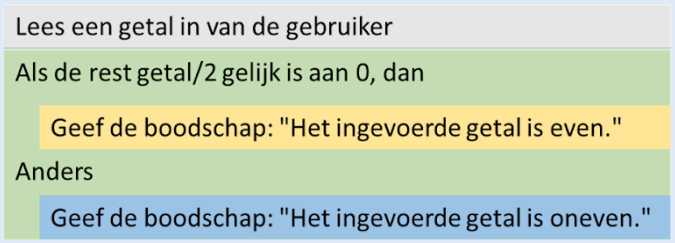
Opdracht 2.1

Een even getal is een geheel getal dat restloos deelbaar is door 2, dat wil zeggen bij deling door 2 is het resultaat weer een geheel getal.

Maak een algoritme dat aangeeft of een getal even of oneven is.

Maak dit in pseudo code (gewone Nederlandse taal).

Oplossing:



Opdracht 2.2

In de wiskunde is een vierkantsvergelijking of kwadratische vergelijking een vergelijking van de vorm

$ax^2 + bx + c = 0$.

Hierbij zijn a, b en c reële getallen (kan ook complex zijn, maar dit laten we hier buiten beschouwing).

Het kan zijn dat in eerste instantie deze vergelijking niet deze vorm heeft, maar als we alle termen naar hetzelfde lid brengen, dan herleidt deze wel naar deze vorm.

Maak een algoritme dat de oplossingen geeft van een vierkantsvergelijking in zijn standaardvorm. Maak dit in pseudo code (gewoon Nederlandse taal).

Oplossing:

Lees coëfficiënten: a,b en c
Bereken discriminant $D = b^2 - 4ac$

Als $D < 0$ dan
  Output: geen oplossingen

Als $D = 0$ dan
  Bereken $x = \frac{-b+\sqrt{D}}{2a}$
  Output: twee gelijke oplossingen x

Als $D > 0$ dan
  Bereken $x_1 = \frac{-b+\sqrt{D}}{2a}$
  Bereken $x_2 = \frac{-b-\sqrt{D}}{2a}$
  Ouput: twee verschillende oplossingen x_1 en x_2

Opdracht 2.3

Het doel van deze oefening is om een algoritme en een script te schrijven dat vraagt om een jaartal, waarvan het script dan bepaald of het een schrikkeljaar is of niet.

Wanneer is een jaar een schrikkeljaar?

Elk jaar dat evenredig deelbaar is door 4 is een schrikkeljaar: bijvoorbeeld 1988, 1992 en 1996 zijn schrikkeljaren.

Er is echter nog steeds een kleine fout die moet worden verantwoord. Om deze fout te elimineren, bepaalt de Gregoriaanse kalender dat een jaar dat gelijkmatig deelbaar is door 100 (bijvoorbeeld 1900) alleen een schrikkeljaar is als deze ook gelijkmatig deelbaar is door 400.

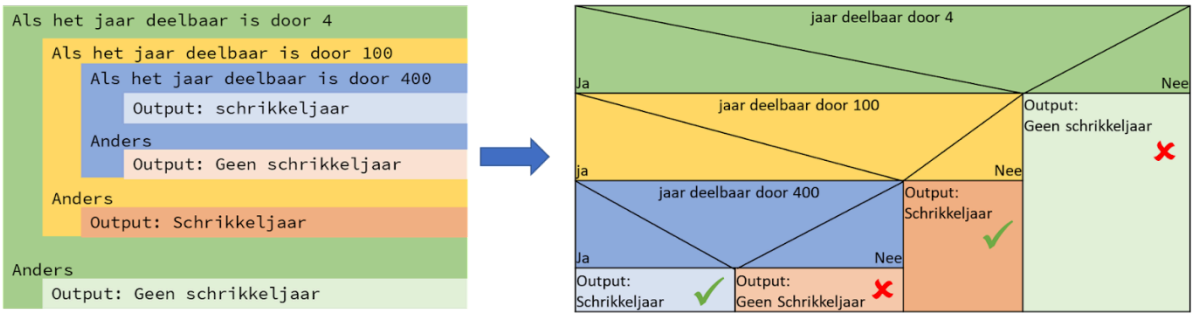
Je merkt dat je vele condities moet controleren voordat je kan bepalen of een jaartal een schrikkeljaar is of niet:

- Deelbaar door 4
- Deelbaar door 100
- Deelbaar door 400

Onderstaande tabel met enkele voorbeelden kan je helpen om de condities te begrijpen:

	Deelbaar door			
Jaartal	**4**	**100**	**400**	**Schrikkeljaar**
1986	☐	☐	☐	☐
1992	✔	☐	☐	✔
1900	✔	✔	☐	☐
2000	✔	✔	✔	✔

Je kan hieronder in pseudo code schrijven wanneer een jaar een schrikkeljaar is



We worden in onze Westerse cultuur aangeleerd om op een bepaalde manier te denken. Bijvoorbeeld, wij denken bijvoorbeeld van links naar rechts. Als gevraagd wordt om de tweede deur te openen, zal bijna iedereen de tweede deur van links nemen.

Ook bekijken we problemen van het algemene naar het specifieke. Zo zal je waarschijnlijk eerst gaan controleren zijn of het jaartal deelbaar is door 4, en daarna specifieker gaan kijken.

Wanneer je de tabel bekijkt, dan kan je je probleem ook op een andere manier naderen.

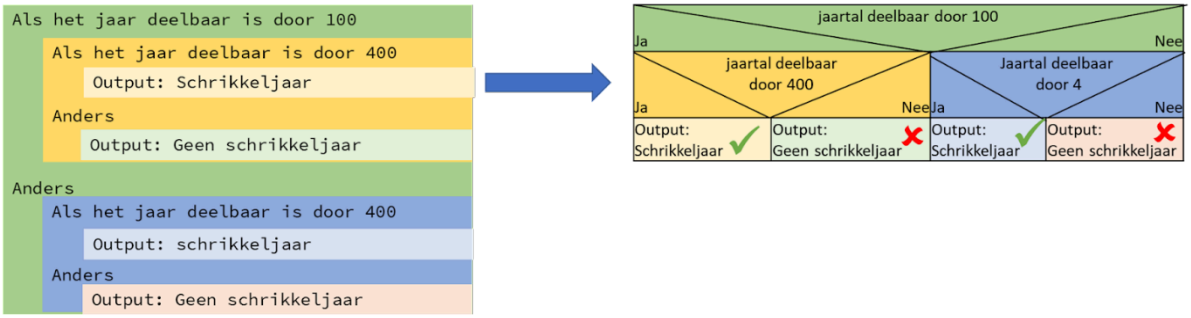
Als we eerst gaan kijken of het getal deelbaar is door 100, dan verdelen we alle mogelijkheden in twee grote groepen:

	Deelbaar door			
Jaartal	**4**	**100**	**400**	**Schrikkeljaar**
1986	☐	☐	☐	☐
1992	✔	☐	☐	✔

	Deelbaar door			
Jaartal	**4**	**100**	**400**	**Schrikkeljaar**
1900	✔	✔	☐	☐
2000	✔	✔	✔	✔

In de bovenste tabel moet ik dan nog enkel gaan kijken of het jaartal deelbaar is door 4, in de onderste tabel of het jaartal deelbaar is door 400.

Tracht zelf een algoritme te schrijven voor dit probleem. Je kan zelf kiezen of je het grafisch of in pseudo code doet.



Oplossing:

```
""" Opdracht 2.3
    Schrikkeljaar of niet?
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

# info gebruiker
print("Schrikkeljaar of niet?")
print("----------------------")
print("Geef een jaartal in en zie of het een schrikkeljaar is of niet.")

#input
jaar = int(input("  jaartal: "))

""" je kan ook gebruik maken van % om te zien of een jaar deelbaar is"""

if jaar//4==jaar/4:
    if jaar//100 ==jaar/100:
        if jaar//400 == jaar/400:
            print("Dit is een schrikkeljaar")
        else:
            print("Dit is GEEN schrikkeljaar")
    else:
        print("Dit is een schrikkeljaar")
else:
    print("Dit is GEEN schrikkeljaar")
```

Alternatieve oplossing:

```
""" Opdracht 2.3 (alternatief)
    Schrikkeljaar of niet?
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

# info gebruiker
print("Schrikkeljaar of niet?")
print("----------------------")
print("Geef een jaartal in en zie of het een schrikkeljaar is of niet.")

#input
jaar = int(input("  jaartal: "))

""" je kan ook gebruik maken van % om te zien of een jaar deelbaar is"""

if jaar//100==jaar/100:
```

```
    if jaar//400 ==jaar/400:
        print("Dit is een schrikkeljaar")
    else:
        print("Dit is GEEN schrikkeljaar")
else:
    if jaar//4 ==jaar/4:
        print("Dit is een schrikkeljaar")
    else:
        print("Dit is GEEN schrikkeljaar")
```

Oefening 2.1

Maak en algoritme en zet dit algoritme om in een script dat:

- Aan de gebruiker twee gehele getallen vraagt.
- Her script drukt het kleinste van de twee getallen op het scherm.

```
""" Oefening 2.1
    Kleinste van 2 getallen
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

# info en input
print("Welk getal is het kleinste?")
getal1=float(input("  Getal 1: "))
getal2=float(input("  Getal 2: "))

# output
if getal1<getal2:
    print("Het kleinste getal is",getal1)
else:
    print("Het kleinste getal is",getal2)
```

Alternatief:

```
""" Oefening 2.1 (Alternatief)
    Kleinste van 2 getallen
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

# info en input
print("Welk getal is het kleinste?")
getal1=float(input("  Getal 1: "))
getal2=float(input("  Getal 2: "))

# Als getal1 niet het kleinste is stel getal1 gelijk aan getal2
if getal2<getal1:
    getal1 = getal2

# Output
print("Het kleinste getal is",getal1)
```

Oefening 2.2

Maak een algoritme en zet dit algoritme om in een script dat:

- Aan de gebruiker een getal vraagt
- Het script zegt of het getal negatief, positief of gelijk aan nul is

```
""" Oefening 2.2
    Positief, negatief of gelijk aan 0
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                        """

# info en input
print("Is het ingegeven getal positief, negatief of gelijk aan nul?")
getal=float(input("  Getal: "))

# condities en output
if getal1<0:
    # Getal is kleiner dan nul
    print("Het getal is negatief.")
else:
    # Getal is groter of gelijk aan 0
    if getal >0:
        print("Het getal is positief.")
    else:
        print("Het getal is gelijk aan nul.")
```

Oefening 2.3

Gegeven 3 gehele getallen, waarvan er 2 aan elkaar gelijk zijn.

Schrijf een script dat laat weten welk van de 3 getallen het afwijkende getal (het zwarte schaap) is. Er zijn dus 3 mogelijke antwoorden:

- "Het eerste getal, nl. … is het zwarte schaap."
- "Het tweede getal, nl. … is het zwarte schaap."
- "Het derde getal, nl. … is het zwarte schaap."

```
""" Oefening 2.3
    Het zwarte schaap
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                        """

# info en input
print("Welk getal is het zwarte schaap?")
getal1=float(input("  Getal1: "))
getal2=float(input("  Getal2: "))
getal3=float(input("  Getal3: "))

# condities en output
if getal1==getal2:
    # getal3 is het zwarte schaap
    print("Het derde getal, nl.",getal3,"is het zwarte schaap.")
else:
    # getal1 of getal2 is het zwarte schaap
    if getal1==getal3:
        # getal1 en getal3 gelijk --> getal2 zwarte schaap
```

```
        print("Het tweede getal, nl.",getal2,"is het zwarte schaap.")
    else:
        print("Het eerste getal, nl.",getal1,"is het zwarte schaap.")
```

Oefening 2.4

Maak een algoritme en zet dit algoritme om in een script dat:

- Aan de gebruiker vraagt een cijfer in te voeren.
- Het script moet bepalen of het cijfer onvoldoende (minder dan 5), voldoende (5-7), goed (8-9) of uitstekend (10) is.

