HANS-GEORG SCHUMANN

PYTHON FÜR KIDS

2. AUFLAGE

PROGRAMMIEREN LERNEN OHNE VORKENNTNISSE

mitp

Hinweis des Verlages zum Urheberrecht und Digitalen Rechtemanagement (DRM)

Inhaltsverzeichnis

Kleine Checkliste

Für
Janne, Julia, Katrin und Daniel

Hans-Georg Schumann

Python für Kids

Programmieren lernen ohne Vorkenntnisse

Impressum

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über <http://dnb.d-nb.de> abrufbar.

ISBN 978-3-7475-0241-9
2. Auflage 2020

www.mitp.de
E-Mail: mitp-verlag@sigloch.de
Telefon: +49 7953 / 7189 - 079
Telefax: +49 7953 / 7189 - 082

Lektorat: Katja Völpel
Covergestaltung: Christian Kalkert, Sandrina Dralle
Covergrafik: © trihubova / stock.adobe.com
electronic **pub**lication: III-satz, Husby, www.drei-satz.de

Einleitung

Python – wer denkt da nicht an eine Schlange? Eine, die zwar nicht giftig ist, aber einem sämtliche Knochen brechen und die Luft abschnüren kann. Du weißt natürlich, dass es hier bei Python um eine Programmiersprache geht.

Python wurde Anfang der 1990er-Jahre entwickelt. Der Name dieser Sprache geht jedoch nicht auf die gleichnamige Schlange zurück, sondern auf eine Truppe von englischen Komikern namens »Monthy Python«, die vor allem in den 70er-Jahren mit ihren skurrilen Filmen erfolgreich war (unter anderem mit »Das Leben des Brian«).

Natürlich hat Python viel von anderen Programmiersprachen übernommen. Sie hat sich einen Namen gemacht, weil sie als leicht erlernbar und übersichtlich gilt. In diesem Buch geht es um die bereits dritte Version von Python, die aktuell auch die neueste (und vielseitigste) ist.

Was heißt eigentlich Programmieren?

Wenn du aufschreibst, was ein Computer tun soll, nennt man das Programmieren. Das Tolle daran ist, dass du selbst bestimmen kannst, was getan werden soll. Lässt du dein Programm laufen, macht der Computer die Sachen, die du ausgeheckt hast. Natürlich wird er dann dein Zimmer nicht aufräumen und dir auch keine Tasse Kakao ans Bett bringen. Aber beherrschst du erst mal das Programmieren, kannst du den Computer sozusagen nach deiner Pfeife tanzen lassen.

Allerdings passiert es gerade beim Programmieren, dass der Computer nicht so will, wie du es gerne hättest. Meistens ist das ein Fehler im Programm. Das Problem kann aber auch

irgendwo anders im Computer oder im Betriebssystem liegen. Das Dumme bei Fehlern ist, dass sie sich gern so gut verstecken, dass die Suche danach schon manchen Programmierer zur Verzweiflung gebracht hat.

Vielleicht hast du nun trotzdem Lust bekommen, das Programmieren zu erlernen. Dann brauchst du ja nur noch eine passende Entwicklungsumgebung, und schon kann’s losgehen.

Was ist eine Entwicklungsumgebung?

Um ein Programm zu erstellen, musst du erst einmal etwas eintippen. Das ist wie bei einem Brief oder einer Geschichte, die man schreibt. Das Textprogramm dafür kann sehr einfach sein, weil es ja nicht auf eine besondere Schrift oder Darstellung ankommt wie bei einem Brief oder einem Referat. So etwas wird **Editor** genannt.

Ist das Programm eingetippt, kann es der Computer nicht einfach lesen und ausführen. Jetzt muss es so übersetzt werden, dass der PC versteht, was du von ihm willst. Weil er aber eine ganz andere Sprache spricht als du, muss ein Dolmetscher her.

Du programmierst in einer Sprache, die du verstehst, und der Dolmetscher übersetzt es so, dass es dem Computer verständlich wird. So etwas heißt dann **Compiler** oder **Interpreter**.

Python bietet solche Dolmetscher gleich für mehrere Betriebssysteme. Dein Computer kann also ein Windows-PC oder ein Linux-PC sein, ein Macintosh oder irgendein anderes Computersystem. Ein und dasselbe Python-

Programm kann so (eventuell mit kleinen Abweichungen) auf jedem beliebigen Computer funktionieren.

