

# Python: 13. Oplossingen

Dr. Cornelis Stal

April 27, 2022

## 1 Oplossingen van de oefeningen

### 1.1 01\_02\_Python\_Variabelen

#### 1.1.1 Opdracht: tjirpende krekel

Krekels tjirpen door hun vleugels langs elkaar te strijken. Bij de meeste soorten zijn het enkel de mannetjes die het zo bekende geluid voortbrengen om zo partners te kunnen aan te trekken. Wanneer je rustig aan het luisteren bent naar het rustgevende geluid van tjirpende krekel, zou je dus eigenlijk de bedenking moeten maken dat je aan het luistervinken bent naar een paringsritueel waarmee degene die de serenade brengt als enige doel heeft om wellustige mannetjeskrekel af te schrikken en geïnteresseerde vrouwtjes het hof te maken. De idee dat het tellen van de tjirpgeluiden die krekel voortbrengen ook kan dienen als een informele manier om de temperatuur te bepalen is echter niet nieuw. Het werd oorspronkelijk beschreven in 1897 door de natuurkundige Amos Dolbear, in een artikel met als titel “De krekel als thermometer”. Daarin stelde Dolbear aanvankelijk dat de buitentemperatuur een belangrijke bepalende factor is voor de frequentie waarmee krekel tjirpen.

Doorheen de jaren werd zijn manier om naar dit verband te kijken echter omgekeerd — mensen tellen nu het aantal tjirpgeluiden om daarmee de temperatuur te bepalen, eerder dan naar een thermometer te kijken om te voorspellen hoeveel tjirpgeluiden ze zullen horen. De wet van Dolbear geeft het verband aan onder de vorm van volgende formule, die aangeeft hoe de temperatuur ( $T_F$ ) in graden Fahrenheit kan geschat worden op basis van het aantal gehoorde tjirps per minuut ( $N_{60}$ ):

$$T_F = 50 + \frac{N_{60} - 40}{4}$$

Deze formule kan ook herschreven worden om de temperatuur in graden Celsius ( $^{\circ}\text{C}$ ) te bepalen:

$$T_C = 10 + \frac{N_{60} - 40}{7}$$

Bovenstaande formules worden uitgedrukt in termen van gehele getallen zodat ze makkelijker kunnen onthouden worden. Ze zijn bijgevolg niet direct geschikt om een exacte temperatuursbepaling te doen.

Schrijf een programma dat de temperatuur in graden celsius berekent op basis van het aantal tjirps per minuut ( $N_{60}$ ).

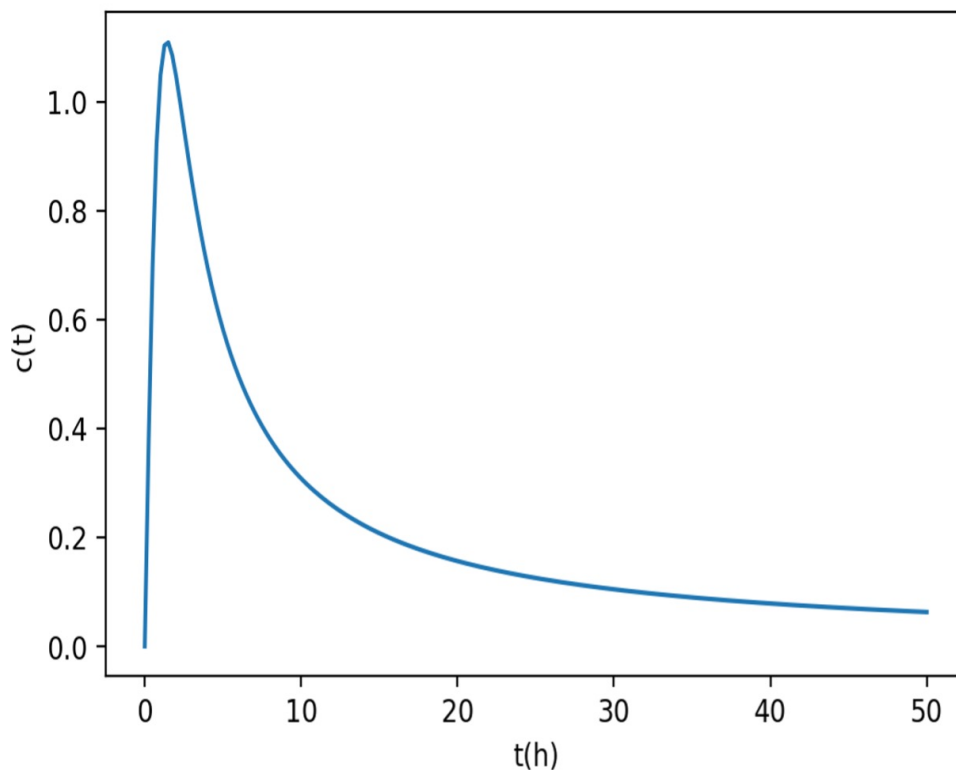
```
[1]: aantalTsjirpen = int(40) #input('Geef het aantal tsjirpen per minuut')) -->
    ↪Geen commentaar hier om input mogelijk te maken
    temperatuurInGradenCelsius = 10 + (aantalTsjirpen - 40) / 7
    print('De temperatuur is ' + str(temperatuurInGradenCelsius))
```

De temperatuur is 10.0

### 1.1.2 Opdracht: concentratie MDMA

Om het even welke drugs je gebruikt (alcohol, cannabis, cocaïne ...), je vindt steeds ruime tijd na inname sporen van de werkzame stof(fen) terug in het bloed. Zo kan je XTC tot 72 uur na de inname detecteren in het bloed. Hieronder zie je de concentratie MDMA (de werkzame stof in XTC) uitgedrukt in g/ml. Het voorschrift van de functie cc die de concentratie MDMA beschrijft in functie van de tijd  $t$ , uitgedrukt in uren, is:

$$c(t) = \frac{\pi * t}{(t^2 + 2)}$$



Vraag aan de gebruiker het aantal uren  $t \in R+$  sinds de opname van een dosis XTC. Bereken vervolgens de concentratie MDMA in het bloed.

```
[2]: import math
    t = int(24) #input('Geef de tijd t uitgedrukt in uren')) --> Geen commentaar
    ↪hier om input mogelijk te maken
    concentratie = (math.pi * t) / (t * t + 2)
```

```
print('De concentratie is ' + str(concentratie))
```

De concentratie is 0.13044675378227516

## 1.2 01\_04\_Python\_Condities

### 1.2.1 Opdracht: pakje sturen

Voor het versturen van een pakje rekent de post het volgende bedrag aan: Voor een pakje van twee kilogram of minder betaalt men 18 euro. Voor elke kilogram meer wordt er een bijkomend bedrag van 5 euro per kilogram gerekend. Dit geldt ook voor de laatste (eventueel onvolledige) kilogram.

