Frank Theede, Klaus Rohe Oktober 09

Eine Einführung in die astronomische Programmierung mit Python –Teil 1

Es wird eine kurze Einführung in die Script-Sprache Python gegeben, die den Leser dabei unterstützen soll, Python für astronomische Berechnungen und Simulationen zu nutzen. Das Ziel ist es, Python als Leistungsfähiges Werkzeug für alle Arten von Berechnungen und Simulationen im Bereich der Amateurastronomie vorzustellen und auch Neulinge auf dem Gebiet der Programmierung zu ermutigen, in dieses faszinierende Gebiet einzusteigen. Es ist nicht die Absicht eine umfassende Einführung in Python zu geben, dazu gibt es gute Lehrbücher. Im Literaturverzeichnis sind zwei Bücher aufgeführt, welche in die Python- Programmierung einführen. Speziell (2) wendet sich an Programmieranfänger.

Warum Python?

Bevor wir einen Überblick und kurze Einführung in die Programmiersprache Python geben, werden wir begründen, warum sich Python unserer Ansicht nach besonders gut für die astronomische Programmierung eignet:

1. Python ist Open Source und für alle Betriebssysteme verfügbar.
2. Die Struktur von Python ist sehr einfach und Python Programme sind sehr gut lesbar. Daher ist sie für Programmieranfänger leicht zu erlernen. Der Einstieg wird ebenfalls erleichtert, weil sie prozedurale und objektorientierte Programmierung unterstützt.
3. Für Python gibt es frei verfügbare, mächtige Ergänzungsbibliotheken für Numerik, Bildverarbeitung, Simulation und grafische Darstellung von Funktionen und Messreihen.
4. Python wird zunehmend im technisch wissenschaftlichen Bereich genutzt, siehe dazu http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=4160249
5. Python hat eine große Anhängerschaft, siehe dazu http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html

Historie und Bedeutung von Python

Python wurde Anfang der 1990er Jahre von Guido van Rossum am Centrum voor Wiskunde en Informatica (NL) als Nachfolger für die Lehrsprache ABC entwickelt. Ursprünglich war Python für das verteilte Betriebssystem Amoeba gedacht. Der Name Python bezieht sich auf die britische Komikergruppe Monty Python und nicht auf die gleichnamige Familie von Riesenschlangen. Python Programme werden in einen Zwischencode übersetzt und interpretiert. Python ist für alle gängigen Betriebssysteme verfügbar: Microsoft Windows, LINUX, Mac OS und UNIX Systeme. Python ist Open Source und wird von der Python Software Foundation (http://www.python.org/) weiterentwickelt. Es gibt eine Python-Implementierungen für die Java Virtual Machine (JVM), genannt Jython und für die .NET Common Language Runtime (CLR), welche IronPython heißt. Python hat sich im Laufe der Jahre als beliebte Programmiersprache im Bereich der technisch-wissenschaftlichen Anwendungen (siehe (Langtangen, 2008) und (Oliphant, 2007)), sowie im Bereich Web-Applikationsentwicklung etabliert. Python wird von der NASA für die Missionsplanung des Space Shuttle eingesetzt (http://www.python.org/about/success/usa/.

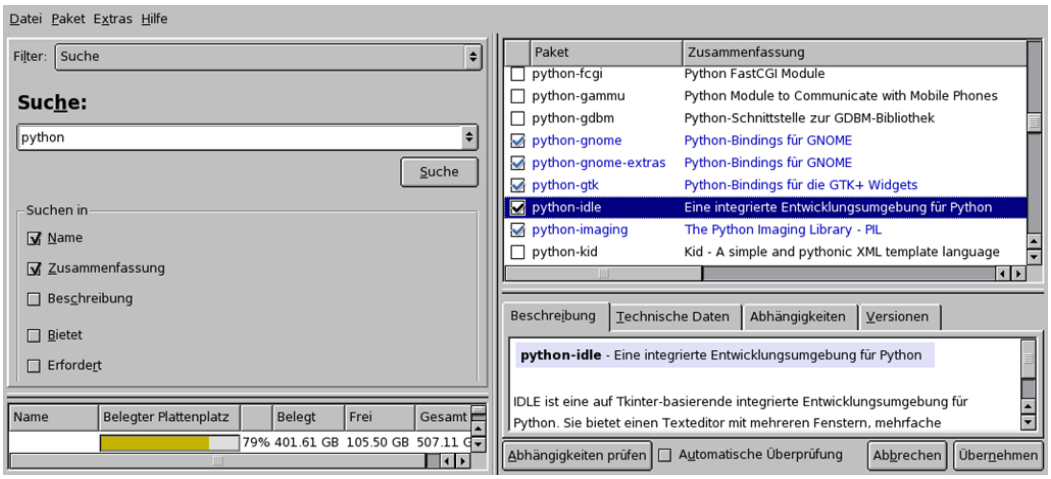
Die aktuellen Versionen von Python sind Python 2.6.2 und Python 3.1. Mit Python 3 wurden neue Elemente in die Sprache eingeführt, welche die Abwärtskompatibilität mit Python 2.x aufheben.