Schließlich müssen Programme getestet, überarbeitet, verbessert, wieder getestet und weiterentwickelt werden. Dazu gibt es noch einige zusätzliche Hilfen. Daraus wird dann ein ganzes System, die Entwicklungsumgebung.

Warum gerade Python?

Leider kannst du nicht so programmieren, wie dir der Schnabel gewachsen ist. Eine Programmiersprache muss so aufgebaut sein, dass möglichst viele Menschen in möglichst vielen Ländern einheitlich damit umgehen können.

Weil in der ganzen Welt Leute zu finden sind, die wenigstens ein paar Brocken Englisch können, besteht auch fast jede Programmiersprache aus englischen Wörtern. Es gab auch immer mal Versuche, z.B. in Deutsch zu programmieren, aber meistens klingen die Wörter dort so künstlich, dass man lieber wieder aufs Englische zurückgreift.

Eigentlich ist es egal, welche Programmiersprache du benutzt. Am besten eine, die möglichst leicht zu erlernen ist. Wie du weißt, bekommst du es in diesem Buch mit der Programmiersprache Python zu tun, die mittlerweile sehr weit verbreitet ist. (Willst du mal in andere Sprachen hineinschnuppern, dann empfehle ich dir z.B. eines der anderen Kids-Bücher über C++ oder Java.)

Der Weg zum guten Programmierer kann ganz schön steinig sein. Nicht selten kommt es vor, dass man die Lust verliert, weil einfach gar nichts klappen will. Das Programm tut etwas ganz anderes, man kann den Fehler nicht finden und

man fragt sich: Wozu soll ich eigentlich programmieren lernen, wo es doch schon genug Programme gibt?

Gute Programmierer werden immer gesucht, und dieser Bedarf wird weiter steigen. Und Python gehört dabei durchaus zu den erwünschten Sprachen. Wirklich gute Programmierer werden auch wirklich gut bezahlt. Es ist also nicht nur einen Versuch wert, es kann sich durchaus lohnen, das Programmieren in Python zu erlernen.

Die Entwicklungsumgebung

Um eine Entwicklungsumgebung für Python musst du dich nicht weiter kümmern, wenn dir eine einfache reicht. Die nämlich bekommst du kostenlos mit dem Python-Paket (sie heißt IDLE = »Integrated Development and Learning Environment«). Die werden wir hier ausgiebig benutzen.

Das komplette Paket kannst du dir von dieser Seite herunterladen:

https://www.python.org/

Dabei muss es nicht unbedingt die neueste Version sein. Dieses Buch bezieht sich auf Python 3 (aktuelle Versionen 3.8 oder 3.9).

Und was bietet dieses Buch?

Über eine ganze Reihe von Kapiteln verteilt lernst du

- das Basiswissen von Python kennen
- etwas über objektorientierte Programmierung

- mit Komponenten des Moduls `tkinter` zu arbeiten (das sind Bausteine, mit denen du dir viel Programmierarbeit sparen kannst)
- die grafischen Möglichkeiten von Python kennen
- etwas über den Umgang mit dem Spiele-Modul `pygame`
- wie man eigene `Game`- und `Player`-Klassen programmiert

Im **Anhang** gibt es dann noch zusätzliche Informationen und Hilfen, unter anderem über Installationen und den Umgang mit Fehlern.

Wie arbeite ich mit diesem Buch?

Grundsätzlich besteht dieses Buch aus einer Menge Text mit vielen Abbildungen dazwischen. Natürlich habe ich mich bemüht, alles so zuzubereiten, dass daraus lauter gut verdauliche Happen werden. Damit das Ganze noch genießbarer wird, gibt es zusätzlich noch einige Symbole, die ich dir hier gern erklären möchte:

Arbeitsschritte

- Wenn du dieses Zeichen siehst, heißt das: Es gibt etwas zu tun. Damit kommen wir beim Programmieren Schritt für Schritt einem neuen Ziel immer näher.