Schrijf code die het te betalen bedrag berekent.

```
[3]: import math
gewicht = float(3.4) #input('Geef het gewicht van het pakje')) --> Geen
      ↪commentaar hier om input mogelijk te maken
gewicht = math.ceil(gewicht)
prijs = 18
if gewicht > 2:
    prijs = prijs + (gewicht - 2) * 5
print('De prijs is €' + str(prijs))
```

De prijs is €28

### 1.2.2 Opdracht: kinderbijslag berekenen

In een vereenvoudigde versie wordt de oude kinderbijslag als volgt berekend (zonder rekening te houden met de leeftijd): - kind 1: €90.28 - kind 2: €167.05 - kind 3 en volgende: €249.91

Bereken de kinderbijslag voor een opgegeven aantal kinderen. (Bron: [Gezinsbond, kinderbijslag](#))

```
[4]: import math
aantalKinderen = int(5) #input('Geef het aantal kinderen')) --> Geen commentaar
      ↪hier om input mogelijk te maken
kinderbijslag = 0
if aantalKinderen == 1:
    kinderbijslag = 90.28
elif aantalKinderen == 2:
    kinderbijslag = 90.28 + 167.05
else:
    kinderbijslag = 90.28 + 167.05 + (aantalKinderen - 2) * 249.91
print('Kinderbijslag ' + str(kinderbijslag))
```

Kinderbijslag 1007.0600000000001

### 1.2.3 Opdracht: internationale prijsvergelijkingen

De Big Mac index is een methode waarmee wisselkoersen op een licht verteerbare manier kunnen geanalyseerd worden. Deze index werd in 1986 door Pam Woodall ontwikkeld voor de Amerikaanse krant The Economist. Sindsdien publiceert die krant elk jaar een geactualiseerde versie van de index. Deze index gaf ook aanleiding tot de term *burgernomics*. Het principe werkt heel eenvoudig. De Big Mac wisselkoers tussen twee landen kan berekend worden door de prijs van een Big Mac in één land (in de lokale munteenheid van dat land) te delen door de prijs van een Big Mac in een ander land (opnieuw in de lokale munteenheid van dat land). Deze berekende wisselkoers wordt dan vergeleken met de reële wisselkoers op dat ogenblik. Indien deze lager uitvalt, dan is de eerste munt ondergewaardeerd ten opzichte van de tweede, anders is de eerste munt overgewaardeerd ten opzichte van de tweede. Dit wordt geïllustreerd aan de hand van volgend voorbeeld:

- in Europa betaalt men €4.74 voor een Big Mac
- in de Verenigde Staten betaalt men \$5.51 voor een Big Mac – de wisselkoers kan dus berekend worden als  $€4.74 / \$5.51$  per dollar = €0.860 – de reële wisselkoers bedraagt echter €0.89 per dollar – de euro is dus met -3.37% overgewaardeerd ten opzichte van de dollar  $(0.86 - 0.89) / 0.89 \times 100 = -3.37$

Schrijf een programma dat op basis van de Big Mac index een analyse maakt van de wisselkoers van een bepaald land ten opzichte van de dollar. Dit programma heeft twee argumenten die als floating point getallen moeten beschouwd worden:

- De prijs van een Big Mac
- De reële wisselkoers ten opzichte van de dollar

Bij de berekening van de Big Mac wisselkoers mag de functie uitgaan van een kostprijs van \$5.51 voor een Big Mac in de Verenigde Staten. Op basis van de procentuele verhouding van de berekende dollarkoers ten opzichte van de reële dollarkoers, moet de functie een string teruggeven overeenkomstig onderstaande tabel.

Verhouding (x)	Waardering
$v < -25$	sterk ondergewaardeerd
$-25 < v < -5$	ondergewaardeerd
$-5 < v < 5$	ongeveer gelijk
$5 < v < 25$	overgewaardeerd
$v > 25$	sterk overgewaardeerd

```
[5]: prijsBigMac = float(4.74) #input('Geef de prijs van een big mac')) --> Geen␣
    ↪commentaar hier om input mogelijk te maken
reeleWisselkoers = float(0.89) #input('Geef de winkelkoers')) --> Geen␣
    ↪commentaar hier om input mogelijk te maken
bigMacWisselkoers = prijsBigMac / 5.51
verhouding = (bigMacWisselkoers - reeleWisselkoers) / reeleWisselkoers * 100
if verhouding <= -25:
    print("Sterk ondergewaardeerd!")
elif verhouding <= -5:
```

```

    print("Ondergewaardeerd")
elif verhouding <= 5:
    print("Ongeveer gelijk")
elif verhouding <= 25:
    print("Overgewaardeerd")
else:
    print("Sterk overgewaardeerd")

```

Ongeveer gelijk

## 1.3 01\_05\_Python\_Iteraties

### 1.3.1 Opdracht: stemgerechtigden

Men schat dat het aantal stemgerechtigden in een bepaalde stad de volgende jaren als volgt zal verlopen:

$$N(t) = 30 + 12t^2 - t^3$$

met  $0 \leq t \leq 8$  waarbij  $t$  de tijd in jaren is en  $N(t)$  het aantal stemgerechtigden (in duizenden) na  $t$  jaar.

