OEFENINGEN PYTHON – REEKS 3

Iteraties

1. Schrijf volgende for-lus met een while-lus:

```
for k in range(9,0,-1):
        print "k= ", k
```

2. Schrijf een functie die het grootste en kleinste element van een lijst vindt zonder gebruik te maken van ingebouwde functionaliteit zoals sort, min en max.

Geneste Lijsten

Bij de introductie van lijsten hebben we gezien dat een lijst een collectie is van elementen van om het even welk type. Aangezien een lijst op zich ook een type is, is het mogelijk om lijsten in lijsten te creëren. Voortaan zullen we deze 'lijsten in lijsten' constructie geneste lijsten noemen. Het aanspreken van elementen gebeurt door meerdere indexen te gebruiken.

1. Schrijf een functie die een geneste lijst meekrijgt, en de lengte van de langste lijst teruggeeft.
   Vb. [[1,2,3],[4,5],[6,7,8,9],[10]] → de functie zou 4 moeten teruggeven: de langste lijst is [6,7,8,9] en zijn lengte is 4.

2. Schrijf een functie die een geneste lijst meekrijgt en de gemiddelden berekent van elk j^{de} element in elke sub-lijst. Gebruik hierbij de functie uit vraag 5a. Let op: de lijst van gemiddelden is even lang als de langste sub-lijst.
   Vb. [[1,2,3],[4,5],[6,7,8,9],[10]] → Het resultaat van de functie is de lijst [5.25, 4.67, 5.5, 9.0] (komende van [(1+4+6+10)/4,(2+5+7)/3,(3+8)/2,9/1])

Praktijkvoorbeeld

Nu zullen we een probleem uit de praktijk proberen op te lossen aan de hand van Python. Het vraagstuk betreft een schuine worp: een voorwerp wordt vanop aarde met een bepaalde beginsnelheid in een schuine hoek weggeschoten. In de lessen fysica van het middelbaar heb je zeker zulke vraagstukken moeten oplossen, waarbij je bijvoorbeeld de totaal afgelegde weg of de maximumhoogte van het voorwerp moest berekenen. Door middel van Python kunnen we dit soort problemen aanpakken door de beweging te simuleren.

We gaan hiervoor een functie maken waaraan je de volgende gegevens kunt meegeven:

- De hoek **α** met het aardoppervlak
- De beginsnelheid $\mathbf{v_0}$
- De berekeningsfrequentie **f** waarmee we de simulatie zullen uitvoeren

De functie geeft het volgende terug:

- Twee lijsten van coördinaten **x** en **y** waarop het voorwerp zich bevind op opeenvolgende tijdstippen. De tijdspanne tussen elke 2 berekeningspunten wordt bepaald door de waarde van **f**. Let op: de functie geeft twee lijsten terug!

Volgende formule zou je zelf moeten kunnen vinden:

$$\begin{cases} x = v_0 \cos\alpha \ t \\ y = v_0 \sin\alpha \ t - \dfrac{gt^2}{2} \end{cases}$$

Laat je functie nu de opeenvolgende waarden **[x,y]** berekenen door **t** stapsgewijs te verhogen, waarbij het aantal stappen per seconde gegeven is door de berekeningsfrequentie **f**. De berekening dient te stoppen wanneer het voorwerp terug de grond raakt.

Tip: de “math” module kan gebruikt worden om sin() en cos() te berekenen (gebruik radialen). Je zult deze bibliotheek eerst moeten importeren.

Om de simulatie grafisch voor te stellen kan je de waarden **[x,y]** in Python kunnen plotten. Dit kan met behulp van de functies in de matplotlib/pylab bibliotheek. Je kan de waarden die je berekend heb netjes in een grafiek plotten aan de hand van de volgende code:

```
from pylab import figure,plot,title,xlabel,ylabel,show

x,y = schuine_worp(alfa,v0,0.01)

figure()
plot(x,y)
title('Schuine worp')
xlabel('Afstand (m)')
ylabel('Hoogte (m)')
show()
```

File I/O

Je hebt het bestand kunnen bemachtigen waar de examencijfers[1] van Informatica in opgeslagen worden. Download het bestand 'data.csv' van de website. Op elke lijn van deze file staan gegevens over 1 student in de vorm: NAAM;voornaam;groepnaam;examencijfer, gesorteerd volgens familienaam.

1. Schrijf een functie die dit bestand inleest en de gegevens als een **geneste lijst** teruggeeft. Elk element in de lijst is dus weer een lijst die de gegevens van een student bevat.

Files lezen is vrij eenvoudig in Python. Met een **for loop** kan je de verschillende lijnen in een tekstbestand één voor één verwerken:

```
f = open('data.csv','r')
for line in f:
   # doe iets met line, bv printen
   print line
f.close()
```

2. Schrijf een functie die in de lijst van vraag 1 de punten van een student opzoekt en teruggeeft. Deze functie neemt een naam en voornaam aan en geeft een getal terug.
3. Schrijf een functie die de punten van een bepaalde student in de lijst kan aanpassen en schrijf dan een tweede functie die deze lijst kan opslaan. Je overschrijft dus de bestaande data.csv file.
4. Schrijf een functie die de lijst van vraag 1 als input aanneemt en 6 lijsten teruggeeft, één lijst per groep (IA-A,IA-B,IR-A,IR-B,IR-C,IR-D).
5. Schrijf een functie die een studentenlijst als input aanneemt en één lijst teruggeeft die de behaalde punten als **float**s bevat.
6. Schrijf een functie die van een lijst met examenresultaten een histogram berekent: i.e. tel hoeveel keer het cijfer 0 voorkomt, hoeveel keer 1 voorkomt, etc. tot en met 20. Normaliseer dit histogram door elk van die getelde waarden te delen door het totaal aantal elementen in de lijst en print je histogram naar de shell door voor elk cijfer te zeggen hoeveel % van de studenten dit cijfer behaalde.

Je kan ook histogrammen mooi plotten door weer gebruik te maken van de pylab library op de volgende manier:

```
from pylab import hist,show,figure,title

figure()
n, bins, patches = hist(puntenlijst, 20, range=(0,20), normed=0, histtype='stepfilled')
title('Groep x')
show()
```

[1] Dit zijn uiteraard fictieve punten die willekeurig gegenereerd werden. Voel je vooral niet aangesproken als jij slechte punten zou hebben.