Grundsätzlich lernt man besser, wenn man einen Programmtext selbst eintippt oder ändert. Aber nicht immer hat man große Lust dazu. Deshalb gibt es alle Projekte im Buch auch als Download:

http://www.mitp.de/0239

Und hinter einem Programmierschritt findest du auch den jeweiligen Namen des Projekts oder einer Datei (z.B. → PROJEKT1.PY). Wenn du also das Projekt nicht selbst erstellen willst, kannst du stattdessen diese Datei laden (zu finden im Ordner PROJEKTE).

Aufgaben

Am Ende eines Kapitels findest du jeweils eine Reihe von Fragen und Aufgaben. Diese Übungen sind nicht immer ganz einfach, aber sie helfen dir, noch besser zu programmieren. Lösungen zu den Aufgaben findest du in verschiedenen Formaten ebenfalls im Verzeichnis PROJEKTE. Du kannst sie dir alle im Editor von Windows oder auch in deinem Textverarbeitungsprogramm anschauen. Oder du lässt sie dir ausdrucken und hast sie dann schwarz auf weiß, um sie neben deinen Computer zu legen. (Auch die Programme zu den Aufgaben liegen im Ordner PROJEKTE.)

Notfälle

Vielleicht hast du irgendetwas falsch gemacht oder etwas vergessen. Oder es wird gerade knifflig. Dann fragst du dich, was du nun tun sollst. Bei diesem Symbol findest du eine Lösungsmöglichkeit. Notfalls kannst du aber auch ganz hinten im Anhang B nachschauen, wo ein paar Hinweise zur Pannenhilfe aufgeführt sind.

Wichtige Stellen im Buch

Hin und wieder findest du ein solch dickes Ausrufezeichen im Buch. Dann ist das eine Stelle, an der etwas besonders Wichtiges steht.

Expertenwissen

Wenn du ein solches »Wow« siehst, geht es um ausführlichere Informationen zu einem Thema.

Was brauchst du für dieses Buch?

Installiert wird Python mit einem Setup-Programm in ein Verzeichnis deiner Wahl, z.B. D:\Python. Dort solltest du später auch deine Python-Projekte unterbringen.

Die Beispielprogramme in diesem Buch gibt es alle als Download von der Homepage des Verlages, falls du mal keine Lust zum Abtippen hast:

http://www.mitp.de/0239

Und auch die Lösungen zu den Fragen und Aufgaben sind dort untergebracht (alles im Ordner Projekte).

Betriebssystem

Die meisten Computer arbeiten heute mit dem Betriebssystem Windows. Davon brauchst du eine der

Versionen 7 bis 10. (Python gibt es unter anderem auch für Linux.)

Speichermedien

Auf jeden Fall benötigst du etwas wie einen USB-Stick oder eine SD-Card, auch wenn du deine Programme auf die Festplatte speichern willst. Auf einem externen Speicher sind deine Arbeiten auf jeden Fall zusätzlich sicher aufgehoben.

Gegebenenfalls bitte deine Eltern oder Lehrer um Hilfe.

Hinweise für Lehrer

Dieses Buch versteht sich auch als Lernwerk für den Informatik-Unterricht in der Schule. Dort setzt natürlich jeder Lehrer seine eigenen Schwerpunkte. Benutzen Sie an Ihrer Schule bereits ein Werk aus einem Schulbuchverlag, so lässt sich dieses Buch auch als Materialienband einsetzen – in Ergänzung zu dem vorhandenen Schulbuch. Weil dieses Buch sozusagen »von null« anfängt, ist ein direkter Einstieg in Python möglich – ohne irgendwelche anderen Programmierkenntnisse.

Ein wichtiger Schwerpunkt in diesem Buch ist die objektorientierte Programmierung (OOP). Auf die wichtigsten Eigenheiten (Kapselung, Vererbung und Polymorphie) wird ausführlich eingegangen. Ein anderer Schwerpunkt ist die Programmierung von Spielen. In den Projekten werden alle wesentlichen Elemente des Python-Wortschatzes wie auch die wichtigsten Grafik-Komponenten von `tkinter` eingesetzt. In den Lösungen zu den Aufgaben finden Sie weitere Vorschläge zur Programmierung.

