OEFENINGEN PYTHON – REEKS 4

Vraag 1: Recursieve functies

Een recursieve functie is per definitie een functie die zichzelf oproept. Belangrijk is dat je bij het programmeren van zulke functies er goed op let dat je een stopvoorwaarde inbouwt waarbij de recursieve aanroepen stoppen. Indien je dit niet zou doen dan zou de functie oneindig veel keer zichzelf blijven oproepen.

a) Schrijf een recursieve functie die de faculteit van N berekent. N geef je als argument aan de functie mee. Gebruik de volgende formule:

$$n! = n(n-1)!$$

b) De grootste gemene deler van twee getallen a en b is het grootste getal dat een deler is van zowel a als b. Onderstaande formule geeft je de Euclidische manier om een ggd te berekenen. Schrijf een recursieve functie *ggd* die de grootste gemene deler van twee getallen berekent op de Euclidische manier.

$$ggd(a,b) = \begin{cases} ggd(b, a-b) & als\ a > b \\ a & als\ a = b \\ ggd(a, b-a) & als\ a < b \end{cases}$$

Vraag 2: Werken met bestaande code

Je werkt samen aan een project en je collegas hebben de volgende code fragmenten geschreven die hier en daar een fout bevatten. Aan jou om de fouten te vinden en te corrigeren.

a) Deze functies horen een temperatuur om te zetten van Farenheit naar Celsius en te controleren of het vriest.

```
def farenheitToCelsius(farenheit):
    celsius = (5/9)*(farenheit - 32)
    print celsius

def isFreezing(farenheit):
    if(farenheitToCelsius(farenheit) <= 0):
        return true
    else:
        return false

print isFreezing(100) #100°F
print isFreezing(0)      #0°F
```

b) De volgende functie zou moeten controleren of een element voorkomt in een lijst en geeft de (eerste) index terug indien dit het geval is.

```
def find(list, element):
    for i in range(list):
        if (list[i] == element):
            return i
        else:
            return 'Element not found in list'

list = [1,4,7,6,2,12,8,9]
print find(list, 7)
```

c) Deze functie hoort de volgorde van een lijst om te draaien.

```
def reverseList(list):
    reversed = ()
    for i in range(len(list), 0, -1):
        reversed.append(list[i])
        return reversed
list = (0,1,2,3,4,5)
print reverseList(list)
```

d) Als resultaat hoort deze functie een string terug te geven met een vooraf bepaald aantal keren *bla.*

```
def bla(amount): #returns "bla" amount times
    i = 0
    while i < amount:
          resultString += bla
    return resultString
print bla(5)
```

e) Deze functie hoort te controleren of een lijst van klein naar groot gesorteerd is. Het resultaat hoort uiteraard een booleaanse waarde te zijn.

```
def gesorteerd(lijst):
    for i in range(len(lijst)):
        eerste-element = lijst[i]
        tweede-element = lijst[i+1]
        if(eerste-element <= tweede-element):
            return “true”
        else:
            return “false”
```

f) De volgende functie bevat géén fouten maar is geschreven door een asociale programmeur die het concept “time is money” iets te extreem toepast. Aan jou om uit te vinden wat de functie doet en waarvoor welke parameter staat. Waarom is deze functie zo moeilijk te ontcijferen? Pas de functie zodanig aan zodat de overzichtelijkheid stijgt maar het resultaat hetzelfde blijft. (De while lus mag veranderd worden naar een for lus etc.)

```
    def f(x,y,z,k=0):
        a = 0
        b = y
        if(k+y <= len(x)):
            while (b > 0):
                if(x[y-b+k]%z == 0 and x[y-b+k]!=0):
                    a+=1
                b-=1
        return a
```

Vraag 3: Geneste lijsten (matrices) en I/O

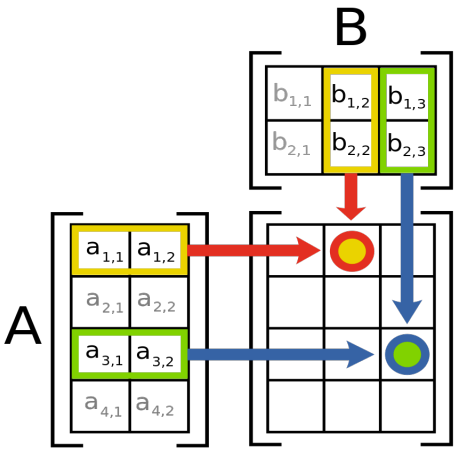
a) Gegeven het tekstbestand matrix1.txt. Dit bestand bevat de waarden van een 4x6 matrix. Maak een functie die de waarden uit het bestand inleest in een tweedimensionale lijst. Je hebt dus uiteindelijk nog maar één enkele variabele die alle waarden bevat. Zorg ervoor dat je functie een matrix van willekeurige grootte kan inlezen.

b) Maak een nieuwe functie die een tweedimensionale lijst wegschrijft naar een bestand, op dezelfde manier als de data in het bestand 'matrix1.txt' is gesorteerd. Test je functie door het resultaat van vraag 1a) naar een nieuw bestand weg te schrijven.

c) Maak een nieuwe functie die het product van twee matrices berekent en teruggeeft als een multidimensionale lijst. De twee matrices als input zijn zelf ook multidimensionale lijsten. De functie moet tevens nagaan of de dimensies van de matrices wel geldig zijn om het product te kunnen uitrekenen.

Ter herhaling de formule van matrixvermenigvuldiging:

$$C = A \times B \qquad C[i, j] = A[i, 1] * B[1, j] + A[i, 2] * B[2, j] + \ ... + A[i, n] * B[n, j]$$



De functie maakt gebruik van 3 verschillende indexen: i, j en n. De functies hiervan zijn als volgt:

- Index i bepaalt de geselecteerde rij van matrix A en resultaat
- Index j bepaalt de geselecteerde kolom van matrix B en resultaat
- Index n wordt gebruikt om de elementen in een rij (matrix A) en kolom (matrix B) aan te spreken, om deze vervolgens te vermenigvuldigen en te sommeren

d) Test je functie door respectievelijk matrices A en B in te lezen uit bestanden matrix1.txt en matrix2.txt en de vermenigvuldiging te maken:

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 7 & 8 & 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 & 17 & 18 \\ 19 & 20 & 21 & 22 & 23 & 24 \end{bmatrix} \times \begin{bmatrix} 50 & 51 & 52 \\ 53 & 54 & 55 \\ 56 & 57 & 58 \\ 59 & 60 & 61 \\ 62 & 63 & 64 \\ 65 & 66 & 67 \end{bmatrix} = \begin{bmatrix} 1260 & 1281 & 1302 \\ 3330 & 3387 & 3444 \\ 5400 & 5493 & 5586 \\ 7470 & 7599 & 7728 \end{bmatrix}$$