Door je mogelijk steeds in 2 te kappen kom je zeker bij grote zoekfuncties sneller bij je resultaat

```
""" Oefening 2.4
    Geslaagd? Beoordeling resultaat
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

# info en input
print("Welk cijfer behaalde je op 10?")
cijfer=float(input("  cijfer: "))
# condities en output
if cijfer>7:
    # Je behaalde GOED of UITSTEKEND
    if cijfer == 10:
        print("Je resultaat is UITSTEKEND")
    else:
        print("Je resultaat is GOED")
else:
    # Je behaalde onvoldoende of voldoende
    if cijfer < 5:
        print("Je resultaat is ONVOLDOENDE")
    else:
        print("Je resultaat is VOLDOENDE")
```

Oefening 2.5

Maak een algoritme en zet dit algoritme om in een script dat:

- Aan de gebruiker vraagt zijn leeftijd in te voeren.
- Het script bepaalt of de gebruiker alcohol en/of sterke drank mag kopen.
    - Alcohol: ouder dan 16
    - Sterke drank: ouder dan 18

```
""" Oefening 2.5
    Mag de gebruiker alcohol of sterke drank drinken
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

# info en input
```

```
print("Bepalen of je alcohol of sterke drank mag drinken")
leeftijd=float(input("  Hoe oud ben je? "))
# condities en output
if leeftijd<16:
    print("Je mag geen alcohol of sterke drank drinken")
    if leeftijd > 18:
        print("Je mag WEL alcohol en sterke drank drinken")
    else:
        print("Je mag WEL alcohol maar GEEN sterke drank drinken")
```

Opdracht 2.4

Je gebruikt een paraplu om onder te zitten als de zon schijnt voor de schaduw, of als het regent om niet nat te worden.

Onderzoek de condities wanneer je onder de paraplu gaat zitten.

Condities:

- Conditie 1: Het regent (?)
- Conditie 2: De zon schijnt (?)

Vul onderstaande tabel aan.

- Schrijf in de tabel onder conditie 1:
  - 1: als aan conditie 1 voldaan is. (Als het regent)
  - 0: als aan de conditie niet voldaan is. (Als het niet regent)
- Doe hetzelfde voor conditie 2
  - 1: als aan conditie 1 voldaan is. (Als het regent)
  - 0: als aan de conditie niet voldaan is. (Als het niet regent)
- Resultaat:
  - 1: als je onder de paraplu gaat zitten
  - 0: als je niet onder de paraplu gaat zitten

Weer	Conditie 1: Het regent?	Conditie 2: De zon schijnt?	Resultaat: Onder de paraplu?
Bewolkt zonder regen	0 (false)	0 (false)	0 (false)
Blauwe lucht zonder regen en stralende zon	0 (false)	1 (true)	1 (true)
Het regent pijpenstelen met een donker wolkendek	1 (true)	0 (false)	1 (true)
De regen en de zon laten een mooie regenboog zien.	1 (true)	1 (true)	1 (true)

Bij welke combinaties van conditie 1 en 2 zal het resultaat waar zijn, en je dus onder de paraplu gebruikt.

Je gaat onder de paraplu als het regent OF als de zon schijnt.
Je gaat dus onder de paraplu als aan minstens 1 van de voorwaarden voldaan is.

Opdracht 2.5

Controleer of de steden in de tabel Europese hoofdsteden zijn.

Schrijf 2 condities op waaraan een stad moet voldoen om een Europese hoofdstad te zijn. Vul daarna de tabel aan.

- Conditie 1:

1: als de stad een hoofdstad (van een land) is
0: als de stad GEEN hoofdstad is

- Conditie 2:

1: als de stad in Europa ligt
0: als de stad NIET in Europa ligt

Stad	Conditie 1:	Conditie 2:	Resultaat: Europese hoofdstad?
Rio de Janeiro	0 (false)	0 (false)	0 (false)
Hasselt	0 (false)	1 (true)	0 (false)
Ottawa	1 (true)	0 (false)	0 (false)
Brussel	1 (true)	1 (true)	1 (true)

Bij welke combinaties van conditie 1 en 2 zal het resultaat waar zijn, en je dus een Europese hoofdstad hebt?

Het is een Europese hoofdstad als het een hoofdstad is EN als ze in Europa ligt.
Het is een Europese hoofdstad als aan alle (beide) voorwaarden voldaan is.

Opdracht 2.6

In het studentenrestaurant van UHasselt krijgen volgende categorieën korting:

- Studenten van UHasselt
- Werknemers van UHasselt

1. Vervolledig onderstaande tabel

Persoon	Conditie 1: **Student UHasselt?**	Conditie 2: **Werknemer UHasselt?**	Resultaat: Korting?
Een student geneeskunde	TRUE	FALSE	TRUE
Een praktijkassistent Fysica	FALSE	TRUE	TRUE

Een gepensioneerde docent	FALSE	FALSE	FALSE
Een praktijkassistente die ook een extra masteropleiding volgt	TRUE	TRUE	TRUE

2. Ontwerp daarna een algoritme dat bepaald of iemand korting krijgt of niet. Je mag wel maar één selectie hebben in je algoritme.

Lees: "Studeer je aan UHasselt?" student

Lees: "Werk je aan UHasselt?" werknemer

Als student = 'JA' of werknemer = 'Ja' dan

    Output: KORTING

Anders

    Output: GEEN KORTEN

3. Zet je algoritme om in een Python script.

Oplossing

```
""" Opdracht 2.6
    Korting in studentenrestaurant
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

# info en input
print("Bepalen of je korting krijgt in het restaurant")
student=float(input("  Studeer je aan UHasselt? (ja/nee) "))
werknemer=float(input("  Werk je bij UHasselt? (ja/nee) "))

# condities en output
if student=="ja" or werkenemer == "ja":
    print("Je krijgt korting.")
else:
    print("Je krijgt GEEN korting.")
```

Oefening 2.6

Schrijf een script dat

- aan de gebruiker een geheel getal vraagt
- het script geeft een antwoord op de vraag "Bestaat het ingevoerde getal uit exact 3 cijfers?"

```
""" Oefening 2.6
    Getal met exact 3 cijfers
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

# info en input
print("Heeft het getal exact 3 cijfers?")
```

```
getal=float(input("  Geef een getal: "))

# condities en output
if getal>99 and getal<1001:
    # Je kan ook zeggen getal>=100 and getal<=999
    print("Het getal heeft exact 3 cijfers.")
else:
    print("Het getal heeft geen 3 cijfers.")
```

Oefening 2.7

Schrijf een programma dat de gebruiker vraagt naar hun leeftijd.

Het programma moet controleren of de persoon oud genoeg is om in aanmerking te komen voor een bepaalde activiteit. De persoon moet tussen 18 en 30 jaar oud zijn.

```
""" Oefening 2.7
    Mag je meedoen?
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

# info en input
print("Je mag enkel meedoen als je tussen 18 en 30 bent.")
leeftijd=float(input("  Wat is je leeftijd? "))

# condities en output
if leeftijd>=18 and leeftijd=<30:
    # 18 en 30 mogen ook meedoen
    print("Je mag meedoen.")
else:
    print("Sorry, je mag niet meedoen.")
```

Oefening 2.8

Schrijf een programma dat de lengtes van drie zijden van een driehoek vraagt en controleert of het een gelijkbenige driehoek is.

```
""" Oefening 2.8
    Gelijkbenige driehoek
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

# info en input
print("Geef telkens de lengte van de zijdes van een driehoek")
zijde1=float(input("  Lengte van zijde 1: "))
zijde2=float(input("  Lengte van zijde 2: "))
zijde3=float(input("  Lengte van zijde 3: "))

# condities en output
if zijde1==zijd2 or zijde1==zijde3 or zijde2==zijde3:
    print("Het is een gelijkzijdige driehoek.")
else:
    print("Het is GEEN gelijkzijdige driehoek.")
```

Oefening 2.9

Schrijf een programma dat de gebruiker vraagt om een cijfer en controleert of het in het bereik van 0-100 ligt en of het een voldoende (>= 60) is.

```
""" Oefening 2.9
    Ingave voldoet en cijfer is voldoende?
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                        """

# info en input
print("Geef het resultaat in op 100")
resultaat=float(input("  Resultaat: "))

# condities en output
if resultaat>=60 and resultaat<=100:
    print("Het resultaat is voldoende.")
else:
    print("Het resultaat is onvoldoende, of de ingaven voldoet niet.")
```

Oefening 2.10

Schrijf een programma dat de gebruiker vraagt naar het totaalbedrag van hun aankoop en of ze een VIP-klant zijn. Als ze een VIP-klant zijn of als het totaalbedrag meer dan 100 euro is, geef dan een bericht over de korting.

```
""" Oefening 2.10
    Korting in winkel of niet
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

# info en input
print("Bepalen of je korting krijgt:")
bedrag=float(input("  Aankoopbedrag: "))
vip=input("  Ben je VIP? (ja/nee) ")

# condities en output
if VIP=='ja' or bedrag>100:
    print("Je hebt recht op korting.")
else:
    print("Je hebt GEEN recht op korting.")
```

Oefening 2.11

Gegeven een veld op het schaakbord.

Schrijf een script dat antwoord geeft op de vraag "Is het een 'donker' of 'licht' veld?". Er zijn twee mogelijke antwoorden:

- "Donker veld", als het opgegeven veld donker is
- "Licht veld", als het opgegeven veld licht is.

```
""" Oefening 2.11
    Schaakbord: donker of licht veld
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

# info en input
print("Dit programma geeft aan of een vakje licht of donker is.")
k=int(input("  kolomnummer: "))
r=int(input("  rijnummer: "))

#als juist 1 cijfer (rij of kolom) oneven is licht veld

if r//2!=k//2:
    print("Licht veld.")
else:
    print("Donker veld")

"""Alternatief
- twee even cijfers optellen --> even getal (donker)
- twee oneven cijfers optellen --> even getal (donker)
- een even en een oneven getal optellen --> oneven getal (licht)

if (r+k)%2==0:
   print('licht')
else:
```

```
   pring('donker')

 getal%2 --> bepaald rest van de deling door 3
 """
```

Oefening 2.12

Gegeven twee velden op een schaakbord.

Schrijf een script dat antwoord geeft op de vraag "Hebben beide velden dezelfde of een verschillende kleur?"

Er zijn twee mogelijke antwoorden:

- "Zelfde kleur", als beide velden dezelfde kleur hebben
- "Verschillende kleur", als beide velden een verschillende kleur hebben

```
""" Oefening 2.12
    Schaakbord: zelfde kleur
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

# info en input
print("Dit programma geeft aan of een vakje licht of donker is.")
print("Vakje 1:")
k1=int(input("  kolomnummer: "))
r1=int(input("  rijnummer: "))
print("Vakje 2:")
k2=int(input("  kolomnummer: "))
r2=int(input("  rijnummer: "))

#als som alle cijfers even is ==> zelfde kleur

if (r1+r2+k1+k2)//2==(r1+r2+k1+k2)/2:
    print("Zelfde kleur.")
else:
    print("Verschillende kleur.")
```

Oefening 2.13

Schrijf een script dat aan de gebruiker vraagt om 2 getallen in te geven

Het script geeft dan antwoord op de vraag "Is er minstens één negatief getal ingegeven?"

Er zijn twee mogelijke antwoorden:

- "Minstens één getal is negatief", als één of beide getallen negatief is.
- "Geen negatieve getallen", als beide getallen positief zijn.

```
""" Oefening 2.13
    Minstens één getal negatief
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

# info en input
print("Kijken of minstens één getal negatief is: ")
```

```
getal1=int(input("  Eerst getal:  "))
getal2=int(input("  Tweede getal: "))

# Conditie en output
if getal1<0 or getal2<0:
    print("Minstens één getal is negatief")
else:
    print("Geen negatieve getallen.")
```

Oefening 2.14

Schrijf een script dat aan de gebruiker vraagt om 2 getallen in te geven.

Het script geeft dan antwoord op de vraag: ”Hebben beide getallen hetzelfde teken?”

Er zijn twee mogelijke antwoorden:

- “De getallen hebben hetzelfde teken”, als de getallen beide negatief of beide positief zijn.
- “De getallen hebben verschillende tekens”, als één van de getallen positief, en het andere negatief is.

```
""" Oefening 2.14
    Zelfde teken
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                        """

# info en input
print("Kijken of beide getallen zelfde teken hebben: ")
getal1=int(input("  Eerst getal:  "))
getal2=int(input("  Tweede getal: "))

# Conditie en output
if getal1*getal2<0 :
    print("Beide getallen hebben en verschillend teken.")
else:
    print("Beide getallen hebben en hetzelfde teken..")
```

Oefening 2.15

Vervolledig onderstaande code zodat de getallen die de gebruiker ingeeft gesorteerd op het scherm komen.