Bepaal voor elk jaar het aantal stemgerechtigden.

```

[6]: for teller in range(1, 9):
      aantalStemgerechtigden = 30 + 12 * teller * teller - teller * teller *   

      ↪ teller
      print("jaar = " + str(teller) + " aantal stemgerechtigden = " +   

      ↪ str(aantalStemgerechtigden))

```

```

jaar = 1 aantal stemgerechtigden = 41
jaar = 2 aantal stemgerechtigden = 70
jaar = 3 aantal stemgerechtigden = 111
jaar = 4 aantal stemgerechtigden = 158
jaar = 5 aantal stemgerechtigden = 205
jaar = 6 aantal stemgerechtigden = 246
jaar = 7 aantal stemgerechtigden = 275
jaar = 8 aantal stemgerechtigden = 286

```

### 1.3.2 Opdracht: vat

Een cilindervormig vat met hoogte van 32 dm heeft een inhoud van 8000 liter en is geheel gevuld met water. Als men de kraan opendraait stroomt het vat leeg. Tijdens het leegstromen geldt voor de hoogte  $h$  van de waterspiegel op tijdstip  $t$  bij benadering de formule:

$$h(t) = 0.0008 * t^2 - 0.32 * t + 32$$

Hierin is  $t$  de tijd in minuten vanaf het moment waarop de kraan opengedraaid wordt en  $h$  de hoogte van de waterspiegel in decimeter.

Schrijf een programma waarmee de volgende vragen beantwoord kunnen worden: - Wat is de hoogte na 2, 4, ... 10 minuten? - Na hoeveel minuten is de hoogte gezakt onder 5 dm?

```
[7]: # Vraag 1
for teller in range(2, 11, 2):
    hoogte = 0.0008 * teller * teller - 0.32 * teller + 32
    print("minuten = " + str(teller) + " hoogte = " + str(hoogte))
```

```
minuten = 2 hoogte = 31.3632
minuten = 4 hoogte = 30.7328
minuten = 6 hoogte = 30.1088
minuten = 8 hoogte = 29.4912
minuten = 10 hoogte = 28.88
```

```
[8]: # Vraag 2
teller = 0
hoogte = 0.0008 * teller * teller - 0.32 * teller + 32

while hoogte >= 5:
    teller = teller + 1
    hoogte = 0.0008 * teller * teller - 0.32 * teller + 32
    if (teller < 3): # Hoort niet bij de code, maar maakt het resultaat
        ↪overzichtelijk
        print("minuten = " + str(teller) + " hoogte = " + str(hoogte))
print("...") # Hoort niet bij de code, maar maakt het resultaat overzichtelijk
print("minuten = " + str(teller) + " hoogte = " + str(hoogte)) # Hoort niet bij
        ↪de code, maar maakt het resultaat overzichtelijk
```

```
minuten = 1 hoogte = 31.6808
minuten = 2 hoogte = 31.3632
...
minuten = 121 hoogte = 4.9928000000000003
```

### 1.3.3 Opdracht: vuurpijl

Van een hoge brug wordt op tijdstip  $t = 0$  een vuurpijl verticaal omhoog geschoten. De hoogte van de vuurpijl (uitgedrukt in meter) wordt beschreven door de functie met het voorschrift:

$$h(t) = -5t^2 + 30t + 80$$

waarbij  $t$  de tijd is (uitgedrukt in seconden).

Na hoeveel seconden landt de vuurpijl op de grond?

```
[9]: teller = 0
      hoogte = -5 * teller * teller + 30 * teller + 80
      while hoogte >= 0:
          teller = teller + 1
          hoogte = -5 * teller * teller + 30 * teller + 80
          print("minuten = " + str(teller) + " hoogte = " + str(hoogte))
```

```
minuten = 1 hoogte = 105
minuten = 2 hoogte = 120
minuten = 3 hoogte = 125
minuten = 4 hoogte = 120
minuten = 5 hoogte = 105
minuten = 6 hoogte = 80
minuten = 7 hoogte = 45
minuten = 8 hoogte = 0
minuten = 9 hoogte = -55
```

### 1.3.4 Opdracht: fietscomputers

Een bedrijf produceert draadloze fietscomputers. De bedrijfsleiding wil weten hoeveel computers er per uur moeten geproduceerd worden om winst te maken. Het verband tussen de winst  $W$  (in euro) en de productie  $x$  (aantal geproduceerde computers) wordt gegeven door:

$$W(x) = 0.5x^3 + 8x^2 - 24x$$

Schrijf een programma om dit vraagstuk op te lossen.

```
[10]: aantal = 0
      winst = 0.5 * aantal * aantal * aantal + 8 * aantal * aantal - 24 * aantal
      while winst <= 0:
          aantal = aantal + 1
          winst = 0.5 * aantal * aantal * aantal + 8 * aantal * aantal - 24 * aantal
          print("aantal = " + str(aantal) + " winst = " + str(winst))
```

```
aantal = 1 winst = -15.5
aantal = 2 winst = -12.0
aantal = 3 winst = 13.5
```

### 1.3.5 Opdracht: firma

Een firma verkoopt maandelijks 5000 stuks van een bepaald type. De vraagprijs per stuk bedraagt 30 euro. Marktonderzoek heeft uitgewezen dat de verkoop met 500 stuks zal stijgen telkens de prijs met 2 euro wordt verlaagd. Welke prijs moet de firma aanrekenen wil men een maximale omzet realiseren?

```
[11]: prijs = 30
      aantal = 5000
```

```

vorigeOmzet = 0
volgendeOmzet = prijs * aantal

while (vorigeOmzet < volgendeOmzet):
    prijs = prijs - 2
    aantal = aantal + 500
    vorigeOmzet = volgendeOmzet
    volgendeOmzet = prijs * aantal
    print("prijs = " + str(prijs) + " aantal = " + str(aantal) + " omzet = " +
↪str(volgendeOmzet))

```

```

prijs = 28 aantal = 5500 omzet = 154000
prijs = 26 aantal = 6000 omzet = 156000
prijs = 24 aantal = 6500 omzet = 156000

```

### 1.3.6 Opdracht: vermoeden van Collatz

We beschouwen de rij met onderstaand voorschrift:

- $a_{i+1} = \frac{a_i}{2}$  indien  $a_i$  is even
- $a_{i+1} = 3a_i + 1$  indien  $a_i$  is oneven

waarbij we de rij starten met  $a_0 = n$ . Het vermoeden van Collatz bestaat erin dat deze rij, ongeacht  $n > 0$  vroeg of laat op 1 uitkomt.

