

Werken met bestanden

Tekstbestanden lezen en schrijven

1. Een veelgebruikte manier om de inhoud van een tekstbestand te lezen is met behulp van een guarded block :

```
with open ("tekst.txt") as f :  
  
    for line in f :  
        print(line)
```

Python zorgt hier zelf voor het sluiten van het bestand.

Lees het bestand pi.txt in en ga na of de laatste vier cijfers (opeenvolgend) van je gsm-nummer voorkomen in de benadering van pi in dit tekstbestand.

2. Schrijven naar een bestand kan als volgt :

```
filename = "tekst2.txt"  
f = open ( filename , "w" )  
  
lijn = "Schrijven naar een tekstbestand"  
  
f.write(lijn)  
f.close()
```

Tik een tekst in, lijn per lijn, eindig met Enter.
Schrijf elke lijn weg naar het tekstbestand.

Laat de inhoud van dit tekstbestand op het scherm verschijnen.

Binaire bestanden lezen en schrijven

Met de module pickle kan je schrijven naar een binair bestand en later weer inladen.

Deze module serialiseert Python-objecten: het zet ze om naar een binair formaat.

Pickle kan ook de omgekeerde bewerking van het deserialiseren: het converteren van het binaire formaat naar het Python-object.

Wees voorzichtig bij het binaire laden van bestanden van onbekende bronnen, pickle kan code bevatten die mogelijk kwaadaardig kan zijn.

Onderstaand voorbeeld toont hoe je data binair kan schrijven naar een bestand :

```
import pickle

# object om op te slaan

data = {"naam" : "Jan" , "leeftijd" : 30 , "beroep" : "Programmeur"}

# Schrijven naar een binair bestand

with open ( "data.pkl" , "wb" ) as bestand :
    pickle.dump( data , bestand )

print("Object opgeslagen in data.pkl ")
```

Hier zorgt "wb" voor het schrijven (write) in binaireodus.

De code pickle.dump(obj , file) slaat het object obj op in het bestand.

Het bestand weer inladen kan zo :

```
# Inladen vanuit het binaire bestand

with open ( "data.pkl" , "rb" ) as bestand :
    geladen_data = pickle.load ( bestand )

print ( " Ingeladen object : " , geladen_data )
```

"rb" staat hier voor lezen (read) in binaireodus.

Laden van het object gebeurt met pickle.load (bestand).

3. Lees het tekstbestand `pi.txt` volledig in, sla de inhoud op in de variabele `volledige_inhoud`.

Schrijf deze inhoud naar het bestand `onleesbaar.pkl`.

Gebruik `pickle` vervolgens om `onleesbaar.pkl` te openen en de inhoud ervan (leesbaar) te tonen op de console.

Beeldverwerking

4. Genereer een zwarte afbeelding van 900 pixels breed en 600 pixels hoog.

Vraag aan de gebruiker waar hij de oorsprong van een assenstelsel wenst. Gebruik hiervoor de variabelen : `oor_ver` en `oor_hor`.

Voorbeeld van invoer :