De gegeven code mag niet aangepast worden, je mag enkel code toevoegen.

```
# oefening 2.15

print("Geef 2 gehele getallen in")
getal1=int(input("getal 1: "))
getal2=int(input("getal 2: "))

# code aanvullen

# Output
print(getal1,"is kleiner dan",getal2)
```

Voorbeeld:


```
# oefening 2.15

print("Geef 2 gehele getallen in")
getal1=int(input("  getal 1: "))
getal2=int(input("  getal 2: "))

# code aanvullen
if getal1>getal2:
    hulp = getal1
    getal1 = getal2
    getal2 = hulp

# Output
print(getal1,"is kleiner dan",getal2)
```

Alternatieve oplossing:

```
# oefening 2.15 (alternatief)

print("Geef 2 gehele getallen in")
getal1=int(input("  getal 1: "))
getal2=int(input("  getal 2: "))

# code aanvullen
if getal1>getal2:
    getal1 , getal2 = getal2 , getal1

# Output
print(getal1,"is kleiner dan",getal2)
```

Oefening 2.16

We noemen een integer een cijferpalindroom wanneer het zowel van voor naar achter als van achter naar voor kan gelezen worden (voorbeeld: 1991).

Schrijf een script dat aan de gebruiker vraagt om een natuurlijk getal van 4 cijfers in te geven. Het script laat dan weten of het ingevoerde getal al dan niet een cijferpalindroom is.

De mogelijke antwoorden zijn:

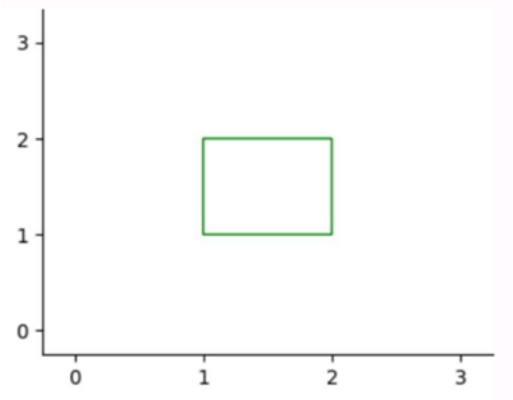
- "EEN CIJFERPALINDROOM"
- "GEEN CIJFERPALINDROOM".

```
""" Oefening 2.16
    Palindroom
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                        """

print("Palindroom?????")
getal = int(input("  Geef een natuurlijk getal van 4 cijfer: "))
```

```
duizend = getal//1000
rest = getal -1000*duizend # rest = getal%1000
honderd = rest//100
rest = rest - 100*honderd # rest = rest%100
tien = rest//10
eenheden = rest - tien*10 # eenheden = rest%10

if duizend==eenheden and honderd==tien:
    print("een cijferpalindroom")
else:
    print("GEEN cijferpalindroom")
```

Oefening 2.17

Gegeven 3 hoekpunten van een rechthoek. De zijden van de rechthoek zijn evenwijdig met de x- en de y-as.



Van de drie gekende hoeken worden telkens de x- en de y-coördinaten ingegeven onder de vorm van 6 gehele getallen.

Het script berekent dan de coördinaten van het vierde hoekpunt.

```
""" Oefening 2.17
    Vierde hoekpunt
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

print("Vierde hoekpunt")
print("---------------")
print()
print("Hoekpunt 1:")
x1 = input("  x: ")
y1 = input("  y: ")
print("Hoekpunt 2:")
x2 = input("  x: ")
y2 = input("  y: ")
print("Hoekpunt 3:")
x3 = input("  x: ")
y3 = input("  y: ")

# x-coördinaat bepalen
if x1==x2:
    x4=x3
else:
```

```
    if x1==x3:
        x4=x2
    else:
        x4=x1
# y-coördinaat bepalen
if y1==y2:
    y4=y3
else:
    if y1==y3:
        y4=y2
    else:
        y4=y1
tekst ="De coördinaten van het vierde hoekpunt zijn: ("+x4+","+y4+")."
print(tekst)
```

Oefening 2.18

Pas het script van oefening 2.17 aan zodat de zijden niet evenwijdig moeten liggen met de assen.

Opmerking

- Chat GPT gaf op 3/11/2023 geen juiste code.
- Bepaal eerst met welk punt je samen met de twee andere een rechte hoek kunt vormen.

```
""" Oefening 2.18
    Vierde hoekpunt - niet evenwijdig met zijden
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

print("Vierde hoekpunt")
print("---------------")
print()
print("Hoekpunt 1:")
x1 = input("  x: ")
y1 = input("  y: ")
print("Hoekpunt 2:")
x2 = input("  x: ")
y2 = input("  y: ")
print("Hoekpunt 3:")
x3 = input("  x: ")
y3 = input("  y: ")

# rico's
m12 = (y2-y1)/(x2-x1)
m23 = (y3-y2)/(x3-x2)
m13 = (y3-y1)/(x3-x1)

if m12**m23==-1:
    #punt 2 is hoekpunt
    x4 = x3+x1-x2
    y4 = y3+y1-y2
else:
    if m12**m13==-1:
        #punt 1 is hoekpunt
        x4 = x3+x2-x1
```

```
        y4 = y3+y2-y1
    else:
        #punt 3 is hoekpunt
        x4 = x1+x2-x3
        y4 = y1+y2-y3

tekst ="De coördinaten van het vierde hoekpunt zijn: ("+x4+","+y4+")."
print(tekst)



# x-coördinaat bepalen
if x1==x2:
    x4=x3
else:
    if x1==x3:
        x4=x2
    else:
        x4=x1
# y-coördinaat bepalen
if y1==y2:
    y4=y3
else:
    if y1==y3:
        y4=y2
    else:
        y4=y1
tekst ="De coördinaten van het vierde hoekpunt zijn: ("+x4+","+y4+")."
print(tekst)
```

Module 3: Iteratie

Opdracht 3.1

Een priemgetal is een natuurlijk getal dat groter is dan 1 en alleen door zichzelf en door 1 gedeeld kan worden.

Onderzoek, met pen en papier, of de getallen 15 en 71 priemgetallen zijn.

Noteer goed welke stappen je onderneemt om dit te achterhalen.

Wanneer ga je stoppen met controleren of het een priemgetal is?

Je kan stoppen als je een deler gevonden hebt (geen priemgetal) of als je over $\sqrt{15}$ of $\sqrt{71}$ zit. Eenmaal daar over is het een priemgetal.

Opdracht 3.2

Schrijf een algoritme en een script dat aan de gebruiker vraagt een getal in te geven.

Geef alle delers van het getal.

Lees een natuurlijk getal getal

Teler 2

Output: 1

Zolang teller<getal doe

    Als getal/teller geen rest dan

        Output: teller

    Verhoog teller met 1

Opdracht 3.3

Pas opdracht 3.2 aan, zodat enkel een getal van 10 t/m 20 aanvaard wordt als ingave.

getal 0

zolang getal<10 OF getal>20 herhaal

    Lees een natuurlijk getal getal

Teler 2

Output: 1

Zolang teller<getal herhaal

    Als getal/teller geen rest dan

        Output: teller

    Verhoog teller met 1

Oefening 3.1

Vraag de gebruiker om een natuurlijk getal.
Druk alle getallen af van 0 tot en met dat getal

Voorbeeld:

```
>>> %Run -c $EDITOR_CONTENT
  Geef een natuurlijk getal: 5
  0
  1
  2
  3
  4
  5
>>>
```

```
""" Oefening 3.1
    Alle natuurlijke getalen tot deingave
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

getal = int(input("Geef een natuurlijk getal: "))
teller = 0
while teller<=getal:
    print(teller)
    teller = teller + 1 #Dit kan verkort worden tot teller+=1
```

Oefening 3.2

Vraag aan een gebruiker om twee natuurlijke getallen in te geven.
Druk alle even getallen af vanaf het kleinste van de twee getallen tot en met het grootste.

Voorbeeld;

```
>>> %Run 'even getallen in range.py'
  Geef eerste getal in: 12
  Geef tweede getal in: 16
  12
  14
  16
>>>
```

```
>>> %Run 'even getallen in range.py'
  Geef eerste getal in: 17
  Geef tweede getal in: 12
  12
  14
  16
>>>
```

```
>>> %Run 'even getallen in range.py'
  Geef eerste getal in: 11
  Geef tweede getal in: 16
  12
  14
  16
>>>
```

```
""" Oefening 3.2
    Alle even getallen tussen 2 waardes
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

getal1 = int(input("Geef eerste getal in: "))
getal2 = int(input("Geef tweede getal in: "))
```

```
if getal2<getal1:
    x =  getal1
    getal1 = getal2
    getal2 = x

if getal1//2 !=getal1/2:
    getal1=getal1+1

while getal1 <= getal2:
    print(getal1)
    getal1=getal1+2
```

Oefening 3.3

Vraag aan de gebruiker welke tafel van vermenigvuldiging je moet printen en print deze.

Voorbeeld:

```
>>> %Run -c $EDITOR_CONTENT
  Welke tafel? 5
  1 x 5 = 5
  2 x 5 = 10
  3 x 5 = 15
  4 x 5 = 20
  5 x 5 = 25
  6 x 5 = 30
  7 x 5 = 35
  8 x 5 = 40
  9 x 5 = 45
  10 x 5 = 50
>>>
```

```
""" Oefening 3.3
    Tafel van vermenigvuldiging printen
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

tafel = int(input("Welke tafel? "))
x = 1

while x <= 10:
    print(x,"x",tafel,"=",x*tafel)
    x = x+1
```

Oefening 3.4

Met onderstaande code kan je de computer een willekeurig natuurlijk getal laten generen tussen de opgegeven getallen. In ons geval 1 en 6.

```
# Script om een getal van 1 t/m 6 te generen
# Importeer de random module
import random
x = random.randint(1,6)
# Vul hieronder de code zelf aan
```

Laat de gebruiker een getal ingeven van 1 tot 10.

Laat de computer dan zoveel keren een getal genereren tussen 1 en 6 en print dit getal op het scherm

```
""" Oefening 3.4
    X keer getal genereren (sim dobbelsteen)
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

import random
aantal = int(input("Geef een natuurlijk getal in tussen 1 en 10: "))

""" uitbreiding: controleren of het getal tussen 1 en 10 ligt

while aantal<0 or aantal>10:
    aantal = int(input("Geef een natuurlijk getal in tussen 1 en 10: "))

"""

x = 1
while x <= aantal:
    print(random.randint(1,6))
    x = x+1
```

Oefening 3.5:

Schrijf een script dat aan de gebruiker een natuurlijk getal vraagt tussen 1 en 20.

Als de gebruiker een getal ingeeft dat niet voldoet aan de voorwaarde moet het script een nieuwe ingave vragen, anders geeft het als output: "Voldoet".

```
Geef een natuurlijk getal in van 1 tot/met 20: 5.2
Geef een natuurlijk getal in van 1 tot/met 20: -3
Geef een natuurlijk getal in van 1 tot/met 20: -3
Geef een natuurlijk getal in van 1 tot/met 20: -3.5
Geef een natuurlijk getal in van 1 tot/met 20: 4
Voldoet
>>>
```

```
""" Oefening 3.5
    Invoer controleren
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

aantal = int(input("Geef een natuurlijk getal van 1 tot/met 20: "))

while aantal<1 or aantal>10:
    aantal = int(input("Geef een natuurlijk getal in tussen 1 en 10: "))

print("Voldoet")
```

Je kan op voorhand ook een waarde aan aantal geven waaraan de conditie reeds voldoet. Dan moet je slechts éénmaal een input-functie gebruiken

```
""" Oefening 3.5
    Invoer controleren
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """
```

```
aantal = 0

while aantal<1 or aantal>10:
    aantal = int(input("Geef een natuurlijk getal in tussen 1 en 10: "))

print("Voldoet")
```

Oefening 3.6:

Laat de computer een getal van 1 tot 20 genereren;

Laat de gebruiker dan het getal raden.

Bij elke poging geeft de computer aan of het getal te hoog of te laag is.