Übungsmedien

Für den Informatik-Unterricht sollte jeder Schüler ein eigenes externes Speichermedium haben, um darauf seine Programmierversuche zu sichern. So wird verhindert, dass sich auf der Festplatte des Schulcomputers mit der Zeit allerlei »Datenmüll« ansammelt. Außerdem dient der eigene Datenträger dem Datenschutz: Nur der betreffende Schüler kann seine Daten manipulieren.

Auf die Dateien zum Buch verzichten?

Vielleicht ist es Ihnen lieber, wenn Ihre Schüler die Projekte alle selbst erstellen. Dann lassen Sie die Download-Dateien einfach (erst einmal) weg.

Regelmäßig sichern

Es kann nicht schaden, die Programmdateien, an denen gerade gearbeitet wird, etwa alle zehn Minuten zu speichern. Denn Computer pflegen gern gerade dann »abzustürzen«, wenn man seine Arbeit längere Zeit nicht gespeichert hat.

Kapitel 1
Erste Schritte

Hier geht es gleich ans »Eingemachte«. Nachdem wir Python installiert und gestartet haben, machen wir unsere ersten Gehversuche. Um später auch größere Programmprojekte erstellen zu können, brauchen wir das passende Werkzeug. Wir richten uns so komfortabel ein, dass schließlich auch dein erstes Programm entsteht.

In diesem Kapitel lernst du

- wie man Python startet
- Anweisungen für Ausgabe und Eingabe kennen
- was Variablen sind
- den Typ String kennen
- etwas über den Einsatz von IDLE
- wie man ein Programm erstellt und speichert
- wie man Python beendet