Schrijf een programma dat voor een ingelezen  $n > 0$  bepaalt na hoeveel stappen de rij op het getal 1 botst.

```

[12]: n = int(8) # input('Geef het aantal n')) --> Geen commentaar hier om input
↪mogelijk te maken
while (n > 1):
    if (n % 2) == 0:
        n = n / 2
    else:
        n = 3 * n + 1
    print(n)

```

```

4.0
2.0
1.0

```

## 1.4 01\_06\_Python\_Functies

### 1.4.1 Opdracht: wandelen en calorieenverbranden

Iemand die wandelt aan 5km/uur verbruikt het volgende aantal calorieën per kilometer (Bron: [calorieenverbranden])(<https://www.calorieenverbranden.nl/wandelen>):

- tot 55 kg: 44 kcal
- tot 65 kg: 51 kcal
- tot 75 kg: 59 kcal



- tot 85 kg: 67 kcal
- tot 95 kg: 75 kcal
- anders: 104 kcal

Een persoon met een gewicht van 85 kg zal dus bij een wandeling van 25 km een totaal van 1875 kcal verbranden.

Schrijf een functie die op basis van het gewicht en het aantal kilometer het aantal verbruikte kcal geeft.

```
[13]: def berekenAantalCalorieen(gewicht, aantalKilometer):
    aantalCalorieen = 0
    if gewicht < 55:
        aantalCalorieen = 44 * aantalKilometer
    elif gewicht < 65:
        aantalCalorieen = 51 * aantalKilometer
    elif gewicht < 75:
        aantalCalorieen = 59 * aantalKilometer
    elif gewicht < 85:
        aantalCalorieen = 67 * aantalKilometer
    elif gewicht < 95:
        aantalCalorieen = 75 * aantalKilometer
    else:
        aantalCalorieen = 104 * aantalKilometer
    return aantalCalorieen

gewicht = float(85) #input('Geef het gewicht')) --> Geen commentaar hier om
↳ input mogelijk te maken
aantalKilometer = int(25) #input('Geef het aantal gewandelde kilometers')) -->
↳ Geen commentaar hier om input mogelijk te maken
aantalCalorieen = berekenAantalCalorieen(gewicht, aantalKilometer)
print('aantalCalorieen ' + str(aantalCalorieen))
```

aantalCalorieen 1875

### 1.4.2 Opdracht: levensverwachting

Veronderstel dat de levensverwachting van een doorsnee persoon 70 jaar is, en dat die levensverwachting kan worden aangepast op basis van de volgende criteria:

- Bij vrouwen wordt er vier jaar bijgeteld
- Bij rokers wordt er vijf jaar afgetrokken en bij niet-rokers wordt er vijf jaar bijgeteld
- Trek drie jaar af voor personen die nooit sporten en tel één jaar bij voor elk uur dat iemand wekelijks aan sport doet
- Trek een half jaar af voor elk geconsumeerd glas alcohol boven de zeven dat iemand per week drinkt (bij iemand die elf glazen alcohol per week drinkt worden dus twee jaren afgetrokken); bij geheelonthouders wordt er twee jaar bijgeteld
- Bij personen die niet vaak fast food eten wordt er drie jaar bijgeteld

Op basis van deze formule kan de levensverwachting van een mannelijke roker die slechts

twee uur per week aan sport doet, tien glazen alcohol per week drinkt en vaak fast food eet worden berekend op 65,5 jaar. Commentaar: Dit is GEEN wetenschappelijke voorspelling.

Schrijf een functie levensverwachting waaraan de volgende vijf parameters moeten doorgegeven worden: - **geslacht**: een string die het geslacht van de persoon aangeeft (man of vrouw) - **roker**: een Booleaanse waarde die aangeeft of de persoon rookt - **sport**: een natuurlijk getal dat aangeeft hoeveel uren per week de persoon aan sport doet - **alcohol**: een natuurlijk getal dat aangeeft hoeveel glazen alcohol de persoon per week drinkt - **fastfood**: een Booleaanse waarde die aangeeft of de persoon vaak fastfood eet

De functie moet op basis van de doorgegeven waarden de levensverwachting voorspellen, en moet deze prognose als resultaat teruggeven. Het resultaat van de functie moet als een reëel getal teruggegeven worden. Bijvoorbeeld: - `print(levensverwachting("man", True, 2, 10, True))` geeft 65.5 - `print(levensverwachting("man", True, 5, 5, True))` geeft 70.0 - `print(levensverwachting("vrouw", False, 5, 0, False))` geeft 89.0 - `print(levensverwachting("vrouw", False, 3, 14, True))` geeft 78.5 - `print(levensverwachting("man", False, 4, 4, False))` geeft 82.0

```
[14]: def levensverwachting(geslacht,roker,sport,alcohol,fastfood):
    resultaat = 70.0;
    if geslacht == "vrouw":
        resultaat += 4
    if roker == True:
        resultaat -= 5
    else:
        resultaat += 5
    if sport == 0:
        resultaat -= 3
    else:
        resultaat += sport
    if alcohol == 0:
        resultaat += 2
    elif alcohol > 7:
        resultaat -= (alcohol - 7) / 2.0
    if fastfood == False:
        resultaat += 3
    return resultaat

print(levensverwachting("man", True, 2, 10, True)) #65.5
print(levensverwachting("man", True, 5, 5, True)) # 70.0
print(levensverwachting("vrouw", False, 5, 0, False)) #89.0
print(levensverwachting("vrouw", False, 3, 14, True)) # 78.5
print(levensverwachting("man", False, 4, 4, False)) # 82.0
```

65.5

70.0

89.0  
78.5  
82.0

## 1.5 01\_07\_Python\_Containers

### 1.5.1 Opdracht: statistische berekeningen op scores

Schrijf een programma dat een aantal berekeningen uitvoert op basis van de punten (op 20) van 10 leerlingen: `resultaten = (13.0, 17.0, 5.0, 4.5, 9.0, 12.0, 8.0, 9.0, 14.0, 7.5)`

De volgende zaken moeten worden beantwoord voor deze scores: - Wat is het gemiddelde? - Hoeveel leerlingen zijn er geslaagd? - Hoeveel leerlingen zijn er niet geslaagd? - Hoeveel leerlingen hebben een score in de volgende intervallen: -  $[0 - 6[$  -  $[6 - 10[$  -  $[10 - 13[$  -  $[13 - 20]$