Als de gebruiker het getal geraden heeft, geeft de computer het aantal pogingen dat nodig was op het scherm weer.

```
>>> %Run -c $EDITOR_CONTENT
  Welk getal denk je dat we zoeken? 10
  Het getal dat we zoeken is groter dan 10

  Welk getal denk je dat we zoeken? 15
  Het getal dat we zoeken is kleiner dan 15

  Welk getal denk je dat we zoeken? 12
  Het getal dat we zoeken is kleiner dan 12

  Welk getal denk je dat we zoeken? 11
  Het getal dat we zoeken is groter dan 11

  Proficiat! Het getal dat we zochten was 11
  Je had 4 pogingen nodig.
>>>
```

```
""" Oefening 3.6
    Getal raden
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

import random

zoek = random.randint(1,20)

aantal = 0
gok = 0
while gok!=zoek:
    gok = int(input("Welke getal denk je dat we zoeken? "))
    aantal = aantal + 1
    if gok>zoek:
        print("Het getal dat we zoeken is kleiner dan",gok)
    else:
        if gok<zoek:
            print("Het getal dat we zoeken is groter dan",gok)
    print()

print("Proficiat! Het getal dat we zochten was",zoek)
print("Je had",aantal,"pogingen nodig.")
```

Oefening 3.7:

Draai oefening 3.6 nu om. Laat de gebruiker een getal ingeven van 1 tot/met 20 en laat de computer raden naar het getal.

De gebruiker geeft steeds aan of het getal dat gezocht wordt groter (G), kleiner (K) of gelijk(=) aan het getal dat gezocht wordt.

Als de computer het getal geraden heeft, geeft de computer het aantal pogingen dat nodig was op het scherm weer.

Je kan de computer natuurlijk gewoon laten gokken, maar er bestaat een algoritme waarbij de computer in maximaal 5 stappen het juiste getal raadt.

```
Ik kies 10
Is het getal dat ik zoek groter (G), kleiner (K) of gelijk aan (=) dit getal? g
Het getal dat we zoeken is groter dan 10

Ik kies 15
Is het getal dat ik zoek groter (G), kleiner (K) of gelijk aan (=) dit getal? k
Het getal dat we zoeken is kleiner dan 15

Ik kies 12
Is het getal dat ik zoek groter (G), kleiner (K) of gelijk aan (=) dit getal? =

Joepie! Het getal dat ik zocht was 12
Ik had 3 pogingen nodig.
```

```
""" Oefening 3.7
    Getal raden door computer
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24
"""

aantal = 0
gok = 0
min = 1
max = 20
antwoord ="G"
while antwoord!="=":
    gok = (min+max)//2
    print("Ik kies",gok)
    antwoord=input("Is het getal dat ik zoek groter (G), kleiner (K) of gelijk aan (=) dit getal? ")
    if antwoord == "G" or antwoord =="g":
        print("Het getal dat we zoeken is groter dan",gok)
        min = gok + 1
    else:
        if antwoord == "K" or antwoord =="k":
            print("Het getal dat we zoeken is kleiner dan",gok)
            max = gok -1
    aantal = aantal +1
    print()

print("Joepie! Het getal dat ik zocht was",gok)
print("Ik had",aantal,"pogingen nodig.")
```

Oefening 3.8:

Schrijf een script dat de eerste 20 getallen van Fibonacci berekent en toont op het scherm.

```
""" Oefening 3.8
    Fibonacci: 20 eerste getallen
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

fib1 = 0
fib2 = 1
teller = 3
print("Getallen van Fibonacci")
print("----------------------")
print(fib1)
print(fib2)
while teller <=20:
    fib3 = fib1 + fib2
    print(fib3)
    fib1 = fib2
    fib2 = fib3
    teller=teller+1
```

Opdracht 3.4

```
# Priemgetallen - tot nul ingegeven
getal = 1
while getal > 0:
    getal=int(input("Geef een natuurlijk getal in:"))
    deler = 2
    priemgetal = True

    while priemgetal==True:
        quotient = getal//deler
        rest = getal - quotient*deler
        if rest == 0:
            priemgetal = False
        else:
            deler=deler+1
    if deler == getal:
        print("Priemgetal")
    else:
        print("Geen priemgetal")
```

Lus B

Lus A

Waarom staat priemgetal == True in Lus A, maar voordat Lus B gestart wordt?

Omdat deze telkens nadat gecontroleerd is of de ingave een priemgetal was, deze opnieuw op True moet gezet worden, anders kan lus B soms niet starten

Oefening 3.9

Schrijf een script dat aan de gebruiker vraagt een natuurlijk getal in te geven van 1 tot en met 10.

Het script toont dan alle tafels vermenigvuldiging die kleiner of gelijk zijn aan het ingegeven getal.

Een voorbeeld is gegeven.

```
Geef een natuurlijk getal van 1 tot/met 10: 2
Tafel van 1
1 x 1 = 1
2 x 1 = 2
3 x 1 = 3
4 x 1 = 4
5 x 1 = 5
6 x 1 = 6
7 x 1 = 7
8 x 1 = 8
9 x 1 = 9
10 x 1 = 10
Tafel van 2
1 x 2 = 2
2 x 2 = 4
3 x 2 = 6
4 x 2 = 8
5 x 2 = 10
6 x 2 = 12
7 x 2 = 14
8 x 2 = 16
9 x 2 = 18
10 x 2 = 20
>>>
```

```
""" Oefening 3.9
    Verschillende tafels van vermenigvuldiging printen
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

max = 0
tafel = 1
while max <1 or max>10:
    max = int(input("Geef een natuurlijk getal van 1 tot/met 10: "))

while tafel<=max:
    x = 1
    print("De tafel van",tafel)
    while x <= 10:
        print(x,"x",tafel,"=",x*tafel)
        x = x+1
    tafel =  tafel + 1
```

Oefening 3.10

Schrijf een script dat aan de gebruiker een natuurlijk getal vraagt. Het script print dan alle priemgetallen, kleiner of gelijk aan dit getal.

Het script blijft dit herhalen totdat je getal "0" ingeeft.

```
""" Oefening 3.10
```

```
    Verschillende tafels van vermenigvuldiging printen
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

max = int(input("Geef een natuurlijk getal: "))
while max!=0:
    teller = 3
    while teller<=max:
        x = 2
        while x <= teller**.5 and teller%x!=0:
            x = x + 1
        if x>=teller**.5:
            print(teller)
        teller=teller + 1
    max = int(input("Geef een natuurlijk getal: "))
```

Oefening 3.11

Elk getal kan voorgesteld worden als sommen van Fibonacci-getallen, maar er is slechts één representatie die elk Fibonacci-getal maximaal één keer gebruikt en opeenvolgende paren van Fibonacci-getallen vermijdt; deze unieke voorstelling staat bekend als de Zeckendorf-voorstelling.

Schrijf een script dat aan de gebruiker een natuurlijk getal vraagt en dan de Zeckendorf-voorstelling van dit getal geeft.

```
""" Oefening 3.11
    Zeckendorf
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

max = int(input("Geef een natuurlijk getal in: "))

while max>1:
    fib1 = 0
    fib2 = 1
    while fib2<=max:
        fib3 = fib1 + fib2
        fib1 = fib2
        fib2 = fib3
    print(fib1)
    max = max - fib1
print(max)
```

Module 4: lijsten

Opdracht 4.1

Neem een dobbelsteen en gooi er 20 keren mee. Hou bij hoeveel keer elk aantal ogen gegooid is.

Als je geen dobbelsteen hebt, dan kan je met onderstaande code een dobbelsteen simuleren.

```
#Dobbelsteen
import random
tel = 0
while tel<=20:
    print("Beurt: ",tel)
    print("Aantal ogen gegooid:",random.randint(1,6))
    wacht=input("druk op ENTER voor volgende beurt")
    print()
    tel=tel+1

print("Einde")
```

Je kan in onderstaande tabel de gegooide waarden turven (streepjes zetten achter het ogen aantal).

Ogen	Aantal
1	
2	
3	

Ogen	Aantal
4	
5	
6	

Opdracht 4.2

Ontwerp een algoritme en vertaal dit in een Python script waarmee je dezelfde opdracht als opdracht 4.1 kunt uitvoeren. Na het ingeven van de data print het script een overzicht van het aantal keren dat elk aantal ogen gegooid is.

Oplossing:

```
#opdracht 4.2

tel = 0
aantal_1 = 0
aantal_2 = 0
aantal_3 = 0
aantal_4 = 0
aantal_5 = 0
aantal_6 = 0

while tel<20:
    tel = tel + 1
    print("Beurt:",tel)
```

```
    ogen = int(input("Aantal ogen gegooid: "))
    if ogen==1:
        aantal_1 = aantal_1 + 1
    else:
        if ogen==2:
            aantal_2 = aantal_2 + 1
        else:
            if ogen==3:
                aantal_3 = aantal_3 + 1
            else:
                if ogen==4:
                    aantal_4 = aantal_4 + 1
                else:
                    if ogen == 5:
                        aantal_5 = aantal_5 + 1
                    else:
                        aantal_6 = aantal_6 + 1
print()
print("Aantal keren gegooid:")
print(" 1 oog: ",aantal_1)
print(" 2 ogen:",aantal_2)
print(" 3 ogen:",aantal_3)
print(" 4 ogen:",aantal_4)
print(" 5 ogen:",aantal_5)
print(" 6 ogen:",aantal_6)
```

Opdracht 4.3

Bekijk onderstaande instructies. Schrijf wat je verwacht dat op ❶, ❷ en ❹ gaat geprint worden.

Welke instructie moet op ❸ staan zodat je ***Mei Lin*** (de eerste naam uit de lijst) print.

```
>>> namen=["Mei Lin","Javier","Aisha","Akio","Isabella","Joren"]
>>> print(namen[1])
          ❶
.....
>>> print(namen[5])
          ❷
.....
>>> .....❸.....
 Mei Lin
>>> print(namen[6])
          ❹
.....
```

❶ Janvier

❷ Joren

❸ print(namen[0])

❹ FOUTMELDING: Index 6 is te groot. De lijst heeft 6 elementen met indexen die gaan van 0 tot/met 5

Opdracht 4.4

Start met onderstaande lijst en print op het scherm de bloemnamen die in het vet (bold) zijn aangeduid.

Je vindt de lijst terug in op de ondersteunende website

Python 3de graad info Mod4 Opdracht 4.4 – Code

```
bloem = ["tulp","anjer","roos","dahlia","lelie","madeliefje",
"boterbloem","begonia","freesia","goudsbloem"]
```

Een voorbeeld van je output is gegeven.

```
['anjer', 'dahlia', 'madeliefje', 'begonia', 'goudsbloem']
>>>
```

Oplossing

```
""" Opdracht 4.4
    Lijst printen - eentje overslaan
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

bloem = ["tulp","anjer","roos","dahlia","lelie",
         "madeliefje", "boterbloem","begonia","freesia","goudsbloem"]
nieuw = []
i = 1
while i<=9:
    nieuw = nieuw + [bloem[i]]
    i = i + 2

print(nieuw)
```

Opdracht 4.5

Voer onderstaande bewerkingen uit in de shell. Je kan het effect controleren door al de elementen van de lijst op het scherm te tonen door de naam van de lijst te tonen. Omschrijf wat de bewerking doet.

Let op, sommige van deze bewerkingen zijn foutief en geven een foutboodschap.

Op de ondersteunende website vind je de code terug.
Door ze één per één te kopiëren naar de shell kan je ze makkelijk uitvoeren.

Python 3de graad info Mod4 Opdracht 4.5 – Code

Begindata en uit te voeren instructies:

```
jongens = ["Mateo","Liam","Joren"]
meisjes = ["Sofia","Elise"]
priem = [2,3,5,7]
volgend = 11
leeg=[]
leeg = leeg +["Niet meer leeg"]
personen = jongens + meisjes
meisjes_plus = meisjes + "Kathleen"
jongens_plus = jongens + ["Erik"]
dubbel_meisjes = meisjes * 2
vol = leeg + dubbel_meisjes
dubbel_priem = 2*priem
priem_plus = priem + volgend
priem_plus = priem + [volgend]
jongens[1]="Pieter"
```

Oplossing

```
>>> %Run -c $EDITOR_CONTENT
>>> leeg = leeg + ["Niet meer leeg"]
>>> leeg
['Niet meer leeg']
>>> personen =  jongens + meisjes
>>> personen
['Mateo', 'Liam', 'Joren', 'Sofia', 'Elise']
>>> meisjes_plus = meisjes + "Kathleen"
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: can only concatenate list (not "str") to list
>>> jongens_plus = jongens +  ["Erik"]
>>> jongens_plus
['Mateo', 'Liam', 'Joren', 'Erik']
```

```
>>> dubbel_meisjes = meisjes * 2
>>> dubbel_meisjes
['Sofia', 'Elise', 'Sofia', 'Elise']
>>> vol = leeg + dubbel_meisjes
>>> vol
['Niet meer leeg', 'Sofia', 'Elise', 'Sofia', 'Elise']
>>> dubbel_priem = 2 * priem
>>> dubbel_priem
[2, 3, 5, 7, 2, 3, 5, 7]
>>> priem_plus = priem + volgend
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: can only concatenate list (not "int") to list
>>> priem_plus = priem + [volgend]
>>> priem_plus
[2, 3, 5, 7, 11]
```

```
>>> jongens[1] = "Pieter"
>>> jongens
['Mateo', 'Pieter', 'Joren']
>>>
```

Opdracht 4.6

Vul onderstaande code aan zodat je een lijstje kan ingeven.