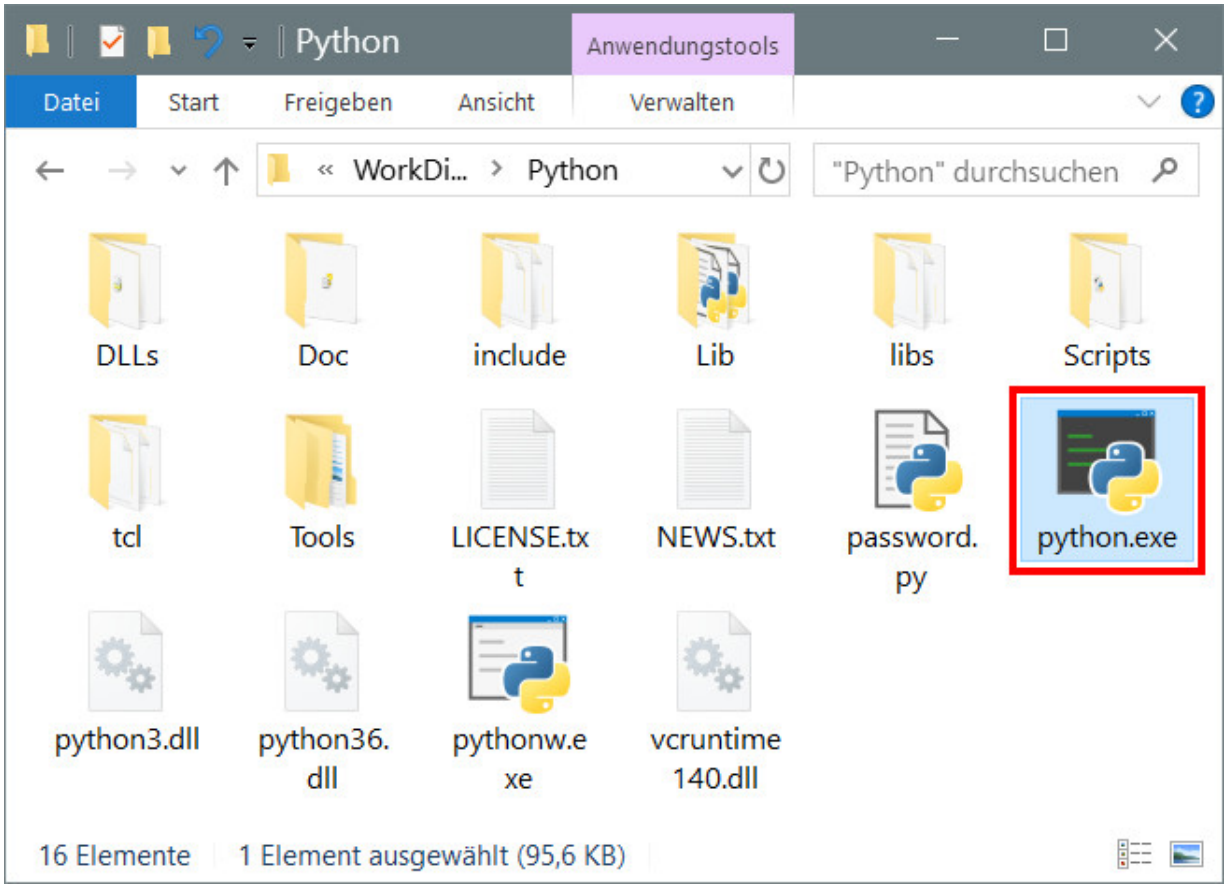
Mit Python loslegen

Bevor wir mit dem Programmieren anfangen können, muss **Python** erst installiert werden.

Die Installation übernimmt ein sogenanntes Setup-Program m. Genaues erfährst du im Anhang A. Hier musst du dir von jemandem helfen lassen, wenn du dir die Installation nicht

allein zutraust. Eine Möglichkeit, Python zu starten, ist diese:

- Öffne den Ordner, in dem du Python untergebracht hast – z.B. C:\PROGRAMME\PYTHON oder D:\PYTHON.



- Hier suchst du unter den vielen Symbolen das mit dem Namen PYTHON.EXE heraus. Doppelklicke auf das Symbol.

Wenn du willst, kannst du auch eine **Verknüpfung** auf dem Desktop anlegen:

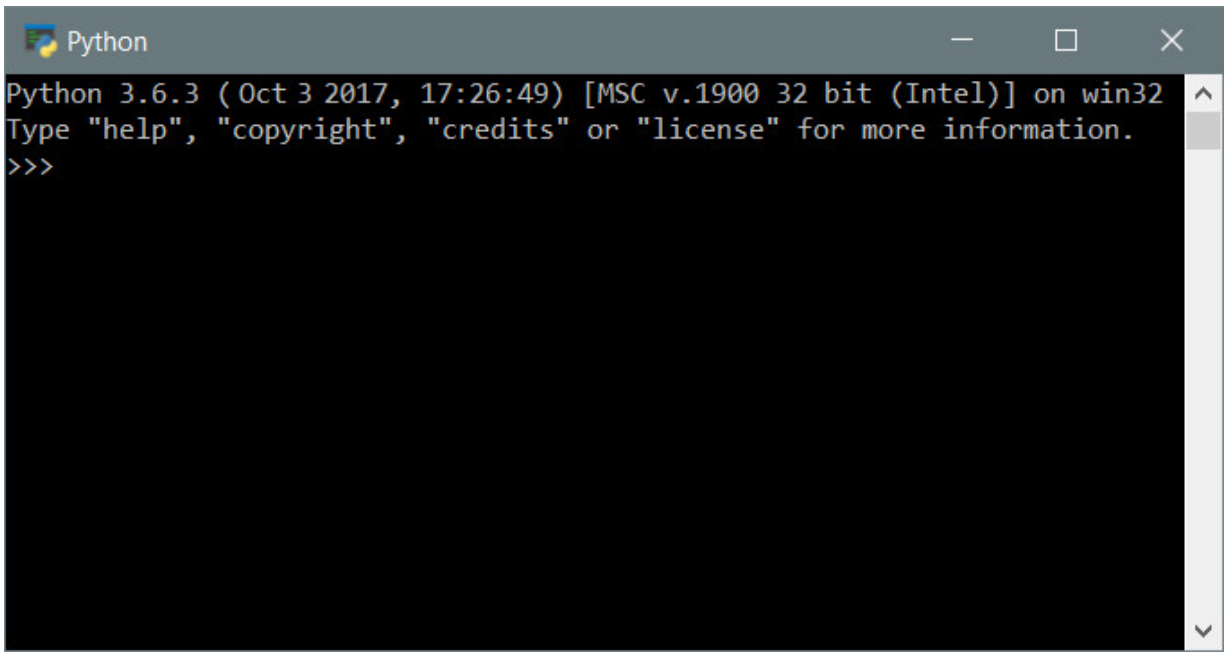
- Dazu klickst du mit der rechten Maustaste auf das Symbol für Python (PYTHON.EXE). Im Kontextmenü

wählst du KOPIEREN.

