Inleiding programmeren

1e jaar natuur- en sterrenkunde

Universiteit van Amsterdam

Tentamen 2016-2017 (blok 1)

28 oktober 2016, 9:00-11:30 uur

Dit tentamen bestaat uit 5 vragen. Er zijn 10 punten te verdelen en je moet minimaal drie vragen goed beantwoorden (6 punten) om te slagen.

Regels:

- Schakel wifi uit: geen toegang tot internet toegestaan tijdens dit tentamen.
- Je mag op je computer alléén een exemplaar van de Python Shell open hebben, en een editor waarin je je programma's schrijft. Geen enkel ander programma mag open staan, dus ook geen Internet Explorer.
- Om in te leveren steek je je hand op en je mag dan samen met de assistent je tentamen submitten op de website. Je kan dan de zaal verlaten. Ook als het tentamen afgelopen is moet je wachten tot je je opdracht mag insturen.
- Schrijf je naam en studentnummer bovenaan je file.
- In elke opdracht schrijf je één functie; voor uitvoer van het berekende antwoord gebruik je in elke functie gewoon `print`-opdrachten.
- Dit opgavenblad moet weer inleverd worden.
- Je mag in dit tentamen geen functies uit de `numpy` bibliotheek gebruiken

Opgave 1: Lijsten manipuleren (2 punten)

Schrijf een functie `Opgave1(L, deler)` die van een gegeven lijst L bepaalt hoeveel getallen in de lijst een veelvoud zijn van het getal deler (een geheel getal groter dan 0) en zowel het aantal veelvouden als de veelvouden zelf op het scherm print.

Als je in je programma de functie aanroept met de volgende input-lijst en deler:

```
L = [2,3,7,6,39,12,3,7,9,6]
deler = 3
Opgave1(L,deler)
```

dan moet er op het scherm de volgende output verschijnen:

```
Er zijn 7 getallen in de lijst die een veelvoud zijn van 3
De lijst van veelvouden = [3,6,39,12,3,9,6]
```

Opdracht: roep de functie aan met de volgende input-lijst en deler:

```
L = [244,629,6401,75,1369,333,456,888,2701,6609]
deler = 37
Opgave1(L,deler)
```

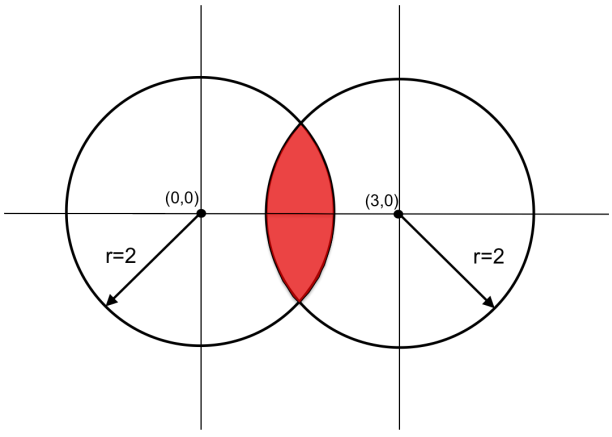
Opgave 2: oppervlakte overlappende cirkels (2 punten)

Schrijf een functie `Opgave2()` die de oppervlakte berekent van het rode gebied (twee overlappende cirkels) met behulp van de Monte-Carlo methode.

Schets van de situatie:

cirkel 1:
middelpunt = (0,0)
straal = 2

cirkel 2:
middelpunt = (3,0)
straal = 2



Ga bij het bepalen van de oppervlakte als volgt te werk:

- Gooi random punten in een rechthoek die beide cirkels omsluit. Voor elk punt (x_i, y_i) trek je dus 2 random getallen: x_i en y_i

- Bepaal de fractie van het aantal punten dat in het rode gebied valt
- Bereken uit deze fractie de oppervlakte van het rode gebied

Als output op je scherm verschijnt dan (met 2 decimalen):
`De oppervlakte van de overlap tussen de twee cirkels is ...`

Tip: als x-waarde van de rechter cirkel op $x = 0 (x = 4)$ ligt is de overlap $4\pi(0)$.