Wanneer je op ENTER drukt zonder iets in te vullen wordt het lijstje afgesloten.

Tenslotte geef je het volledige lijstje weer op het scherm.

```
#Opdracht 4.6
boodschappen=[]
print("Geef je boodschappen in.")
print("Eindig je lijstje door gewoon op ENTER te drukken.")
print()

toevoegen=input("Volgende boodschap: ")

while toevoegen !="":
    # code hier aanvullen
    # zorg ervoor dat je niet in een oneindige lus terecht komt.
```

Voorbeeld:

```
Geef je boodschappen in.
Eindig je lijstje door gewoon op ENTER te drukken.

Volgende boodschap: boter
Volgende boodschap: jam
Volgende boodschap: suiker
Volgende boodschap:
['boter', 'jam', 'suiker']

>>>
```

Oplossing:

```
""" Oefening 4.6
    Boodschappenlijstje
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """
boodschappen=[]
print("Geef je boodschappen in.")
print("Eindig je lijstje door gewoon op ENTER te drukken.")
print()

toevoegen=input("Volgende boodschap: ")

while toevoegen !="":
    boodschappen = boodschappen + [toevoegen]
    toevoegen=input("Volgende boodschap: ")

print(boodschappen)
```

Opdracht 4.7

Bij een experiment wordt om de 0,02 seconden de positie van een vallend voorwerp bepaald. In totaal heb ik 30 metingen, maar je wilt de eerste 3 en de laatste 4 metingen niet gebruiken vanwege het overgangsverschijnsel.

```
metingen=[0.0000, 0.0019, 0.0078, 0.0176, 0.0314, 0.0491, 0.0706, 0.0961, 0.1256,
0.1589, 0.1962, 0.2374, 0.2825, 0.3316, 0.3846, 0.4415, 0.5023, 0.5670, 0.6357,
0.7083, 0.7848, 0.8652, 0.9496, 1.0379, 1.1301, 1.2263, 1.2263, 1.2263, 1.2263,
1.2263]
```

Schrijf een script zodat je een (nieuwe) lijst krijgt zonder deze metingen.

Je nieuwe lijst moet ze zo uitzien.

```
nieuw=[0.0176, 0.0314, 0.0491, 0.0706, 0.0961, 0.1256, 0.1589, 0.1962, 0.2374,
0.2825, 0.3316, 0.3846, 0.4415, 0.5023, 0.5670, 0.6357, 0.7083, 0.7848, 0.8652,
0.9496, 1.0379, 1.1301, 1.2263]
```

```
""" Oefening 4.6
    Lijst, voor en achterkant wegsnijden
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

metingen=[0.0000, 0.0019, 0.0078, 0.0176, 0.0314, 0.0491, 0.0706,
          0.0961, 0.1256, 0.1589, 0.1962, 0.2374, 0.2825, 0.3316,
          0.3846, 0.4415, 0.5023, 0.5670, 0.6357, 0.7083, 0.7848,
          0.8652, 0.9496, 1.0379, 1.1301, 1.2263, 1.2263, 1.2263,
          1.2263, 1.2263]

nieuw = []

i = 3
while i<26:
    nieuw = nieuw + [metingen[i]]
    i = i+1

metingen = nieuw
print(metingen)
```

Opdracht 4.8

Vraag aan de gebruiker om een letter in te geven.

Controleer of de ingegeven letter een klinker is.
Indien het een klinker geeft het script als output "Klinker", anders "Medeklinker".

Voorbeeld:

```
Geef letter in: a
Klinker
>>>
```

```
Geef letter in: x
Medeklinker
>>>
```

```
""" Oefening 4.8
    klinker of medeklinker
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

klinkers=["a","e","i","o","u"]
medeklinker = True
i = 0

letter = input("Geef letter in: ")

while medeklinker and i<=4:
    if klinkers[i]==letter:
        medeklinker = False
    i = i+1
```

```
if not medeklinker:
    print('Klinker')
else:
    print('Medeklinker')
```

Uitbreiding: "in"

Je kunt testen of een element voorkomt in een list via de ***in*** operator (of het niet voorkomen van een element via de ***not in*** operator).

```
>>> richting=["links","rechts","boven","onder"]
>>> "links" in richting
True
>>> "schuin" in richting
False
>>> "schuin" not in richting
True
>>>
```

Oefening 4.1

Pas het script van opdracht 4.2 (gooien van de dobbelsteen) aan zodat er gebruik gemaakt wordt van lijsten.

```
""" Oefening 4.1
    simulatie dobbelsteen - resultaat in lijsten

    Auteur: Jurgen Nijs - UHasselt
    Datum: 12/2/24                          """
import random
tel = 0

#lijst gevuld met 6 nullen, eentje voor elk cijfer
# kan ook aantal = [0,0,0,0,0,0]
aantal=[0]*6

while tel<20:
    tel = tel + 1
    ogen = random.randint(1,6)
    aantal[ogen-1]=aantal[ogen-1]+1


tel = 0
print()
print("Aantal keren gegooid:")
print(" ogen       aantal")
while tel <6:
    #print(" ogen       aantal")
    print(" ",tel+1,"       ",aantal[tel])
    tel=tel+1
```

Voor snelle leerlingen kan je vragen dat er wordt ingegeven hoeveel keer de dobbelsteen gegooid wordt. Dit is slechts een zeer kleine uitbreiding en goed om even een klein verschil op te vangen.

```
""" Oefening 4.1 uitbreiding
    simulatie dobbelsteen - resultaat in lijsten
    Uitbreiding: Gevraagd aantal keren gooien
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """
import random
tel = 0
aantal=[0]*6 #lijst gevuld met 6 nullen, eentje voor elk cijfer

totaal = int(input("Hoeveel keer moet de dobbelsteen gegooid worden? "))

while tel<totaal:
    tel = tel + 1
    ogen = random.randint(1,6)
    aantal[ogen-1]=aantal[ogen-1]+1


tel = 0
print()
print("Aantal keren gegooid:")
print(" ogen       aantal")
while tel <6:
    #print(" ogen       aantal")
    print(" ",tel+1,"       ",aantal[tel])
    tel=tel+1
```

Oefening 4.2

Vervolledig onderstaand script zodat de gegeven lijst gesorteerd wordt. Bekijk voor inspiratie oefening 2.9.

```
# oefening 4.2
Getal=[7,5]
# vul hier je code aan
# Output
print(getal[0],"is kleiner dan",getal[1])
```

```
# oefening 4.2
getal=[7,5]
# vul hier je code aan

if getal[0]>getal[1]:
    hulp = getal[0]
    getal[0] = getal[1]
    getal[1] = hulp

# Output
print(getal[0],"is kleiner dan",getal[1])
```

Uitbreiding: laat de leerlingen zelf twee getallen ingeven in een lijst met een input-functie.

```
""" Oefening 4.2 uitbreiding
    Sorteren lijst met 2 elementen
    Uitbreiding: ingeven met input
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

getal=[]
# vul hier je code aan
getal = getal+[float(input("Geef getal 1: "))]
getal = getal+[float(input("Geef getal 2: "))]

if getal[0]>getal[1]:
    hulp = getal[0]
    getal[0] = getal[1]
    getal[1] = hulp

# Output
print(getal[0],"is kleiner dan",getal[1])
```

Oefening 4.3

Pas het script van oefening 4.2 aan zodat de gebruiker zelf de twee getallen kan ingeven.

Je mag wel slechts één input-opdracht gebruiken.

```
""" Oefening 4.3
    Sorteren lijst met 2 elementen
    ingeven lijst met 1 input
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

getal=[]
# vul hier je code aan
i = 0 # Wegwerpvariabelen niet betekenisvol
while i<=1:
    prompt = "Geef getal "+str(i+1)+": "
    getal = getal+[float(input(prompt))]
    i+=1 #Dit komt overeen met i = i + 1

if getal[0]>getal[1]:
    hulp = getal[0]
    getal[0] = getal[1]
    getal[1] = hulp

# Output
print(getal[0],"is kleiner dan",getal[1])
```

Oefening 4.4

Vervolledig onderstaand script zodat de tekst die je gaat ingeven toegevoegd wordt aan de lijst.

```
# Oefening 4.4
```

```
# initialisatie
ingave=" "
eten=["friet","frikandel","puree"]
# gebruikersinformatie
print("Vul je favoriet eten aan.")
print("Om te stoppen druk gewoon op enter (zonder een getal in te geven)")
print()
# ingeven data
while ingave!="":
 ingave=input("Jouw favoriet eten: ")
 if ingave!="":
     # Vul in deze if-functie in wat je gaat toevoegen als je
     # iets hebt ingegegeven. (Als ingaven niet leeg is)
     #.....................
# output
print()
print("De volledige lijst is...")
print(eten)
```

```
# Oefening 4.4
# initialisatie
ingave=" "
eten=["friet","frikandel","puree"]
# gebruikersinformatie
print("Vul je favoriet eten aan.")
print("Om te stoppen druk gewoon op enter (zonder een getal in te geven)")
print()
# ingeven data
while ingave!="":
 ingave=input("Jouw favoriet eten: ")
 if ingave!="":
     eten = eten +[ingave]

# output
print()
print("De volledige lijst is...")
print(eten)
```

Oefening 4.5

Schrijf een script dat 10 willekeurige natuurlijke getallen genereert en deze toont op het scherm.

Het script vraagt dan aan de gebruiker welk (het hoeveelste getal in de rij), het moet aanpassen gevolgd door het nieuwe getal.

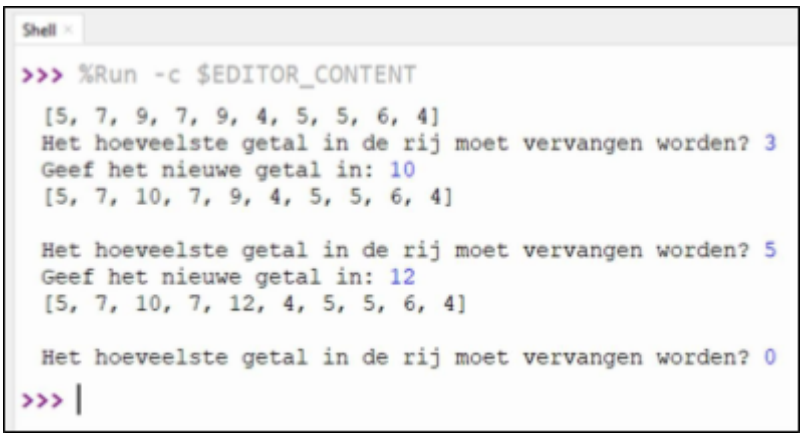
Je toont telkens de nieuwe cijfers. (Zie voorbeeld)

Herhaal dit totdat je 0 ingeeft bij de vraag welk getal je wenst aan te passen.

Onderstaande code genereert een getal tussen 1 en 10. Je kan deze code gebruiken om je probleem op te lossen.

```
import random
getal=random.randint(1,10)
```

Voorbeeld:




```
""" Oefening 4.5
    Lijst generen en aanpassen via index

    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

import random
getallen=[]
i = 0
while i<10:
    getallen = getallen+[random.randint(1,10)]
    i+=1

print(getallen)

nr = int(input("Het hoeveelste getal in de rij moet vervangen worden? "))
while nr>0:
    getallen[nr-1] = int(input("Geef het nieuwe getal in: "))
    print(getallen)
    print()
    nr = int(input("Het hoeveelste getal in de rij moet vervangen worden? "))
```

Oefening 4.6

Schrijf een script dat aan de gebruiker vraagt om een natuurlijk getal in te geven. Het script gaat dan zoveel willekeurige getallen tussen 1 en 10 genereren.

Als output geeft het script de gegenereerde getallen in oplopende volgorde.

```
""" Oefening 4.6
    Sorteren
    Auteur: Jurgen Nijs - UHasselt
    Datum: 11/2/24                         """

import random
getal=[]
i = 0

while i<10:
    getal = getal+[random.randint(1,10)]
    i+=1
```

```
print(getal)
j = 10
# Sorteren
while j>1:
    i = 0
    while i<j-1:
        #print(getal[i],getal[i+1])
        if getal[i]>getal[i+1]:
            hulp = getal[i]
            getal[i] = getal[i+1]
            getal[i+1] = hulp
        i+=1
    j-=1

print("Gesorteerd: ")
print(getal)
```

Oefening 4.7

Schrijf een script dat aan de gebruiker vraagt om een natuurlijk getal in te geven. Het script gaat dan zoveel willekeurige getallen tussen 1 en 10 genereren.

