PLANETENSTELSELS - WERKCOLLEGE 1 EN 2

Opdracht 1: Introductie Python

Tijdens dit werkcollege en tijdens je verdere studie zul je vaak gebruik willen maken van natuurkundige constanten en veel voorkomende natuurkundige formules. In deze eerste opdracht gaan we een `python` module maken die de waardes van deze constanten en een aantal standaard natuurkundige functies bevat. In de rest van het college zullen deze functies en constanten terugkomen. Om de opdrachten uit te voeren, moet je de `numpy` en `matplotlib` modules van `python` gebruiken. Deze modules zijn standaard beschikbaar op de computers van de Sterrewacht.

(1) Maak een `python` module met de naam 'astronomy' die constantes bevat met de waardes (in SI-eenheden) van de volgende natuurkundige constanten: Newton's Gravitatie constante (G), de constante van Planck (h) en lichtsnelheid (c). Je maakt een module door je broncode op te slaan als *modulenaam.py*; onze module wordt dus `astronomy`.
Tip: Zet als commentaar de eenheden erbij.

In `python` laad je een module met:

```
import astronomy
```

Uiteindelijk is de bedoeling dat je deze constantes in `python` op de volgende manier kunt gebruiken:

```
import astronomy

# druk de constante G af
print astronomy.G
# zet snelheid op 0.2 maal de lichtsnelheid
velocity = 0.2 * astronomy.c
```

Tip: Je kunt de module naam ook afkorten met het `import .. as` commando. Als je een reeds geladen module wilt herladen (bijvoorbeeld omdat je de code hebt veranderd), gebruik dan het `reload` commando:

```
import astronomy as p

# druk de constante G af
print p.G
```

Date: 10 februari 2014.

(2) Verder is het vaak handig om SI eenheden te kunnen omrekenen in eenheden die in de sterrenkunde vaker gebruikt worden. Voeg daarom ook de volgende astronomische eenheden toe: AU, zonsmassa, straal van de zon, aardmassa, straal van de aarde, massa van Jupiter en parsec. *Tip: Zet ook hier eenheden erbij.*

(3) Het is ook handig om veel gebruikte natuurkundige formules in een module op te nemen. Schrijf een functie die de zwaartekracht F berekent voor twee gegeven massa's m_1 en m_2 op een onderlinge afstand r.

(4) Gebruik je zwaartekrachtfunctie om de versnelling door zwaartekracht die het internationale ruimtestation ISS voelt als gevolg van (a) de aarde, (b) de maan en (c) de zon. Je mag ervan uitgaan dat het ISS zich op een cirkelbaan op 407 km boven het aardoppervlak beweegt. Neem aan dat het ISS zich tussen de aarde en de maan bevindt.

(5) Op de website van het vak kun je enkele eigenschappen van alle 8 planeten in het zonnestelsel vinden, gecodeerd als python lists. Kopieer deze lists naar je module zodat je ze in het vervolg makkelijk kan gebruiken.

(6) Bereken de gemiddelde dichtheid van elke planeet in kg/m^3 en leg uit wat dit je vertelt over de planeten. Vergelijk deze dichtheden met de dichtheden van alledaagse materialen als water, ijs, steen, stoom, etc.
Tip: Verander de definities van je lists in numpy arrays om de dichtheid te berekenen. Hieronder staat hoe je dat doet voor een voorbeeld. De module bevat ook het getal π: `numpy.pi`.

(7) Bereken de valversnelling op het oppervlak van elke planeet. *Tip: De variabelen van functies kunnen ook arrays zijn.*

Met `a=[2,4,6,8]` definieer je een `python` lijst. Op `python` lists kun je geen wiskundige operaties doen, zoals delen of machtsverheffen, op `numpy` arrays wel. In het voorbeeld hieronder geeft `python` een foutmelding als je een lijst door 2 probeert te delen. De `numpy` module voert het delen door 2 uit op elk element in het array.De `numpy` module is standaard beschikbaar op de computers van de Sterrewacht en op de website van het college staan links naar de documentatie erover. De module kan geladen worden, op dezelfde manier als boven beschreven is voor de astronomy module.

```
>>> a=[2,4,6,8]
>>> print a/2
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: unsupported operand type(s) for /: 'list' and 'int'
>>> import numpy
>>> a=numpy.array([2,4,6,8])
>>> print a/2
array([1, 2, 3, 4])
```

Opdracht 2: Snelheden

Bij interactie tussen meerdere objecten is de ontsnappingssnelheid van die objecten een belangrijke parameter voor de dynamica van het gehele systeem. Voor een object, dat beinvloed wordt door een zwaartekrachtveld, is het de minimale snelheid die het nodig heeft om dat zwaartekrachtveld te ontsnappen.

(1) Leid de formule voor de ontsnappingssnelheid af. Schrijf een `python` functie die de ontsnappingssnelheid voor een gegeven massa en een gegeven afstand tot die massa berekent.

(2) Verleden jaar is het ESA gelukt om op een ruimtevaartuig op een komeet te landen. De sonde Rosetta werd 10 jaar geleden vanaf de aarde gelanceerd en bereikte vorig jaar de komeet 67P. Bereken wat de maximale snelheid is die deze komeet mogelijkerwijs kan hebben, gegeven dat de komeet in een elliptische baan om de zon beweegt en dat de minimale afstand tot de zon 1.24 AU is. Bedenk hiervoor eerst op welk punt in de baan de snelheid van de komeet maximaal is, en ga vervolgens na hoe hoog deze snelheid maximaal kan zijn.