- Dann klicke auf eine freie Stelle auf dem Desktop, ebenfalls mit der rechten Maustaste. Im Kontextmenü wählst du VERKNÜPFUNG EINFÜGEN.
- Es ist sinnvoll, für das neue Symbol auf dem Desktop den Text `python.exe – Verknüpfung` einfach durch `Python` zu ersetzen.

Von nun an kannst du auf das neue Symbol **doppelklicken**, um die Arbeitsumgebung von Python direkt zu starten.

Was dich nach dem Start erwartet, sieht etwa so aus:



Die ersten beiden Zeilen informieren dich unter anderem über die aktuelle Python-Version, aber du bekommst auch

schon ein paar Befehle vorgeschlagen, die du hinter den drei spitzen Klammern (`>>>`) eintippen kannst.

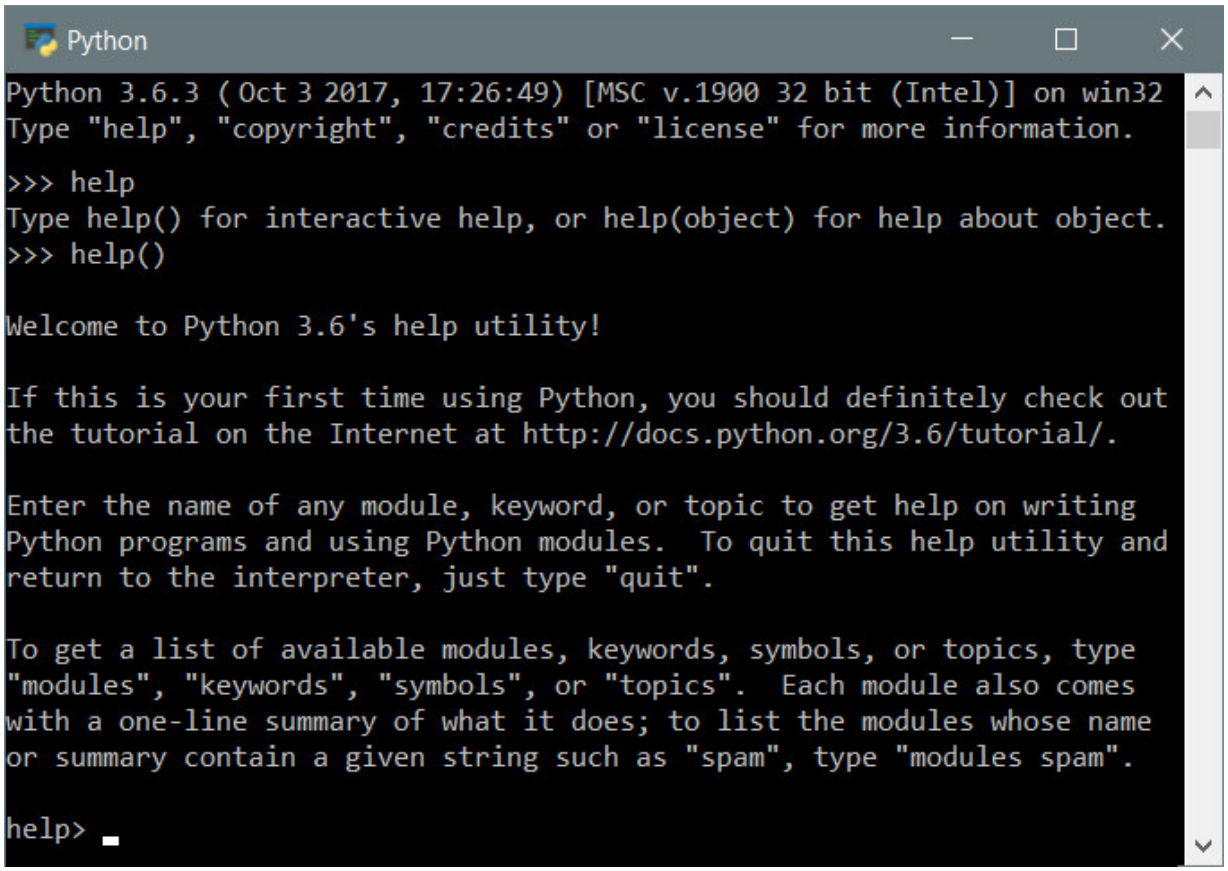
Die drei Zeichen werden hier auch **Prompt** genannt. Das ist eine Art Eingabeaufforderung, weil du dahinter etwas eingeben kannst (und musst, wenn es weitergehen soll).