Als output geeft het script de gegenereerde getallen en de mediaan.

Mediaan:

De mediaan is de middelste waarde van een groep getallen die gerangschikt wordt volgens grootte. Het is het getal dat exact in het midden ligt zodat 50% van de gerangschikte getallen boven 50% ligt en 50% onder de mediaan.

Voorbeeld

Om de mediaan te vinden van dezelfde 9 getallen: 10, 12, 11, 15, 13, 35, 41, 23, 20, plaats je ze eerst in stijgende volgorde, d.w.z. 10, 11, 12, 13, 15, 20, 23, 35, 41

Het middelste getal is 15: de mediaan is 15, omdat 4 getallen onder 15 liggen en 4 getallen boven 15 liggen.

Als er een even aantal getallen is: 10, 11, 12, 13, 15, 20, 23, 35 - de twee in het midden (13 en 15) worden opgeteld (13+15=28) en dan gedeeld door 2 (28/2= 14), dat betekent dat de mediaan in dit geval 14 is.

```
""" Oefening 4.7
    Mediaan bepalen
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

import random
getal=[]
i = 0

while i<10:
    getal = getal+[random.randint(1,10)]
    i+=1

print(getal)
```

```
j = 10
# Sorteren
while j>1:
    i = 0
    while i<j-1:
        #print(getal[i],getal[i+1])
        if getal[i]>getal[i+1]:
            hulp = getal[i]
            getal[i] = getal[i+1]
            getal[i+1] = hulp
        i+=1
    j-=1

print("Gesorteerd: ")
print(getal)

mediaan = (getal[4]+getal[5])/2
print("Mediaan: ",mediaan)
```

Uitbreiding: Laat de gebruiker de lengte van de lijst ingeven.

```
""" Oefening 4.7 (uitbreiding)
    Mediaan bepalen
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

import random
getal=[]
i = 0

totaal = int(input("Hoeveel getallen genereren? "))

while i<totaal:
    getal = getal+[random.randint(1,10)]
    i+=1

print(getal)
j = totaal
# Sorteren
while j>1:
    i = 0
    while i<j-1:
        #print(getal[i],getal[i+1])
        if getal[i]>getal[i+1]:
            hulp = getal[i]
            getal[i] = getal[i+1]
            getal[i+1] = hulp
        i+=1
    j-=1

print("Gesorteerd: ")
print(getal)

midden = totaal//2

if totaal%2==0:
```

```
    mediaan = (getal[midden-1]+getal[midden])/2
else:
    mediaan = getal[midden]

print("Mediaan: ",mediaan)
```

Een alternatief om de mediaan te berekenen is

```
y1 = int(totaal/2+.5)-1
y2 = int(totaal/2+1)-1
mediaan = (getal[y1]+getal[y2])/2
```

Bij oneven getallen is y1 = y2, en bij even getallen is y2 = y1+1.

Oefening 4.8:

Schrijf een Python script dat een lijst genereerd met de eerste 50 priemgetallen.

Een priemgetal is een getal dat enkel door 1 en zichzelf deelbaar is.

Je kan deze lijst genereren door te kijken of een getal een priemgetal is. Wanneer het hieraan voldoet voeg je dat toe aan de lijst, of je kan gebruik maken van de zeef van Euclides. Je start met alle getallen en gooit er die getallen uit die geen priemgetallen zijn.

Hieronder zie je een voorbeeld. Je verwijderd uit de lijst het getal 4.

```
getallen=[1,2,3,4,5]
getallen.remove(4)
```

```
>>> getallen=[1,2,3,4,5]
>>> getallen.remove(4)
>>> getallen
[1, 2, 3, 5]
>>>
```

En dit voorbeeld verwijder de letter C

```
letters=["A","B","C","D","E"]
letters.remove("C")
```

```
>>> letters=["A","B","C","D","E"]
>>> letters.remove("C")
>>> letters
['A', 'B', 'D', 'E']
>>>
```

```
"""
    Oefening 4.8
    Priemgetallen
    Auteur: Jurgen Nijs - UHasselt
    Datum: 12/2/24                          """

priem= []
j = 0
getal=2
while j<50:
    i = 2
    while i<=getal**0.5 and getal%i!=0:
        i = i + 1

    if i>getal**0.5:
        priem=priem+[getal]
```

```
        j = j +1
    getal =  getal + 1

print(priem)
```

Oplossing Zeef van Euclides

```
"""
    Oefening 4.8
    Priemgetallen - Zeef van Euclides

    Auteur: Jurgen Nijs - UHasselt
    Datum: 12/2/24                          """

priem = []
i = 2
while i<=2000:
    priem=priem + [i]
    i+=1

i = 1
while i<50:
    j = priem[i-1]
    k = j
    while j<=2000:
        j+=k
        if j in priem:
            priem.remove(j)
    i = i+1

priem = priem[:50]
print(priem)
```

Oefening 4.9

Een coopertest is een oefening waarbij de conditie van een deelnemer wordt gemeten. De bekendste vorm daarvan is die waarin een hardloper in 12 minuten een zo groot mogelijke afstand aflegt. Met deze afstand en de leeftijd kan in onderstaande tabel de conditie afgelezen worden.

Leeftijd		Zeer zwak	Zwak	Matig	Voldoende	Ruim voldoende	goed
20 - 29	M	<1916	1916 - 2156	2157 - 2333	2334 - 2478	2479 - 2655	2656 - 2911
	V	<1658	1658 - 1866	1867 - 2011	2012 -2140	2141 - 2333	2334 - 2589
30 - 39	M	<1884	1884 - 2076	2077 - 2237	2238 - 2397	2398 - 2590	2591 - 2847
	V	<1626	1626 - 1786	1787 - 1947	1948 - 2043	2044 - 2220	2221 - 2461
40 - 49	M	<1771	1771 - 1979	1980 - 2140	2141 - 2285	2286 - 2478	2479 - 2750
	V	<1546	1546 - 1689	1690 - 1818	1819 - 1947	1948 - 2124	2125 - 2332
50 - 59	M	<1626	1626 - 1850	1851 - 2011	2012 - 2140	2141 - 2333	2334 - 2606
	V	<1449	1449 - 1577	1578 - 1706	1707 - 1818	1819 - 1947	1948 - 2139
60 - 69	M	<1433	1433 - 1689	1690 - 1850	1851 - 1995	1996 - 2204	2205 - 2525
	V	<1385	1385 - 1512	1513 - 1593	1594 - 1722	1723 - 1899	1900 - 2071
70 - 79	M	<1272	1272 - 1520	1521 - 1665	1666 - 1793	1794 - 1985	1986 - 2265
	V	<1205	1205 - 1337	1338 - 1409	1410 - 1520	1521 - 1638	1639 - 1835
80 - 90	M	<1000	1000 - 1232	1233 - 1348	1349 - 1450	1451 - 1602	1603 - 1828
	V	<975	975 - 1082	1083 - 1184	1185 -1278	1279 - 1375	1376 - 1537

Ontwerp voor de mannen in leeftijdscategorie 20-29 een script dat je een antwoord geeft op de vraag hoe de conditie van een testpersoon is.

Denk hierbij goed na over welke lijsten je gaat gebruiken.

```
""" Oefening 4.9
    Coopertest voor mannen - twintigers

    Auteur: Jurgen Nijs - UHasselt
    Datum: 12/2/24                          """

m20 = [0,1916,2156,2333,2478,2655]
resultaten =["zeer zwak","zwak","matig","voldoende","ruim
voldoende","uitstekend"]

i = 0
gelopen = int(input("Geef de gelopen afstand: "))

while i<6 and gelopen>m20[i]:
    i = i+1

print("De conditie is:",resultaten[i-1])
```

Oefening 4.10

Bij oefening 4.11 heb je de Zeckendorf-voorstelling van een getal gegeven, waarbij je gebruik maakte van geneste lussen. Pas dit script aan zodat je gebruik kunt maken van een lijst van Fibonacci-getallen.

```
""" Oefening 4.10
    Zeckendorf - met lijsten
    Auteur: Jurgen Nijs - UHasselt
    Datum: 9/2/24                          """

fibonacci=[0,1]
i = 2

while i <100:
```

```
    fibonacci = fibonacci+[fibonacci[i-1]+fibonacci[i-2]]
    i=i+1

max = int(input("Geef een natuurlijk getal in: "))

while max>1:
    i = 0
    while fibonacci[i+1]<=max:
        i = i+1
    print(fibonacci[i])
    max = max - fibonacci[i]
print(max)
```

Oefening 4.11

Ontwerp voor de mannen en vrouwen in leeftijdscategorie 20-29 een script dat je een antwoord geeft op de vraag hoe de conditie van een testpersoon is.

Denk hierbij goed na over welke lijsten je gaat gebruiken.

Gebruik de data van oefening 4.9.

```
""" Oefening 4.11
    Coopertest - twintigers

    Auteur: Jurgen Nijs - UHasselt
    Datum: 12/2/24                          """

m20 = [0,1916,2156,2333,2478,2655]
v20 = [0,1658,1866,2011,2140,2333]
resultaten =["zeer zwak","zwak","matig","voldoende","ruim
voldoende","uitstekend"]

i = 0
gelopen = int(input("Geef de gelopen afstand: "))
sex = input("Man (m) of vrouw (v): ")
```

```
if sex =="m":
    cooper = m20
else:
    cooper = v20

while i<6 and gelopen>cooper[i]:
    i = i+1

print("De conditie is:",resultaten[i-1])
```

Opdracht 4.8

Met welke bewerking kan ik de bbq lijst uitbreiden met de gegevens van An (zie onderstaande tabel)

Naam	Kip	Vis	Worst	Spek
Piet	1	1	0	2
Jan	0	1	1	2
An	2	1	0	0

Mogelijkheid 1:

```
bbq = bbq + ["An", 2, 1, 0, 0]
```

Mogelijkheid 2:

```
bbq = bbq + [["An", 2, 1, 0, 0]]
```

Controleer je antwoord.

Mogelijkheid 2

Oefening 4.12: BBQ

Maak een script dat aan de verbruiker vraagt om volgende informatie:

- Naam
- Aantal keer kip, vis, worst en spek

Het ingeven stopt als er geen naam wordt ingegeven (en gewoon op ENTER gedrukt wordt).

```
""" Oefening 4.12
    BBQ
    Auteur: Jurgen Nijs - UHasselt
    Datum: 12/2/24                          """

bbq =[]

naam = input("Naam: ")

while naam!="":
    #kip, vis, worst en spek
    kip = int(input("  Kip: "))
    vis = int(input("  Vis: "))
    worst = int(input("  Worst: "))
    spek = int(input("  Spek: "))
```

```
    bbq=bbq+[[naam,kip,vis,worst,spek]]
    naam = input("Naam: ")
```

Oefening 4.13: BBQ+

Breidt het script van oefening 4.12 uit zodat op het einde het totaal aantal keer kip, vis, worst en spek berekent wordt en verschijnt op het scherm.

```
""" Oefening 4.13
    BBQ+
    Auteur: Jurgen Nijs - UHasselt
    Datum: 12/2/24                          """

bbq =[]

naam = input("Naam: ")

while naam!="":
    #kip, vis, worst en spek
    kip = int(input("  Kip: "))
    vis = int(input("  Vis: "))
    worst = int(input("  Worst: "))
    spek = int(input("  Spek: "))
    bbq=bbq+[[naam,kip,vis,worst,spek]]
    naam = input("Naam: ")

soort = 1
aantal =["Kip",0,"Vis",0,"Worst",0,"Spek",0]
while soort<=4:
    klant=0
    while klant<len(bbq):
        aantal[2*soort-1]=aantal[2*soort-1]+bbq[klant][soort]
        klant = klant +1
    soort=soort+1
print(aantal)
```

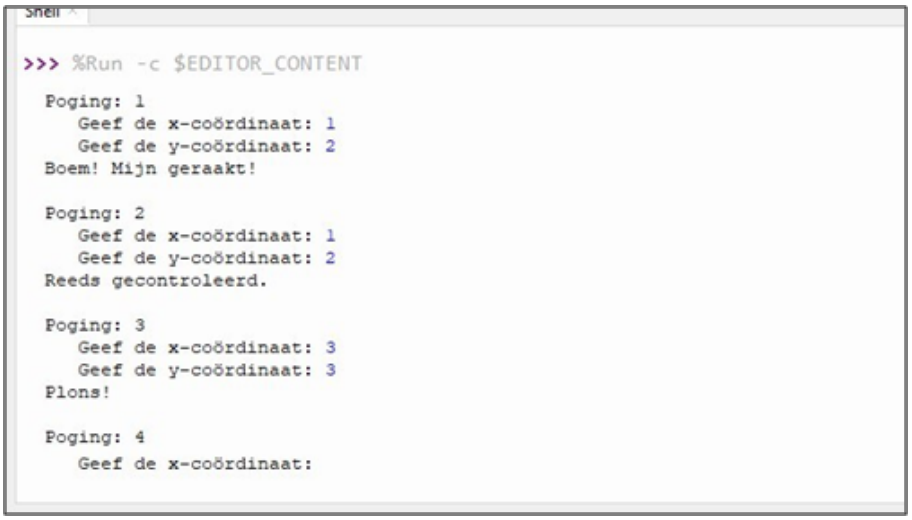
Oefening 4.14: Mijnenjagen

Maak een script dat in een 6 x 6 grid willekeurig 4 vakjes aanduidt. De bedoeling van het spel is om deze mijnen onschadelijk te maken door ze te laten ontploffen.