Installation von Python

Es wird kurz beschrieben wie Python unter den gängigen Betriebssystemen installiert wird.

Installation unter LINUX

Bei den meisten gängigen Linux-Distributionen wird Python schon mit dem Basissystem installiert, da manche Systemtools darauf aufsetzen. Wenn notwendig, kann zum Beispiel unter OpenSuse mit Yast2 aber auch das notwendige Paket einfach installiert werden. Zu beachten ist hierbei, dass die Python IDLE (s.u.) zusätzlich einzurichten ist:

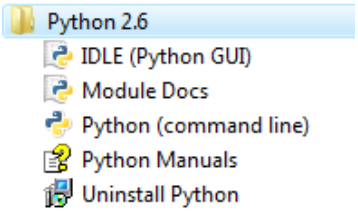


Ein Download der Python-Software und händische Installation ist meist nicht nötig und sollte nur dann durchzuführen sein, wenn z.B. eine neue Version noch nicht vom Distributor angeboten wird.

Unter Linux ist auch die Verwendung von Eclipse mit dem Plug-in PyDev sehr anzuraten, da dann auch die Nutzung des Debuggers wesentlich einfacher ist.

Installation unter Windows

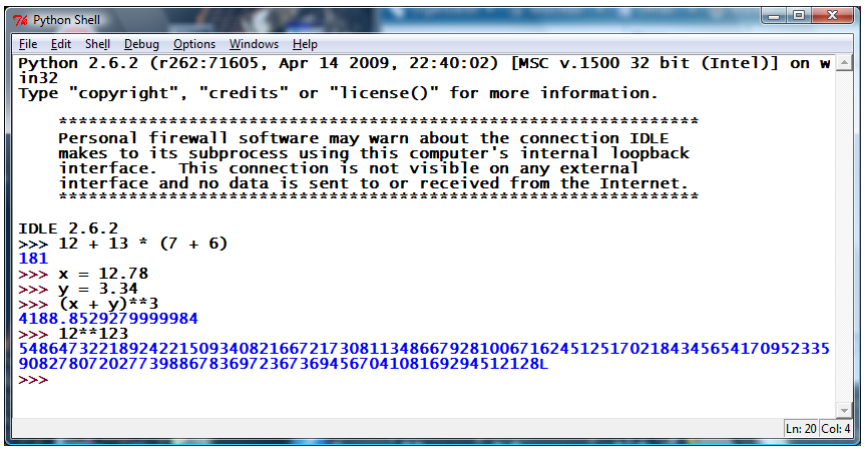
Von der Web-Seite http://www.python.org/ kann man ein Windows- Installationspaket („Windows Installer Package“) mit dem Namen python-2.6.2.msi herunterladen. Durch einen Doppelklick auf das Symbol des Installationspaketes wird die Installation gestartet. Die Installation erfolgt im Verzeichnis C:\Python26, wenn man nichts anderes angibt. Nach der Installation findet man die installierten Programme und Dokumentation im Startmenü im Ordner Python, wie in dem Screenshot unten zusehen.



Die Bestandteile im Python-Ordner, die für die Arbeit mit Python am wichtigsten sind, sind die interaktive Entwicklungsumgebung „***IDLE (Python GUI)***“ und die „***Python Manuals***“.