Schrijf de resultaten uit.

```
[15]: resultaten = (13.0, 17.0, 5.0, 4.5, 9.0, 12.0, 8.0, 9.0, 14.0, 7.5)

# gemiddelde
som = 0
aantal = len(resultaten)
aantalGeslaagden = 0
aantal0Tot6 = 0
aantal6Tot10 = 0
aantal10Tot13 = 0
aantal13Tot20 = 0
for resultaat in resultaten:
    som += resultaat
    if resultaat >= 10:
        aantalGeslaagden += 1
    if resultaat < 6:
        aantal0Tot6 += 1
    elif resultaat < 10:
        aantal6Tot10 += 1
    elif resultaat < 13:
        aantal10Tot13 += 1
    else:
        aantal13Tot20 += 1
gemiddelde = som / aantal
print("Het gemiddelde is " + str(gemiddelde))
print("Het aantal geslaagden is " + str(aantalGeslaagden))
print("Het aantal niet geslaagden is " + str(aantal - aantalGeslaagden))
print("Het aantal leerlingen met punten tussen 0 en 6 " + str(aantal0Tot6))
print("Het aantal leerlingen met punten tussen 6 en 10 " + str(aantal6Tot10))
print("Het aantal leerlingen met punten tussen 10 en 13 " + str(aantal10Tot13))
print("Het aantal leerlingen met punten tussen 13 en 20 " + str(aantal13Tot20))
```

Het gemiddelde is 9.9  
 Het aantal geslaagden is 4  
 Het aantal niet geslaagden is 6  
 Het aantal leerlingen met punten tussen 0 en 6 2  
 Het aantal leerlingen met punten tussen 6 en 10 4  
 Het aantal leerlingen met punten tussen 10 en 13 1  
 Het aantal leerlingen met punten tussen 13 en 20 3

### 1.5.2 Opdracht: schoonspringen

In een wedstrijd schoonspringen wordt na elke sprong een score (tussen 0 en 100) toegekend door verschillende juryleden. De finale score van de deelnemer wordt dan bepaald door de hoogste en de laagste score te negeren en het gemiddelde te nemen van de andere scores. Dit gemiddelde wordt tenslotte afgerond naar het dichtstbijzijnde natuurlijke getal.

Bereken de eindscore voor een schoonspringer met de volgend deelscores:

scores = [59, 70, 75, 46, 89, 55, 91, 74, 56, 55, 62, 74, 83, 48]

```
[16]: scores = (59, 70, 75, 46, 89, 55, 91, 74, 56, 55, 62, 74, 83, 48)
minScore = 0
maxScore = 0
som = 0
aantal = len(scores)
for score in scores:
    if score < minScore:
        minScore = score
    elif score > maxScore:
        maxScore = score
    som += score

som = som - minScore - maxScore
aantal = aantal - 2
gemiddelde = round(som / aantal, 0)
print("Het gemiddelde is " + str(gemiddelde))
```

Het gemiddelde is 70.0

### 1.5.3 Opdracht: Manhattan-afstand

Voor twee vectoren  $x$  en  $y$  in  $N$  dimensies, met respectieve coördinaten  $(x_0, x_1, \dots, x_{N-1})$  en  $(y_0, y_1, \dots, y_{N-1})$  definieert men de Manhattan-afstand  $d_M(x, y)$  (geïnspireerd op het dambord-achtige stratenpatroon van Manhattan) als:

$$d_M(x, y) = \sum_{i=0}^{N-1} |x_i - y_i|$$

Schrijf een functie `manhattan(a, b)` die de hierboven gedefinieerde Manhattan-afstand

als resultaat oplevert. De argumenten zijn lists die reële getallen bevatten. Indien de dimensies van de argumentrijen *a* en *b* niet gelijk zijn, wordt *-1.0* als resultaat gegeven. Bijvoorbeeld:

- `print(manhattan([1.0, 2.0], [-1.0, -2.0]))` geeft een waarde 6.0
- `print(manhattan([1.0, 2.0, 3.0], [-1.0, -2.0]))` geeft een waarde -1

```
[17]: import math
def manhattan(x, y):
    if len(x) != len(y):
        return -1
    else:
        som = 0
        for i in range(0, len(x)):
            som += math.fabs(x[i] - y[i])
        return som

print(manhattan([1.0, 2.0], [-1.0, -2.0]))
```

6.0

#### 1.5.4 Opdracht: afwijkende lijsten

Gegeven zijn twee lijsten *a* en *b* die gehele getallen bevatten. Beide lijsten bevatten evenveel getallen. Het is de bedoeling om locaties *i* te vinden in deze lijsten, waarvoor geldt dat  $|a_i - b_i|$  strikt groter is dan *d*.

Programmeer de functie `afwijkende_lijsten()` met als argumenten: - de lijst *a* - de lijst *b* - de waarde *d*>0 en waarbij *d* geheel is

Het resultaat van de functie is een lijst van locaties *i* waarvoor de bovenstaande voorwaarde geldt. De resultaatlijst is van klein naar groot geordend. Voorbeeld:

`print(afwijkende_lijsten([1, 2, 4, 1, 4, 2],[1, 6, 4, 1, 5, -2], 3))` resulteert in `[1, 5]`

```
[18]: def afwijkende_lijsten(a, b, d):
    resultaat = []
    for i in range(0, len(a)):
        if abs(a[i] - b[i]) > d:
            resultaat.append(i)
    return resultaat

print(afwijkende_lijsten([1, 2, 4, 1, 4, 2],[1, 6, 4, 1, 5, -2], 3))
```

[1, 5]

#### 1.5.5 Opdracht: ozonconcentraties

Een weerkundig instituut krijgt dagelijks een bestand met de metingen van het ozongehalte in de lucht op een aantal vastgelegde meetplaatsen. Het ozongehalte wordt hierbij

uitgedrukt als een reëel getal en de meetplaatsen als een String. In deze oefening krijg je van enkele dagen de metingen van al de vastgelegde meetplaatsen. Er wordt gevraagd om te bepalen in welke meetplaatsen het ozongehalte op meer dan  $k$  dagen een bepaalde gegeven kritische waarde overschreed. Ontwerp en implementeer een algoritme voor dit probleem en geef aan hoe daarbij de standaard datatypes kunnen worden gebruikt.