(3) We gaan er nu van uit dat de snelheid van de komeet, op het punt waar zijn snelheid het grootst is, 95% is van de snelheid die je in de vorige vraag hebt uitgerekend. Rosetta ontmoette de 67P op een afstand van 3.9 AU van de zon. Wat was dan de snelheid van de komeet ten tijde van de ontmoeting?

(4) Na het bereiken van 67P laat Rosetta op een hoogte van 30 km boven het oppervlak de lander Philae zakken. Als Philae het oppervlak bereikt, stuitert deze terug omhoog. De snelheid waarmee Philae omhoog stuitert, is de helft van de snelheid waarmee Philae het oppervlak raakte. Als de snelheid na de stuiter hoog genoeg is, kan de lander wegzweven van de komeet. Deze stuitersnelheid hangt af van de snelheid waarmee Philae Rosetta verliet.
Bereken of de lander wegzweeft voor snelheden tussen 0 km/s en 3 km/s. Neem aan dat de straal van de komeet 2 km is, dat deze 10^{13} kg weegt en dat Philae 100 kg weegt. *Hieronder staat een voorbeeld hoe je arrays voor deze opgave kunt gebruiken.*

Het commando $(a > b)$ laat zien bij welke items in array a de vergelijking waar is en waar deze niet waar is. Het commando $a[(a > b)]$ geeft alle items van array a die groter zijn dan b. Als je een selectie wil maken die gelimiteerd is door 2 restricties, gebruik dan het & teken. Let op dat dit soort operaties alleen mogelijk zijn op numpy arrays.

```
>>> mijnarray=numpy.arange(1,5,1)
>>> print mijnarray>2
[False False  True  True]
>>> print mijnarray[mijnarray>2]
[3 4]
```

(5) Bereken hoe hoog boven het oppervlak van de komeet de lander stuitert als functie van de snelheid waarmee Philae Rosetta verlaat. Plot deze snelheid tegen de hoogte. Leg je resultaat uit.

Met `matplotlib` kun je data plotten met `python`. Grafieken (of plots) worden gebruikt om grote hoeveelheden data overzichtelijk weer te geven of om trends inzichtelijk te maken. Het goed kunnen plotten van data is dus onontbeerlijk in de rest van je studie. Hieronder staat een voorbeeld.

```
from matplotlib import pyplot as plt
import numpy

# Vul x array met nummers van 0 tot 100 in stappen van 10
x = numpy.arange(0, 100, 10)
# Vul y array met nummers van 0 tot 5 in stappen van -0.5
y = numpy.arange(5,0,-0.5)

# Maak de plot
plt.plot(x, y, 'ro', label='rode punten')
plt.plot(x, y, label='blauwe lijn')
plt.title("Title")
plt.xlabel("X-as")
plt.ylabel("Y-as")
plt.legend(loc='upper left')
#opslaan naar png format
plt.savefig('figuur.png')
# Weergeven op scherm
plt.show()
# Het pop up scherm moet handmatig gesloten worden, dit zorgt
# er ook voor dat de figuur vergeten wordt.
# Wil je alleen opslaan, laat pyplot dan de figuur vergeten,
# anders is deze ook zichtbaar in de volgende plot
plt.clf()
```

(6) Ruimtevaartorganisaties sturen regelmatig sondes naar interessante plekken in het zonnestelsel. De Voyager 1 en 2 zijn zelfs het zonnestelsel uitgestuurd. Kost het meer brandstof om een ruimtevaartuig het zonnestelsel uit te lanceren, dan om het in de Zon te sturen? Zijn interstellaire missies dus makkelijker of moeilijker dan missies naar de binnenste delen van het zonnestelsel?

Uitwerking

Maak een klein verslag over je antwoorden.
Let bij het maken van je uitwerking erop dat antwoorden met de juiste (en astronomisch zinnige!) eenheden gegeven worden en met een beredenering toegelicht worden. Ze hebben een zinnig aantal significante cijfers hebben (de afstand tussen Jupiter en de Zon is dus geen 778547200000m, maar 5.20AU).
Laat ook zien hoe je berekeningen hebt gedaan. Doe je dit niet, dan zal de vraag niet goed gerekend worden!
Antwoorden horen in het verslag te staan, de python code moet niet gebruikt hoeven worden om de antwoorden te krijgen.
Let erop dat alle grafieken een titel hebben en dat er eenheden langs de assen staan. Bedenk of de eenheden die je ziet op de assen, goed kunnen zijn. Leg uit wat er in een plot te zien is en of het resultaat is wat je verwacht.

De uitwerkingen moeten individueel digitaal ingeleverd worden (graag als PDF) met naam en studentnummer erop en de python code die je hebt geschreven moet mee gestuurd worden als losse file die meteen uit te voeren is (dus niet als tekst in het verslag). **Deadline voor het inleveren is 2 maart 2015 (begin hoorcollege).** Uitwerkingen kun je sturen naar herbonnet@strw.leidenuniv.nl (Linuxzaal 4de verdieping) of hoeijmakers@strw.leidenuniv.nl (Windowszaal 3de verdieping). Zodra het is nagekeken, ontvang je een email met commentaar en/of correcties op jouw ingeleverde werk.

Nuttige webadressen

- Website van het college:
  `http://home.strw.leidenuniv.nl/ linnartz/2015.html`
- Introductie in het gebruik van `python` is te vinden op
  `http://www.strw.leidenuniv.nl/local/computers/it_local/practicum/html/python.php?node=7` (deze link vind je ook op de website van het college).
- Informatie over `numpy` is te vinden op `http://numpy.scipy.org/` (deze link vind je ook op de website van het college).