Benodigde bibliotheken: om random getallen te gebruiken heb je de `random()` functie nodig. Zorg dat je programma dus begint met: `import random`.

Opgave 3: Snijpunt 2 grafieken (2 punten)

We definiëren de volgende 2 functies:

$$\begin{aligned} f(x) &= ax^2 + bx + c \\ g(x) &= dx + e \end{aligned}$$

Schrijf een functie `Opgave3(a,b,c,d,e)` die voor gegeven waardes van a, b, c, d en e bepaalt of er snijpunten zijn. Als dat zo is, wat zijn dan de x-waardes van die snijpunten ? Geef je antwoord op 2 decimalen nauwkeurig.

Deze som kan je analytisch oplossen. Dit probleem vertaalt zich namelijk naar het zoeken van nulpunten van de functie $h(x) = f(x) - g(x) = ax^2 + (b-d)x + (c-e) = 0$. Gebruik vervolgens de abc-formule om de nulpunten te vinden.

deelopgave 1: roep je functie aan met de volgende parameters:

```
Opgave3(1,2,3,4,5)
```

Er zijn dan 2 snijpunten en er moet op het scherm verschijnen:
`Er zijn 2 snijpunten:  x1 = -0.73 en x2 = 2.73`

deelopgave 2: roep je functie aan met de volgende parameters:

```
Opgave3(5,4,3,2,1)
```

De grafieken kruisen elkaar in dat geval niet. Op het scherm moet dan verschijnen:
`Er zijn geen snijpunten`

Opgave 4: Wiskunde: perfecte getallen (2 punten)

Een perfect getal is een getal waarvan de som van de delers (met rest 0) het getal zelf is. Door voor elk getal de som van de delers te bepalen kan je evalueren of het een perfect getal is of niet. Het getal 12 bijvoorbeeld is `geen` perfect getal. De som van de delers (1,2,3,4,6) is namelijk 16 en dat is `niet` gelijk aan 12. Een voorbeeld van een perfect getal is 6. De som van de delers van 6 (1,2,3) is namelijk precies 6.

Schrijf een functie `Opgave4()` die de eerste 4 perfecte getallen vindt. De eerste 3 perfecte getallen zijn kleiner dan 500, maar vind ook de vierde.

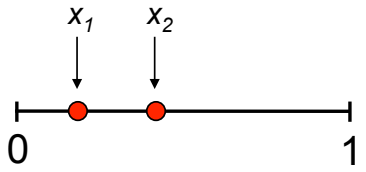
Als output op je scherm verschijnt dan:

```
Perfect getal 1 = 6
Perfect getal 2 = ...
Perfect getal 3 = ...
Perfect getal 4 = ...
```

Opgave 5: Gemiddelde afstand tussen 2 punten (2 punten)

Schrijf een functie `Opgave5()` die de gemiddelde afstand tussen 2 punten op een lijn bepaalt. Gebruik de volgende strategie:

1) Genereer twee random punten op de lijn: x_1 en x_2 en bepaal de onderlinge afstand (Δ)
2) Doe dit bovenstaande een groot aantal (N) keer en bewaar de totale afstand (som afstanden)
3) Bereken de gemiddelde afstand door de totale afstand te delen door N.



Zorg dat je programma ook bijhoudt wat de fractie van punten is waarvoor geldt dat de punten relatief dicht bij elkaar liggen (f_{12}: $0.1 < \Delta < 0.2$) ten opzichte van de kans dat ze relatief ver van elkaar liggen ($f_{89} : 0.8 < \Delta < 0.9$).

Aan het eind van je programma wordt dus op het scherm geprint (met 2 decimalen nauwkeurigheid):

```
De gemiddelde afstand tussen 2 punten op een lijn = ...
De relatieve kans f12/f89 = ...
```

Om random getallen te gebruiken heb je de `random()` functie nodig. Zorg dat je programma dus begint met: `from random import *`