Schrijf een Python-functie `risicoGebieden(metingen: list, k: int, kritischeWaarde: float)`, die een lijst van metingen als argument heeft, alsook een natuurlijk getal  $k$  en een reëel getal dat de kritische waarde voorstelt. De elementen van de lijst metingen zijn van het type `tuple(str, float)` waarbij de string de meetlocatie voorstelt en de float de meting zelf. De uitvoer van de functie `risicoGebieden` is een set van meetplaatsen waarvan het ozongehalte de kritische waarde meer dan  $k$  maal overschreden heeft. Voorbeelden:

```
print(risicoGebieden(("Gent",90.1),("Antwerpen",120.9),("Brussel",181.1),("Brugge",70.7),("Gent",150.50),("Antwerpen",190.3), ("Brussel",179.4),("Brugge",120.2),("Gent",190.2),("Brussel",200.1),("Brugge",110.1), ("Gent",160.4),("Antwerpen",162.1),("Brussel",190.9),("Brugge",120.1),("Gent",180.7),("Antwerpen",125.3),("Brussel",190.1),("Brugge",177.5)], 2, 180)) geeft ['Brussel']
```

```
print(risicoGebieden(("Gent",90.1),("Antwerpen",120.9),("Brussel",181.1),("Brugge",70.7),("Gent",150.50),("Antwerpen",190.3),("Brussel",179.4),("Brugge",120.2),("Gent",190.2),("Antwerpen",185.1),("Brussel",200.1),("Brugge",110.1),("Gent",160.4),("Antwerpen",182.1),("Brussel",190.9),("Brugge",120.1),("Gent",180.7),("Antwerpen",125.3),("Brussel",190.1),("Brugge",177.5),("Gent",155.2),("Antwerpen",199.2),("Brussel",200.1),("Brugge",160.1),("Gent",145.2),("Antwerpen",179.2),("Brussel",170.1),("Brugge",170.1)], 1, 180 )) geeft ['Brussel', 'Antwerpen', 'Gent']
```

```
print(risicoGebieden(("Gent",90.1),("Antwerpen",120.9),("Brussel",181.1),("Brugge",70.7),("Gent",150.50),("Antwerpen",190.3),("Brussel",179.4),("Brugge",120.2),("Gent",190.2),("Antwerpen",185.1),("Brussel",200.1),("Brugge",110.1),("Gent",160.4),("Antwerpen",182.1),("Brussel",190.9),("Brugge",120.1),("Gent",180.7),("Antwerpen",125.3),("Brussel",190.1),("Brugge",177.5),("Gent",155.2),("Antwerpen",199.2),("Brussel",200.1),("Brugge",160.1),("Gent",145.2),("Antwerpen",179.2),("Brussel",170.1),("Brugge",170.1)], 3, 180)) geeft ['Brussel', 'Antwerpen']
```

```
[19]: def risicoGebieden(metingen, k, kritischeWaarde):
```

```
    kritisch = {}
```

```
    risico = []
```

```
    for meting in metingen:
```

```
        if meting[1] > kritischeWaarde:
```

```
            if meting[0] in kritisch:
```

```
                kritisch[meting[0]] += 1
```

```
            else:
```

```
                kritisch[meting[0]] = 1
```

```
            if kritisch[meting[0]] == k + 1:
```

```
                risico.append(meting[0])
```

```
    print(risico)
```

```
metingen = (("Gent",90.1),("Antwerpen",120.9),("Brussel",181.1),("Brugge",70.7),
```

```

        ("Gent", 150.50), ("Antwerpen", 190.3), ("Brussel", 179.4), ("Brugge", 120.
↪2),
        ("Gent", 190.2), ("Antwerpen", 185.1), ("Brussel", 200.1), ("Brugge", 110.
↪1),
        ("Gent", 160.4), ("Antwerpen", 182.1), ("Brussel", 190.9), ("Brugge", 120.
↪1),
        ("Gent", 180.7), ("Antwerpen", 125.3), ("Brussel", 190.1), ("Brugge", 177.
↪5),
        ("Gent", 155.2), ("Antwerpen", 199.2), ("Brussel", 200.1), ("Brugge", 160.
↪1),
        ("Gent", 145.2), ("Antwerpen", 179.2), ("Brussel", 170.1), ("Brugge", 150.
↪1),
        ("Gent", 185.2), ("Antwerpen", 190.2), ("Brussel", 185.3), ("Brugge", 160.
↪7)]
k = 3
kritischeWaarde = 180
risicoGebieden(metingen, k, kritischeWaarde)

```

```
['Brussel', 'Antwerpen']
```

## 1.6 01\_08\_Python\_Werken met tekst

### 1.6.1 Opdracht: morsecode

In morsecode wordt elk karakter van een woord vertaald naar een unieke opeenvolging van punten (.) en streepjes (-), en worden de vertalingen van de individuele karakters telkens van elkaar gescheiden door één enkele spatie. Voor letters en cijfers worden de volgende punten en streepjes gebruikt:

```
{'A': '.-.', 'B': '-...', 'C': '-.-.', 'D': '-...', 'E': '...', 'F': '..-.',
'G': '--.', 'H': '....', 'I': '...', 'J': '.---', 'K': '-.-.', 'L':
'.-..', 'M': '--', 'N': '-.', 'O': '---', 'P': '.--.', 'Q': '--.-',
'R': '-.-.', 'S': '...', 'T': '-', 'U': '..-', 'V': '...-', 'W': '---',
'X': '-.-.-', 'Y': '-.-.-', 'Z': '---.', '0': '-----', '1': '.-----', '2':
'..---', '3': '...--', '4': '....-', '5': '.....', '6': '-....', '7':
'--...', '8': '---..', '9': '----.'}
```

Schrijf een programma waarmee een gebruiker een willekeurige tekst op kan geven, die vervolgens omgezet wordt in morsecode. Tekens worden geschreven door een spatie.