- Probieren wir es doch gleich einmal mit »help«. Tippe dieses Wort ein.

Und prompt gibt es etwas zu meckern: Na ja, es ist eher ein netter Hinweis: Man muss `help` mit zwei runden Klammern dahinter eintippen.

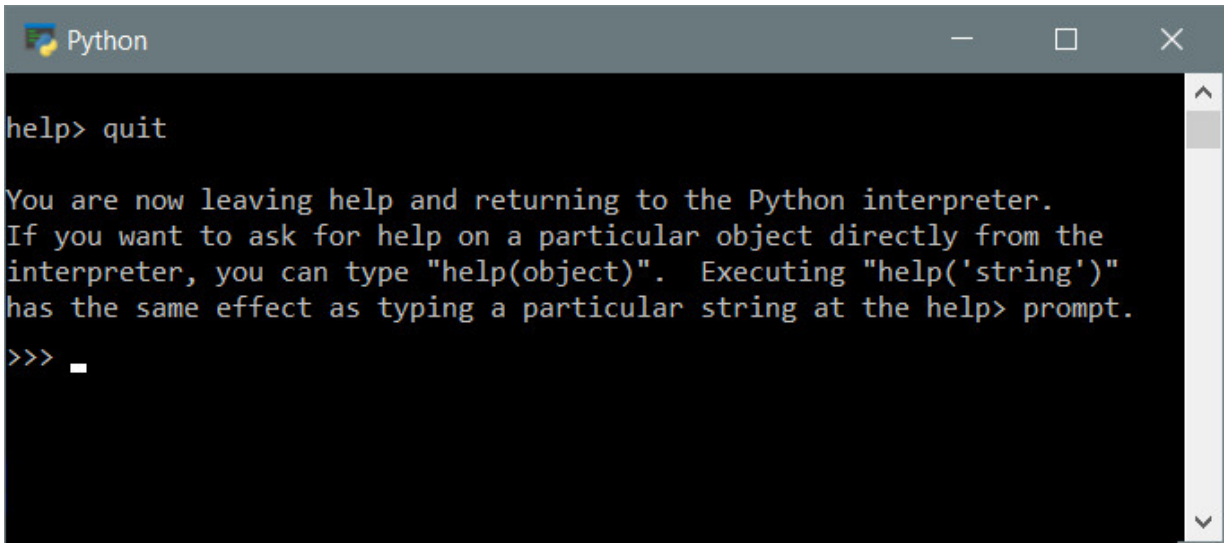
- Gib also `help()` ein.

Und du bekommst gleich eine ganze Menge Text serviert:



Nun steht da `help>` als Prompt. Du kannst dahinter ein Wort eingeben, und wenn es zum Python-Wortschatz gehört, bekommst du dazu eine (kurze) Erläuterung.

- Um zum ursprünglichen Prompt zurückzukehren, tippe `quit` ein.



Und du bist wieder zurück im Python-Interpreter.

Was ist ein **Interpreter**? Zuerst solltest du wissen, dass das, was du als Befehl hinter dem Prompt eintippst, für den Computer erst einmal völlig unverständlich ist. Normalerweise kann er also den jeweiligen Befehl gar nicht ausführen.

Ein Interpreter übersetzt die Befehlszeile in eine Sprache, die der Computer versteht, sodass er den Befehl ausführen kann – genannt **Maschinensprache**. Bei einem Programm, das aus einigen bis sehr vielen Zeilen bestehen kann, wird von einem Interpreter jede Zeile **einzeln** übersetzt und dann ausgeführt.

Im Gegensatz dazu gibt es **Compiler**, die das **gesamte** Programm in Maschinensprache übersetzen. Erst wenn das Programm komplett und fehlerfrei ist, kann es vom Computer ausgeführt werden. Für Python benutzen wir hier einen Interpreter, es gibt aber auch Python-Compiler.