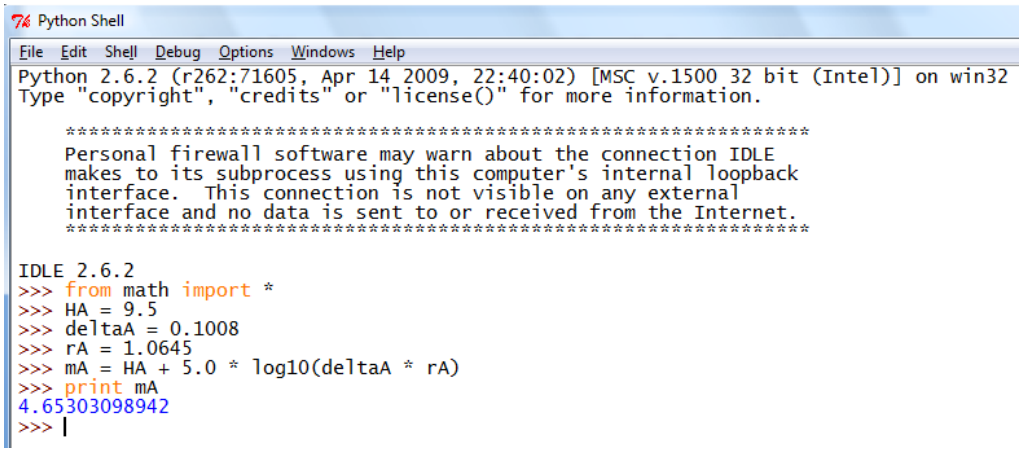
Python als programmierbarer Taschenrechner

Python ist eine interaktive Programmiersprache. Man kann die Python-Shell starten und Python-Befehle eingeben, die nach dem Drücken der <Enter>-Taste sofort ausgeführt werden. Die einfachsten Befehle sind solche zum Rechnen. Man kann Python also als „Taschenrechner“ benutzen, wie in dem Screenshot unten zusehen. Hier wurde die interaktive Python Entwicklungsumgebung IDLE benutzt

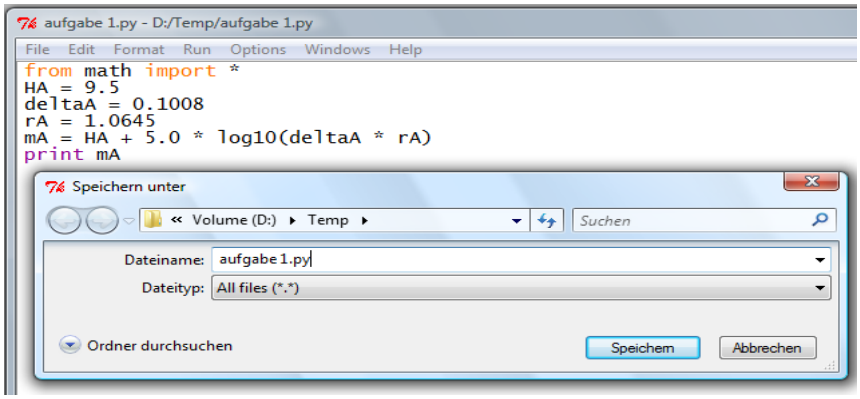


und einige Rechnungen ausgeführt. In Python bedeutet ***(x + y)**3*** bedeutet $(x + y)^3$, die letzte Rechnung in dem Screenshot, ***12**123***, weist darauf hin, dass Python im Bereich der ganzen Zahlen mit beliebig großen Werten Rechnen kann.