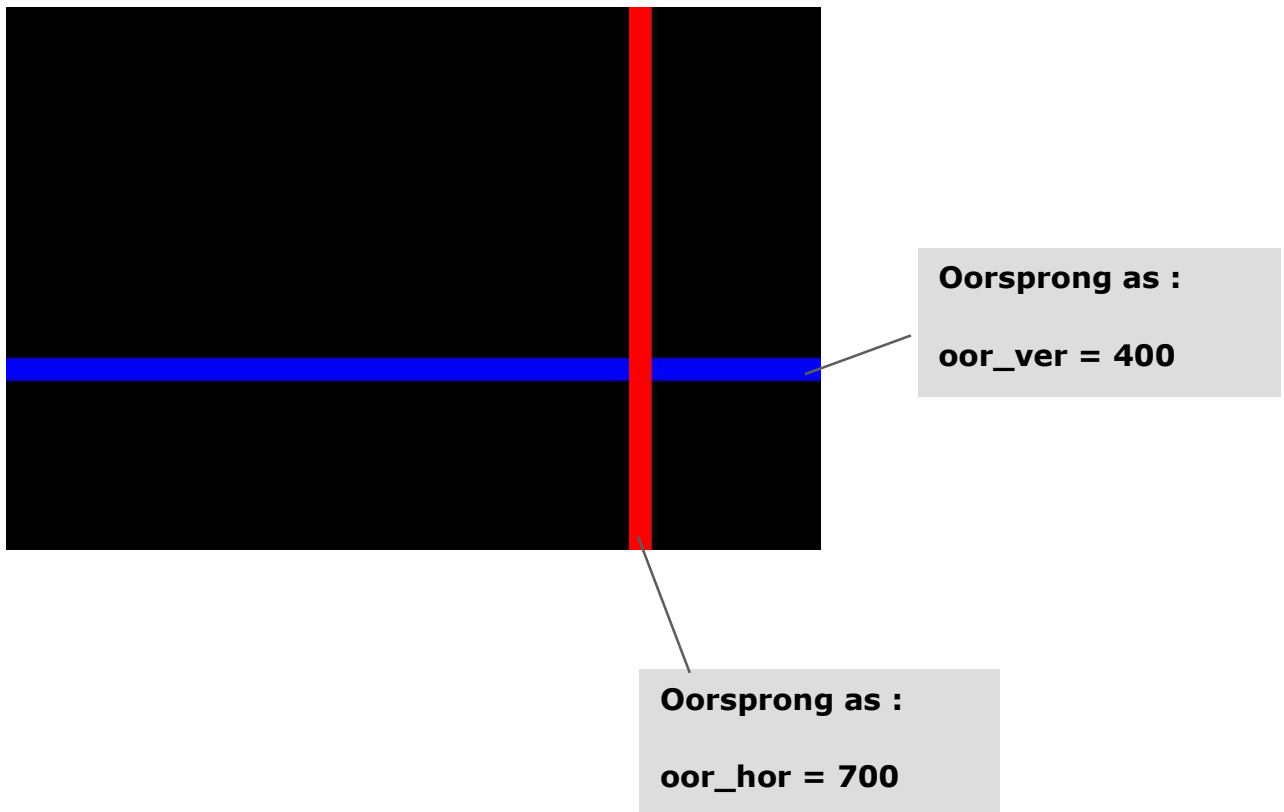
`oor_ver = 400`

`oor_hor = 700`

Teken een rode verticale as en een blauwe horizontale as door deze oorsprong.

Zorg ervoor dat de dikte van beide assen 25 pixels bedraagt.

De oorsprong van het assenstelsel ligt gecentreerd tussen de verticale en horizontale as.



```
from PIL import Image
import numpy as np

breedte = 900
hoogte = 600

r_out = np.zeros (( hoogte , breedte))
g_out = np.zeros (( hoogte , breedte))
b_out = np.zeros (( hoogte , breedte))

# jouw code


# eindig met het converteren van de drie matrices naar een
afbeelding

r_image_out = Image.fromarray(np.uint8(r_out))
g_image_out = Image.fromarray(np.uint8(g_out))
b_image_out = Image.fromarray(np.uint8(b_out))

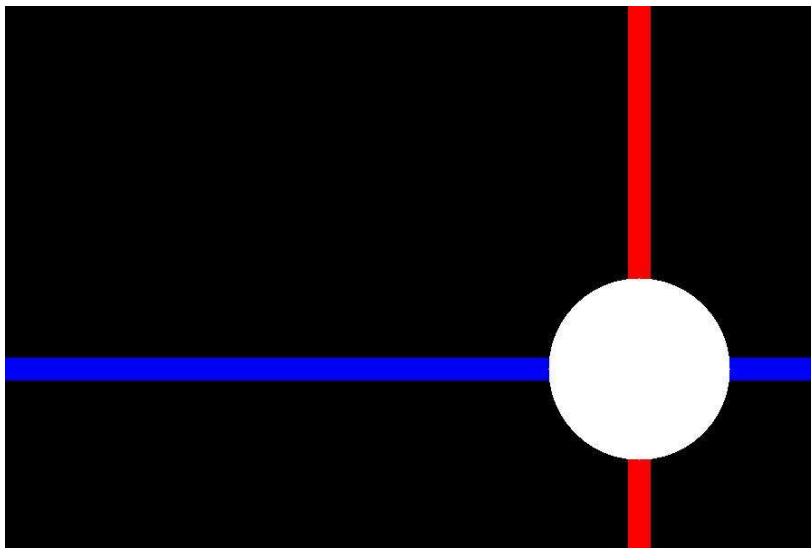
output_im = Image.merge("RGB", (r_image_out , g_image_out ,
b_image_out) )

output_im.show()
```

5. Vervolg Opgave 4.

Vraag nu aan de gebruiker de straal van een cirkel op.

Teken vervolgens een (volledig opgevulde) witte cirkel door de oorsprong.



6. Genereer een zwarte afbeelding met witte diagonalen.

Neem als breedte 900 pixels en als hoogte 600 pixels.

Zorg dat de diagonalen 10 pixels breed zijn.

Voor het bepalen van de diagonalen kan je gebruik maken van de vergelijking van een rechte r door de oorsprong:

$$r \leftrightarrow y = m x$$

Hierin is $m = \Delta y / \Delta x$ de richtingscoëfficiënt.

Voor de rico m kan je de verhouding breedte/hoogte gebruiken.