*Optioneel kan gebruik gemaakt worden van de `winsound`-bibliotheek of het `/a`-teken om de punten en strepen om te zetten in geluiden. Dit werkt echter niet altijd!*

```
[ ]:
```

```

morseCode = {'A': '.-', 'B': '-...', 'C': '-.-.', 'D': '-..', 'E': '.', 'F': '..-
↵-.', 'G': '--.', 'H': '....', 'I': '..', 'J': '.---', 'K': '-.-', 'L': '..-..
↵', 'M': '--', 'N': '-.', 'O': '---', 'P': '.---', 'Q': '--.-', 'R': '.-.',
↵'S': '...', 'T': '-', 'U': '..-', 'V': '...-', 'W': '---', 'X': '-.-.-', 'Y':
↵'-.-', 'Z': '--..', '0': '-----', '1': '.-----', '2': '..----', '3': '...--',
↵'4': '....-', '5': '.....', '6': '-.....', '7': '--....', '8': '---..', '9':
↵'----.'}

thisString = input("Give a string...")
morse = ''
for char in thisString.upper():
    morse += morseCode.get(char, ' ') + ' '
print(morse)

```

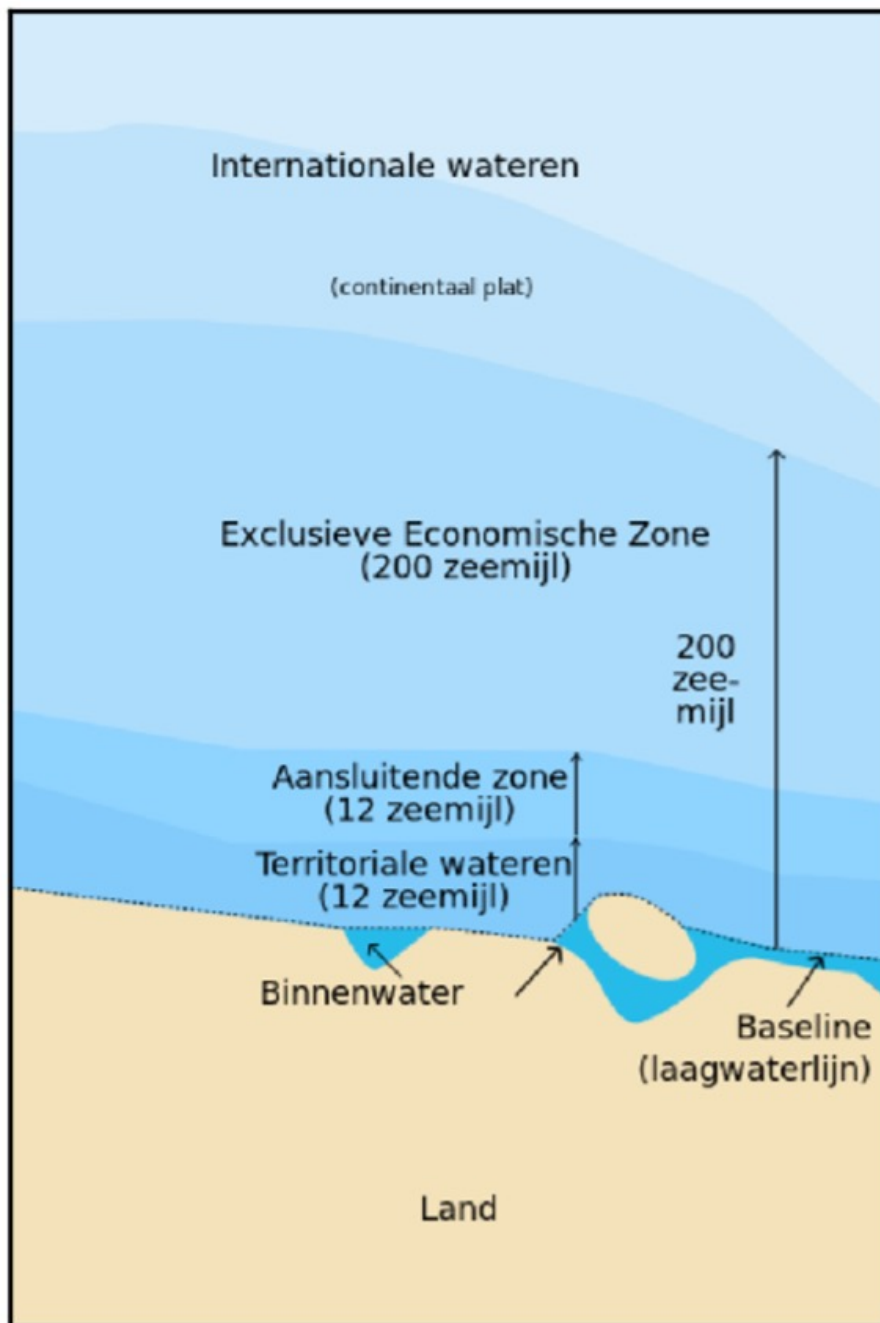
## 1.7 01\_09\_Python\_Werken met bibliotheken

### 1.7.1 Opdracht: maritieme indeling

#### Introductie.

De territoriale wateren worden gevormd door de zeestreek die grenst aan het landgebied van een kuststaat en waarover de soevereiniteit van deze staat zich uitstrekt (met inbegrip van het luchtruim, de bodem en de ondergrond). De territoriale wateren worden gescheiden van het landgebied en de binnenwateren door de basislijn (of laagwaterlijn). De maximale territoriale zone bedraagt 12 zeemijlen (1 zeemijl = 1852 meter). Binnen de territoriale zone kan een land zelf zijn wetten bepalen en rechtspraak toepassen. Zo kunnen schepen uit een ander land bijvoorbeeld niet gaan vissen in de territoriale wateren zonder de toestemming van de kuststaat. Ook zeezenders zijn strafbaar als ze zonder toestemming uitzenden vanuit de territoriale wateren, en kunnen dus enkel uitzenden vanuit de internationale wateren.