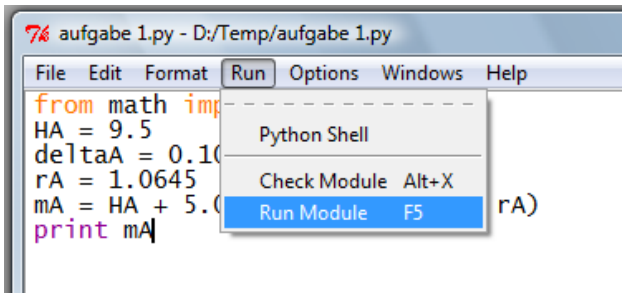
Im Folgenden wird die Programmiersprache Python anhand von Beispielen eingeführt, in denen Python als „programmierbarer Taschenrechner“ genutzt wird. Die Beispiele beziehen sich auf die Lösung der Aufgaben 1- 4 in der Rubrik „Zum Nachdenken“ der Zeitschrift Sterne und Weltraum, Ausgabe 06/2009. Als erstes startet man die interaktive Python Konsole IDLE. Der Ablauf der interaktiven „Taschenrechner-Sitzung“ mit Python zum Lösen der Aufgabe 1 ist in dem Screenshot unten dargestellt:

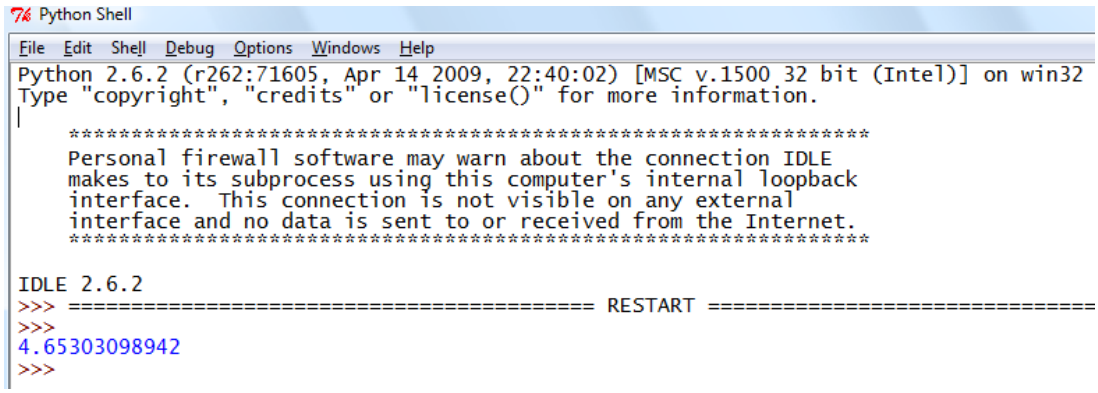


Damit man mathematische Funktionen und Konstanten zur Verfügung hat, wird in der ersten Zeile das Python Mathematikmodul über den Python-Befehl ***from math import ** importiert. Jede Eingabe wird mit der <Return>-Taste abgeschlossen. In der zweiten Zeile wird der Variablen ***HA***, welche die absolute Helligkeit von Apophis bezeichnet, der Wert 9.5 zugewiesen. In den beiden nächsten Zeilen werden den Variablen ***deltaA*** und ***rA*** die Werte 0.1008 respektive 1.0645 zugewiesen. Dann wird die scheinbare Helligkeit ***mA*** nach der in Referenz (1) angegebenen Formel berechnet. Die Funktion ***log10*** ist im Python Mathematikmodul definiert und berechnet den Logarithmus zur Basis 10. Mit dem Befehl ***print mA*** wird der berechnete Wert ausgeben, in diesem Fall ***4.65303098942***. Hat man vor das Ganze als wieder verwendbares Python-Skript zu speichern, so ist es am besten nach dem Start von IDLE im File-Menü über „New Window“ den Editor zu starten und die Zeilen in den Editor einzugeben. Bevor man das Skript über das Run-Menü des Editors ausführen kann, muss man es in einer Datei speichern, z.B. in der Datei „***aufgabe1.py***“. Python-Skript-Dateien haben die Endung ***py.*** Dies ist in dem Screenshot unten dargestellt.



Führt man das Skript über das Run-Menü des Editors aus (Eintrag „Run Module“ oder Drücken der Taste F5), so wird es in der Python-Shell ausgeführt. Das Ergebnis ist in den beiden Screenshots unten dargestellt.





Das vollständige Python-Skript zur Lösung der Aufgaben 1-4 aus Sterne und Weltraum (SuW) 06/2009 ist in den zwei Screenshots unten zu sehen.