Het script vraagt aan de gebruiker de coördinaten van een vakje. Afhankelijk van het resultaat geeft het volgend antwoord:

- “Boem! Mijn geraakt”
- “Plons!”
- “Reeds gecontroleerd”

Het script geeft ook het aantal pogingen weer en stopt als al de mijnen ontploft zijn.




```
""" Oefening 4.14
    Mijnenjagen
    Auteur: Jurgen Nijs - UHasselt
    Datum: 12/2/24                          """

import random


mijn =[['-','-','-','-','-','-'],['-','-','-','-','-','-']
       ,['-','-','-','-','-','-'],['-','-','-','-','-','-'],
       ['-','-','-','-','-','-'],['-','-','-','-','-','-']]
aantal = 0

while aantal < 4:
    x = random.randint(0,5)
    y = random.randint(0,5)
    if mijn[x][y]=='-':
        mijn[x][y]='M'
        aantal = aantal + 1

for i in range(5):
    print(mijn[i])

poging = 0

while aantal>0:
    print("Poging:",poging)
    poging = poging+1
    x = int(input("Geef de x-coördinaat: "))
    y = int(input("Geef de y-coördinaat: "))
    if mijn[x-1][y-1]=="G":
        print("Reeds gecontroleerd")
    else:
        if mijn[x-1][y-1]=="M":
            print("Boem! Mijn geraakt.")
            aantal=aantal-1
        else:
```

```
            print("Plons")
        mijn[x-1][y-1]="G"
    print()

print("Alle mijnen gevonden in",poging,"pogingen.")
```

Oefening 4.15: Mijnenjagen+

Breidt het script van oefening 4.14 uit zodat twee vakjes die elkaar raken (zijde of hoek) allebei geen mijn kunnen bevatten (slechts één van de twee kan een mijn bevatten)

Omdat het controleren aan de kanten een probleem kan opleveren maken we een kader rond het vak met dummy data. Op die manier kan je op dezelfde manier de vakjes aan de kant controleren als de vakjes in het midden.

```
""" Oefening 4.15
    Mijnenjagen +
    Auteur: Jurgen Nijs - UHasselt
    Datum: 12/2/24                          """

import random


mijn =[['/','/','/','/','/','/','/','/'],['/','-','-','-','-','-','-','/'],
       ['/','-','-','-','-','-','-','/'],['/','-','-','-','-','-','-','/'],
       ['/','-','-','-','-','-','-','/'],['/','-','-','-','-','-','-','/'],
       ['/','-','-','-','-','-','-','/'],['/','/','/','/','/','/','/','/']]
aantal = 0

while aantal < 8:
    x = random.randint(1,6)
    y = random.randint(1,6)
    print(x,y)
    print("------")
    if mijn[x][y]=='-':
        i = x-1
        j = y-1
        inorde = True
        while i <=x+1 and inorde:
            j = y - 1
            while j <=y+1 and inorde:
                print(i,j)
                if mijn[i][j]=="M":
                    inorde=False
                j = j+1
            i = i+1
        if inorde:
            aantal = aantal + 1
            mijn[x][y]="M"


for i in range(8):
```

```
    print(mijn[i])

poging = 0

while aantal>0:
    print("Poging:",poging)
    poging = poging+1
    x = int(input("Geef de x-coördinaat: "))
    y = int(input("Geef de y-coördinaat: "))
    if mijn[x][y]=="G":
        print("Reeds gecontroleerd")
    else:
        if mijn[x][y]=="M":
            print("Boem! Mijn geraakt.")
            aantal=aantal-1
        else:
            print("Plons")
        mijn[x][y]="G"

print("Alle mijnen gevonden in",poging,"pogingen.")
```

Module 5: String

Opdracht 5.1

Bij lijsten kunnen we elementen selecteren m.b.v. de index.

```
#opdracht 5.1 (a)
weekdagen=["Ma","Di","Wo","Do","Vr","Za","Zo"]
print(weekdagen[4])
```

Wat verwacht je dat de bovenstaande code als resultaat heeft?

Wat verwacht je dat onderstaande instructie als resultaat gaat geven? Als je denkt dat deze instructie niet kan gebruikt worden bij strings, dan is je antwoord "Foutmelding".

```
#opdracht 5.1 (b)
weekdagen="Ma, Di, Wo, Do, Vr, Za, Zo"
print(weekdagen[4])
```

Juist antwoord: "D"

Opdracht 5.2

Je kan in een lijst met de index van een element individuele elementen aanpassen.

Wat verwacht je dat onderstaande instructie als resultaat gaat geven? Als je denkt dat deze instructie niet kan gebruikt worden bij strings, dan is je antwoord "Foutmelding".

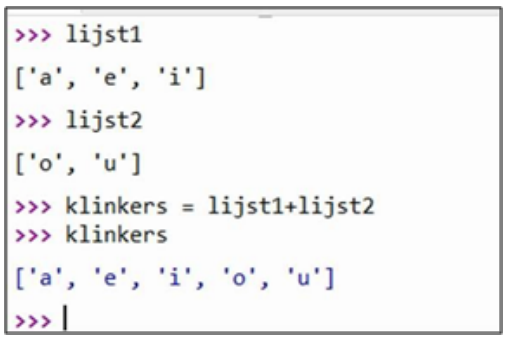
```
# opdracht 5.2
zin="Met deze zin gaan we op onderzoek."
print(zin[4])
print(zin[5:10])
```

Oplossing

d
eze z

Opdracht 5.3

Je kan met het '+'-teken lijsten samenvoegen.



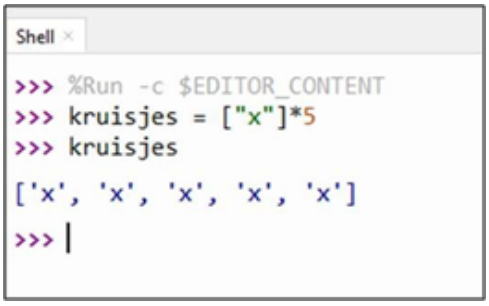
Wat verwacht je dat onderstaande instructie als resultaat gaat geven? Als je denkt dat deze instructie niet kan gebruikt worden bij strings, dan is je antwoord "Foutmelding".

```
#Opdracht 5.3
tekst1 = "De klinkers zijn: "
tekst2 = "a, e, i, o, u. "
zin = tekst1 + tekst2
print(zin)
```

Oplossing:
De klinkers zijn: a, e, i, o, u.

Opdracht 5.4

Je kan met het vermenigvuldigingsteken "*" een lijst vullen met allemaal dezelfde elementen, bijvoorbeeld:



Wat verwacht je dat onderstaande instructie als resultaat gaat geven? Als je denkt dat deze instructie niet kan gebruikt worden bij strings, dan is je antwoord "Foutmelding".

```
tekenreeks = "x"*5
print(tekenreeks)
```

Oplossing
xxxxx

Opdracht 5.5

Met de instructie "in" kunnen we controleren of een lijst een bepaald element bevat.

Bekijk onderstaande code

```
#opdracht 5.5
```

```
weekdagen=["Ma","Di","Wo","Do","Vr","Za","Zo"]
print("Ma" in weekdagen)
print("Maandag" in weekdagen)
```

of uitgevoerd in de Shell

```
>>> weekdagen=["Ma","Di","Wo","Do","Vr","Za","Zo"]
>>> len(weekdagen)
7
>>>
```

Wat verwacht je dat onderstaande instructie als resultaat gaat geven? Als je denkt dat deze instructie niet kan gebruikt worden bij strings, dan is je antwoord “Foutmelding”.

```
naam = “Python”
print(len(naam))
```

Oplossing
6

Opdracht 5.6

Onderzoek zelf wat zal gebeuren met onderstaande instructies

```
>>>zin = "Deze zin gaan we gebruiken."
>>>zin[:10]
>>>zin[10:]
>>>zin[2:12:3]
>>>zin[::]
>>>zin[12:3:-2]
>>>zin[::-1]
```

Oplossing:
Deze zin g
aan we gebruiken.
zz a
Deze zin gaan we gebruiken.
na i
.nekiurbeg ew naag niz ezeD

Opdracht 5.7

Schrijf een script dat aan de gebruiker vraagt om tekst in te geven. Het script print de verschillende karakters van de ingegeven tekst onder elkaar.

Oplossing:

```
""" Opdracht 5.7

    Auteur: Jurgen Nijs - UHasselt
    Datum: 12/2/24                          """


tekst = input("Geef tekst in: ")

i = 0

while i<len(tekst):
    print(tekst[i])
    i = i + 1
```

Opdracht 5.8

Maak een script dat aan de gebruiker vraagt een tekst in te geven, en daarna een letter. Het script geeft dan een antwoord op de vraag hoeveel keer deze letter in de tekst voorkomt.

Oplossing

```
""" Opdracht 5.8

    Auteur: Jurgen Nijs - UHasselt
    Datum: 12/2/24                          """

tekst = input('Geef tekst in: ')
letter = input('Geef een letter in: ')
tel = 0
i=0
while i<len(tekst):
    if letter==tekst[i]:
        tel=tel+1
    i=i+1
print(tel)
```

Alternatieve oplossing

```
""" Opdracht 5.8 - alternatief

    Auteur: Jurgen Nijs - UHasselt
    Datum: 12/2/24                          """

tekst = input('Geef tekst in: ')
letter = input('Geef een letter in: ')
print(tekst.count(letter))
```

Opdracht 5.9

Schrijf een script dat aan de gebruiker vraagt om een woord of zin in te geven. Het script beantwoord dan de vraag of het een palindroom of niet is.

Palindroom = woord dat omgekeerd hetzelfde woord oplevert, bijvoorbeeld LEPEL

Oplossing:

```
""" Opdracht 5.9
    Palindroom
    Auteur: Jurgen Nijs - UHasselt
    Datum: 12/2/24                          """

tekst = input('Geef woord in: ')
if tekst == tekst[::-1]:
    print("Het woord",tekst,"is een palindroom")
else:
    print("Het woord",tekst,"is GEEN palindroom")
```

Opdracht 5.10

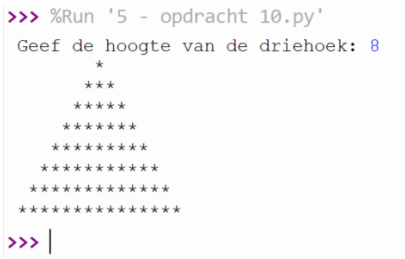
Schrijf een script dat een rechthoekige driehoek tekent. De gebruiker geeft de grootte van de rechthoekszijde in.

Oplossing:

```
""" Opdracht 5.10
    Gelijkzijdige rechthoekige driehoek
    Auteur: Jurgen Nijs - UHasselt
    Datum: 12/2/24                          """

zijde = int(input('Geef de grootte van de rechtshoekszijde: '))
t = 0
while t<zijde:
    t+=1
    print("*"*t)
```

Opdracht 5.11

Schrijf een script dat een kerstboom tekent. De gebruiker geeft de hoogte van de kerstboom in.