De aansluitende zone is de strook volle zee die grenst aan de territoriale wateren. Ze vormt een soort buffer tussen de territoriale wateren en de volle zee, waarin welomschreven en beperkte bevoegdheden gelden voor de kuststaat. De aansluitende zone strekt zich uit tot maximaal 24 zeemijl van de basislijn. De exclusieve economische zone is de strook volle zee die zich uitstrekt tot 200 zeemijl buiten de kust van een kuststaat. Het scheepvaartverkeer is daar vrij, als bevond het zich in internationale wateren, maar de kuststaat heeft het recht op de bodemrijkdom, het recht op visserij en

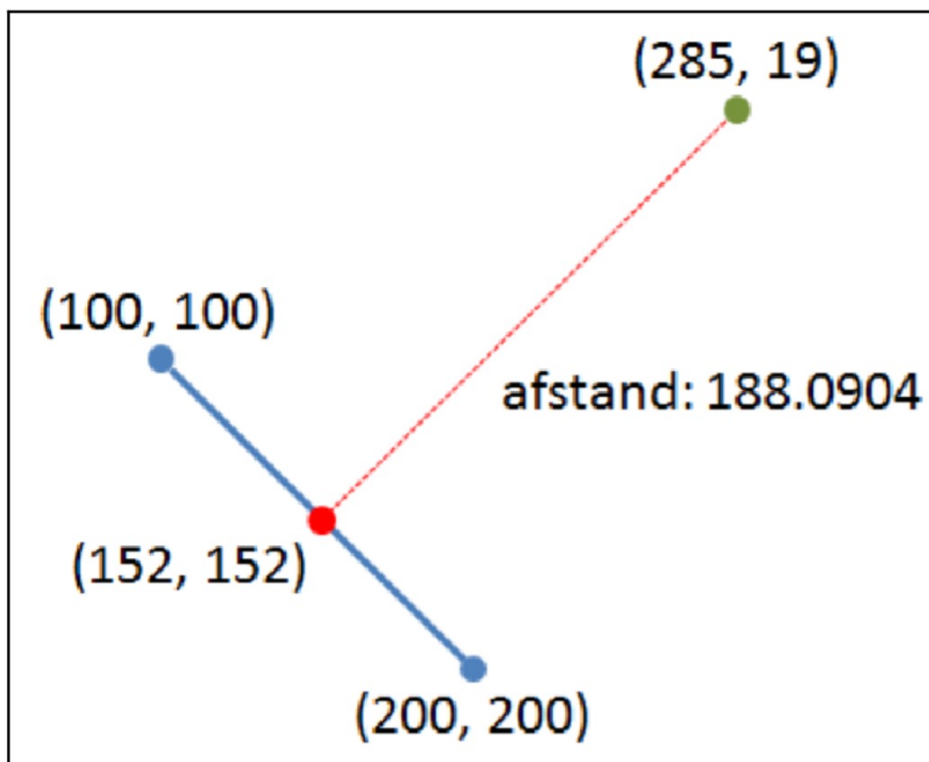
het recht op wetenschappelijk onderzoek. De internationale wateren worden tenslotte gevormd door de volle zee buiten de grenzen te land en ter zee van een staat.

afstand tot basislijn (zeemijl)	naam maritieme zone
$[0,12[$	territoriale wateren
$[12,24[$	aansluitende zone
$[24,200[$	exclusieve economische zone
$[200,\infty[$	internationale wateren

Uw opdracht bestaat erin om voor een gegeven punt in zee te bepalen in welke zone het zich bevindt, ten opzichte van een gegeven basislijn. Hierbij worden de zones gedefinieerd op basis van de volgende afstanden ten opzichte van de basislijn.

#### Afstand van een punt tot een rechte

De kortste afstand van een punt tot een rechte kan op de volgende manier berekend worden.



Stel dat de rechte door de twee punten  $(x_1, y_1)$  en  $(x_2, y_2)$  loopt en dat we de kortste afstand tussen deze rechte en het punt  $(x_3, y_3)$  zoeken. De loodlijn uit het punt  $(x_3, y_3)$  snijdt de rechte in het punt  $(x_v, y_v)$ , dat het voetpunt genoemd wordt. De kortste afstand is dan gelijk aan de afstand tussen de punten  $(x_3, y_3)$  en  $(x_v, y_v)$ . De coördinaten van het voetpunt  $(x_v, y_v)$  kunnen als volgt bepaald worden:

$$u = \frac{(x_3 - x_1)(x_2 - x_1) + (y_3 - y_1)(y_2 - y_1)}{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

$$x_v = x_1 + u(x_2 - x_1)$$

$$y_v = y_1 + u(y_2 - y_1)$$

De Euclidische afstand  $a$  tussen het voetpunt  $(x_v, y_v)$  en het punt  $(x_3, y_3)$  kan dan als volgt berekend worden:

$$a = \sqrt{(x_v - x_3)^2 + (y_v - y_3)^2}$$

**Invoer:** De invoer bestaat uit zes getallen  $x_1, y_1, x_2, y_2, x_3, y_3 \in R$ . De rechte door de punten  $(x_1, y_1)$  en  $(x_2, y_2)$  stelt de basislijn van een kuststaat voor. Het punt  $(x_3, y_3)$  stelt een punt in zee voor. Alle coördinaten worden uitgedrukt in zeemijl.

**Uitvoer:** Er moeten drie regels uitvoer gegenereerd worden, die respectievelijk de coördinaten van het voetpunt aangeven, de afstand van het gegeven punt in zee tot de gegeven basislijn (in zeemijl), en de zone waarin het punt zich bevindt (relatief ten opzichte van de basislijn). De naam van de maritieme zone moet bepaald worden op basis van de tabel uit de inleiding. Voorbeeld:

- water(100.0,100.0,200.0,200.0,285.0,19.0)
- voetpunt: 152.0, 152.0
- afstand: 188.090403796
- zone: Exclusieve economische zone

```
[23]: import math
def water(x1, y1, x2, y2, x3, y3):
    u = ((x3 - x1) * (x2 - x1) + (y3 - y1) * (y2 - y1)) / ((x2 - x1) * (x2 -
↪x1) + (y2 - y1) * (y2 - y1))
    xv = x1 + u * (x2 - x1)
    yv = y1 + u * (y2 - y1)
    a = math.sqrt((xv - x3) * (xv - x3) + (yv - y3) * (yv - y3))
    maritiemeZone = "Internationale wateren"
    if a < 12:
        maritiemeZone = "Territoriale wateren"
    elif a < 24:
        maritiemeZone = "Aansluitende zone"
    elif a < 200:
        maritiemeZone = "Exclusieve economische zone"
    print("voetpunt: " + str(xv) + ", " + str(yv))
    print("afstand: " + str(a))
    print("zone: " + maritiemeZone)

water(100.0,100.0,200.0,200.0,285.0,19.0)
```

voetpunt: 152.0, 152.0  
afstand: 188.09040379562165  
zone: Exclusieve economische zone