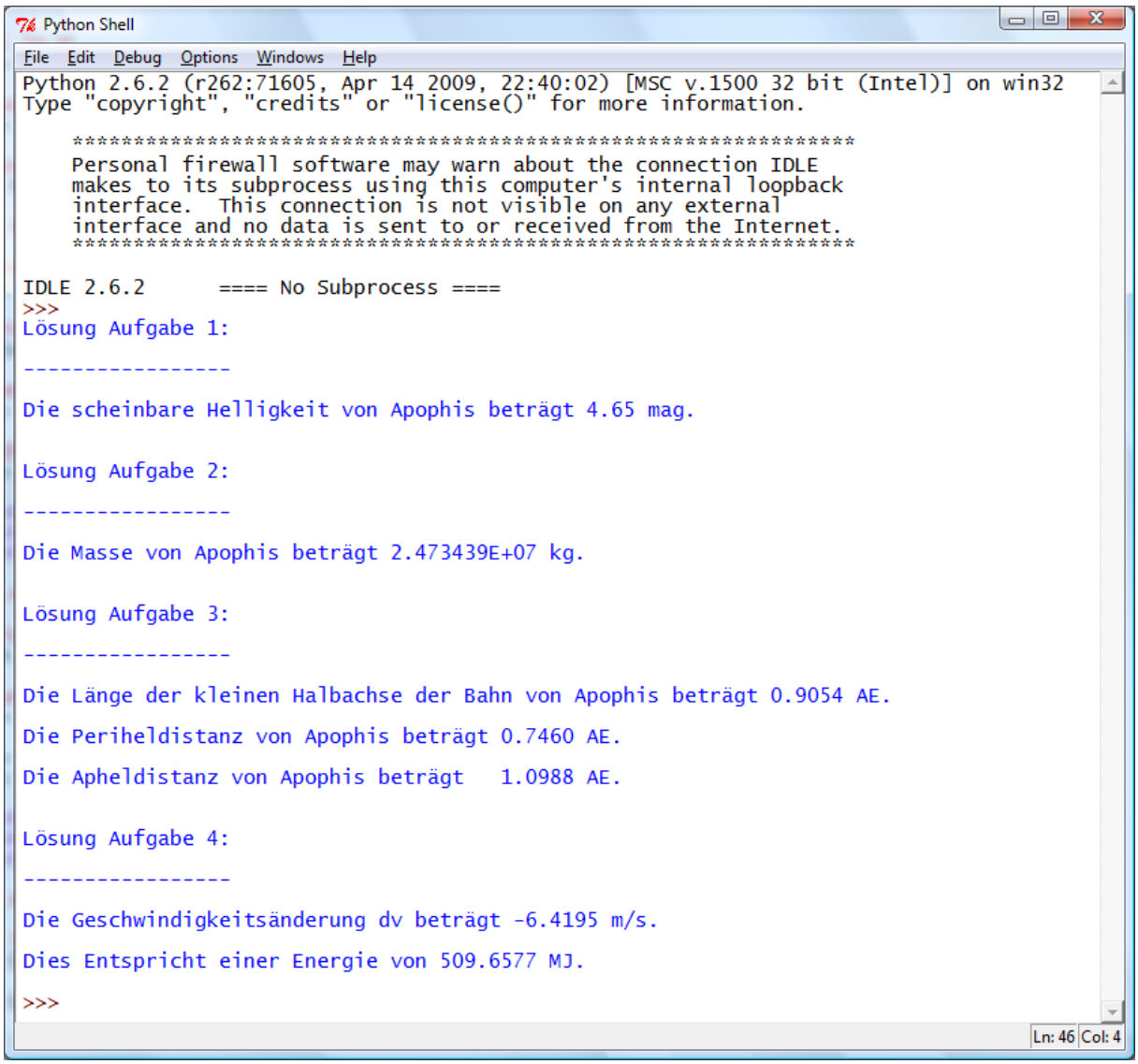
```
1  from math import *
2
3  # AE in m
4  AE = 149.6 * 1000000000.0
5
6  # Masse der Sonne in kg
7  Ms = 1.989E30
8
9  # Newtonsche Gravitationskonstante
10 G = 6.6743E-11
11
12 #  Aufgabe 1
13
14 # Absolute Helligkeit von Apophis in mag
15 HA = 9.5
16
17 # Erddistanz von Apophis in AE
18 deltaA = 0.1008
19
20 # Sonnendistanz von Apophis in AE
21 rA = 1.0645
22
23 # Scheinbare Helligkeit von Apophis
24 mA = HA + 5.0 * log10(deltaA * rA)
25
26 print "Lösung Aufgabe 1:\n"
27 print "-----------------\n"
28 print "Die scheinbare Helligkeit von Apophis beträgt %2.2f mag.\n\n" %(mA)
29
30 #  Aufgabe 2
31
32 # Dichte von Apophis in kg/m³
33 rhoA = 2.4
34
35 # Durchmesser von Apophis in m
36 dA = 270.0
37
38 VA = (4.0 / 3.) * pi * (dA / 2.0)**3
39 MA = VA * rhoA
40
41 print "Lösung Aufgabe 2:\n"
42 print "-----------------\n"
43 print "Die Masse von Apophis beträgt %E kg.\n\n" %(MA)
```