Oplossing

```
""" Opdracht 5.11
    Gelijkzijdige driehoek
    Auteur: Jurgen Nijs - UHasselt
    Datum: 12/2/24                          """

zijde = int(input('Geef de lengte van de gelijkzijdige rechthoekszijde: '))
t = 0
while t<zijde:
    t+=1
    print("*"*t)
```

Tip:

stel voor elke lijn een string samen bestaande uit een aantal spaties en de kruisjes.

String-methoden

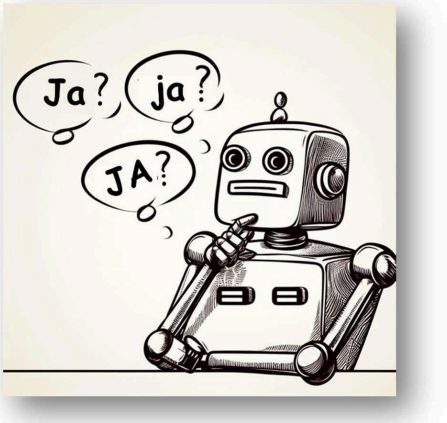
Normaal gezien gaan we niet in de toeters en bellen van Python, maar in dit geval maken we een uitzondering omdat dit sommige scripts echt wel kunnen vereenvoudigen.

upper( ) en lower( )

De string methoden upper( ) en lower( ) zetten de string respectievelijk om in hoofd of kleine letters.

```
Shell
Python 3.10.11 (C:\Users\LUCP11650\Pictures\Python\thonny-
>>> tekst = "Humpty Dumpty zat op de muur"
>>> tekst.upper()
'HUMPTY DUMPTY ZAT OP DE MUUR'
>>> tekst.lower()
'humpty dumpty zat op de muur'
>>>
```

Opdracht 5.12



Maak een script dat aan de gebruiker vraag om "ja" of "nee te antwoorden. Je controleert of de invoer één van beide mogelijkheden is, maar je aanvaard alle mogelijke schrijfwijzen (Ja, JA, ja en Nee, NEE, nee).

Maak gebruik van upper() of lower().

Indien geen van beide toegestane antwoorden gegeven wordt, dan moet je een nieuwe ingave vragen totdat één van de toegestane antwoorden ingegeven worden.

Oplossing:

```
""" Opdracht 5.10
    Gelijkzijdige rechthoekige driehoek
    Auteur: Jurgen Nijs - UHasselt
    Datum: 12/2/24                          """

toegelaten = ["ja","nee"]

antwoord = input("Wat is je antwoord: ja of nee? ")
while antwoord.lower() not in toegelaten:
    antwoord = input("Wat is je antwoord: ja of nee? ")
```

Verdieping

Oefening 5.13

Schrijf een Python-script voor een eenvoudig woordraadspel. Gebruik onderstaande stappen als leidraad:

1. Het script kiest uit een lijst met geheime woorden willekeurig een woord
2. Initialiseer een variabele om het aantal levens van de speler bij te houden. Begin met 6 levens.
3. Geef aan dat er een woord moet worden geraden.
4. Toon de speler hoeveel levens hij nog heeft.
5. Laat de speler telkens één letter raden. Vraag om invoer van de speler en controleer of de invoer een enkele letter is.
6. Controleer of de geraden letter voorkomt in het geheime woord. Als dat zo is, toon het geheime woord met de geraden letters op de juiste plaatsen.
7. Als de geraden letter niet in het geheime woord voorkomt, verminder het aantal levens van de speler en toon hoeveel levens ze over hebben.
8. Herhaal stap 4 tot en met 7 totdat de speler het geheime woord heeft geraden of geen levens meer over heeft.
9. Toon een overwinningsbericht als de speler het woord heeft geraden en een verliesbericht als de levens van de speler op zijn.
10. Vraag de speler of ze opnieuw willen spelen.

```
from random import randint
from time import sleep

woorden=
["wolkendek","schildpad","bloemenvaas","krokodil","muzieknoten","rijbewijs","koff
iemok"

,"paraplu","koelkast","watermeloen","televisie","gitaarkoffer","trampoline",

"sinaasappel","handschoenen","theepot","fotolijst","krokodil","schildersezel","pu
zzelstukje"]


"""Woord selecteren"""
# genereer index voor te zoeken woord
aantal = len(woorden)
kies = randint(0,aantal)
# lijst met te zoeken letters en lijst met juiste antwoorden
zoek = list(woorden[kies])
antwoord=["_"]*len(zoek)

"""Woord zoeken"""
levens = 6
while "_" in antwoord and levens>0:
    print(antwoord)
    letter = input("Geef een letter: ")
    if letter in zoek:
        #Juist geraden
        while letter in zoek:
            x=zoek.index(letter)
            antwoord[x]=letter
            zoek[x]="*"
```

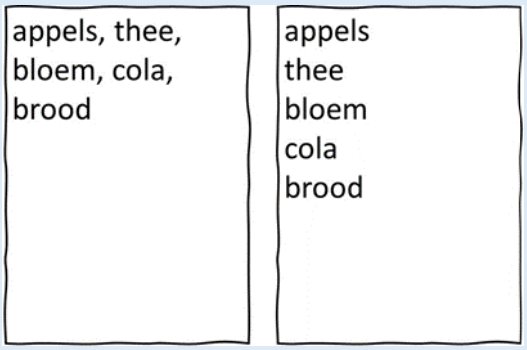
```
        print("Juist geraden.")
        print()
    else:
        """Fout geraden"""
        levens-=1
        print("Deze letter komt niet voor.")
        print("Je hebt nog",levens,". Probeer opnieuw.")
        print()
if levens==0:
    print("Sorry, je hebt verloren.")
else:
    print("Je hebt gewonnen.")
print("Het woord dat we zochten was",woorden[kies])
sleep(2)
```

Module 6: Externe bestanden

Opdracht 6.1

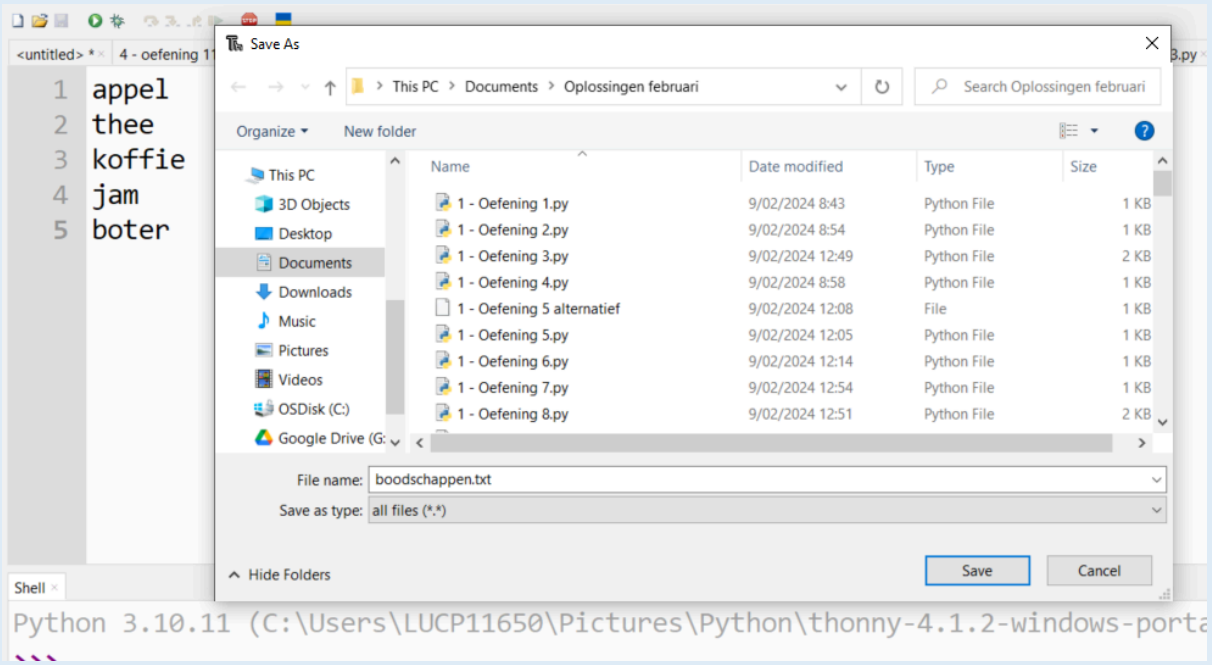
Neem een blad papier en maak een boodschappenlijstje met een vijftal producten.

- Welke informatie heb je nodig?
- Hoe heb je deze informatie voorgesteld?



Opdracht 6.2

Neem een eenvoudig tekstverwerkingsprogramma of maak gebruik van je editor om een tekstbestand aan te maken. Bewaar het tekstbestand als boodschappen.txt.

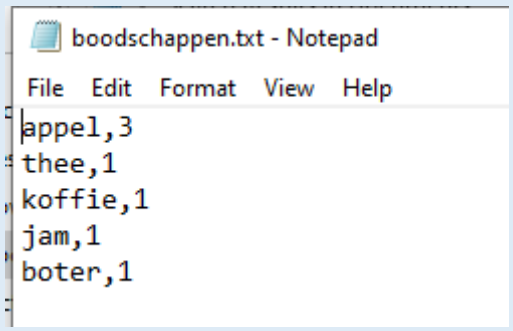


Opdracht 6.3

Pas je bestand boodschappen.txt en de code aan zodat niet enkel ingelezen wordt wat je moet kopen, maar ook hoeveel.

```
fp = open("boodschappen.txt","r")
boodschappen=[]
buffer = fp.readline()

teller=0
while buffer!="":
    #buffer=str(buffer)
    if '\n' in buffer:
        buffer=buffer[:-1]
    boodschappen=boodschappen+[buffer.split(',')]
    buffer = fp.readline()
    teller+=1
fp.close()
print(boodschappen)
```

Aangepast bestand:



Opdracht 6.4

Pas het woordraapspel (oefening 5.1) zo aan dat het script de geheime woorden inleest van een tekstbestand.

Oplossing

```
from random import randint
from time import sleep


"""Woorden inlezen"""
fp = open("woorden.txt","r")
woorden=[]
buffer = fp.readline()
while buffer!="":
    buffer=str(buffer)
    if '\n' in buffer:
        buffer=buffer[:-1]
    woorden=woorden+[buffer]
    buffer = fp.readline()
fp.close()
```

```
"""Woord selecteren"""
# genereer index voor te zoeken woord
aantal = len(woorden)-1
kies = randint(0,aantal)
# lijst met te zoeken letters en lijst met juiste antwoorden
zoek = list(woorden[kies])
antwoord=["_"]*len(zoek)
print(woorden[kies])

"""Woord zoeken"""
levens = 6
while "_" in antwoord and levens>0:
    print(antwoord)
    letter = input("Geef een letter: ")
    if letter in zoek:
        """Juist geraden"""
        while letter in zoek:
            x=zoek.index(letter)
            antwoord[x]=zoek[x]
            zoek[x]="*"
        print("Juist geraden.")
        print()
    else:
        """Fout geraden"""
        levens-=1
        print("Deze letter komt niet voor.")
        print("Je hebt nog",levens,". Probeer opnieuw.")
        print()
if levens==0:
    print("Sorry, je hebt verloren.")
else:
    print("Je hebt gewonnen.")
print("Het woord dat we zochten was",woorden[kies])
sleep(2)
```

Opdracht 6.5

Pas de code van opdracht 6.3 aan zodat je, nadat je het bestand gelezen hebt, je nieuwe boodschappen kunt ingeven.

Voordat je het script afsluit, schrijf je de nieuwe lijst weg naar het tekstbestand.

Oplossing:

```
fp = open("boodschappen.txt","r")
boodschappen=[]
buffer = fp.readline()

teller=0
while buffer!="":
    #buffer=str(buffer)
    if '\n' in buffer:
        buffer=buffer[:-1]
    boodschappen=boodschappen+[buffer.split(',')]
    buffer = fp.readline()
    teller+=1
fp.close()
print(boodschappen)
```

```
extra = input("Wat moet je nog kopen? ")
while extra!='':
    hoeveel = input("Aantal: ")
    boodschappen = boodschappen+[[extra,hoeveel]]
    extra = input("Wat moet je nog kopen? ")

fp = open( "boodschappen.txt", "w" )
index = 0
lengte = len(boodschappen)
while index<lengte:
    tekst = str(boodschappen[index][0])+','+str(boodschappen[index][1])+"\n"
    fp.write( tekst)
    index=index+1
fp.close()
```