Python-Skript zum Lösen der Aufgaben 1-4 in SuW 06/09, Zeile 1 - 43

```
45  #  Aufgabe 3
46
47  # a in AE
48  a = 0.9224
49
50  e = 0.1912
51
52  b = a * sqrt(1.0 - e**2)
53
54  q = a * (1.0 - e)
55
56  Q = a * (1.0 + e)
57
58  print "Lösung Aufgabe 3:\n"
59  print "-----------------\n"
60  print "Die Länge der kleinen Halbachse der Bahn von Apophis beträgt %4.4f AE.\n" %(b)
61  print "Die Periheldistanz von Apophis beträgt %4.4f AE.\n" %(q)
62  print "Die Apheldistanz von Apophis beträgt   %4.4f AE.\n\n" %(Q)
63
64  #  Aufgabe 4
65
66  # a in m
67  a = a * AE
68
69  # b in m
70  b = b * AE
71
72  #B in m
73  B = 6500 * 1000.0
74
75  bm = b + B
76
77  def VQ(a, b):
78      return (b / (a + sqrt(a**2 - b**2))) * sqrt(G * Ms / a)
79
80  dv = VQ(a, b) - VQ(a, bm)
81
82  Ekin = 0.5 * MA * dv**2
83
84  print "Lösung Aufgabe 4:\n"
85  print "-----------------\n"
86  print "Die Geschwindigkeitsänderung dv beträgt %4.4f m/s.\n" %(dv)
87  print "Dies Entspricht einer Energie von %4.4f MJ.\n" %(Ekin / 1e6)
```

Python-Skript zum Lösen der Aufgaben 1-4 in SuW 06/09, Zeile 45 – 87

In Zeile 77 des Skripts wird durch „***def VQ(a, b):“*** eine Funktion definiert, welche die Formel für die Aphel-Geschwindigkeit berechnet, siehe dazu Referenz (3). Funktionen benutzt man für Formeln, die man öfter anwenden muss. Die Funktion „***VQ(a, b)“*** wird in der Zeile 80 dann benutzt um die in (3), Aufgabe 4, geforderte Geschwindigkeitsdifferenz zu berechnen. Führt man das Skript aus, so bekommt man folgende Ausgabe:



Im nächsten Teil der Serie über Python, wird weiter auf Funktionen in Python eingegangen, sowie auf Verzweigungen und Schleifen und gezeigt wie man in Python Deklination und Aufgangsazimuts der Sonnen berechnen kann.

Literaturhinweis

(1) Hans Petter Langtangen, A Primer on Scientific Programming with Python, ISBN: 978-3-642-02474-0, Springer 2009, http://www.springer.com/math/cse/book/978-3-642-02474-0
(2) Warren D. Sande, Carter Sande, Hello World! Computer Programming for Kids and Other Beginners, ISBN: 1933988495, Manning Publications 2009, http://www.manning.com/sande
(3) Quetz, A. M.: ZUM NACHDENKEN (99942) Apophis. In: Sterne und Weltraum 06/2009, S. 24