

1 Python - neue Programmiersprache?

Ein Python-Programm ist eine einfache Textdatei, welche mit einem beliebigen Texteditor bearbeitet werden kann.

Das Ausführen des Programms bedeutet einfach, dass Python die Anweisungen in der Textdatei Zeile für Zeile durcharbeitet, bis es am Ende der Datei angelangt ist.

Das erste Beispielprogramm, welches üblicherweise in einer Programmiersprache verfasst wird, ist ein so genanntes „Hello World“-Programm, welches nichts anderes macht, als den Text „Hello World“ auf dem Bildschirm auszugeben.

In Python ist ein solches Programm im Vergleich zu anderen Programmiersprachen sehr einfach umzusetzen. Es besteht nur aus einer Zeile:

```
>>> print("Hello World!")
```

1.1 Editoren und IDEs

Wie oben erwähnt reicht zum Programmieren ein einfacher Texteditor. Im Gegensatz zu Textverarbeitungsprogrammen unterstützt reiner Text keine Formatierungen wie Schriftgrößen oder Schriftarten. Alle Farben, welche du im Texteditor siehst, werden vom Editor erstellt und sind nicht in der Datei selbst gespeichert. Sie dienen nur der Lesbarkeit vom Programmcode und werden vom Editor abhängig von der benutzten Programmiersprache festgelegt.

Texteditoren wurden schon in den Anfängen der Computerprogrammierung benutzt, gewisse aus dieser Zeit werden heute noch in weiterentwickelter Form benutzt, unter ihnen zum Beispiel Emacs und Vim. Wer einen moderneren Editor benutzen möchte, installiert zum Beispiel Atom.


Jedoch werden oft sogenannte IDEs (Integrated Development Environment) zum schreiben von Programmen benutzt. Diese besitzen viele hilfreiche Funktionen, welche ein Texteditor nicht hat, welche einem beim Programmieren unterstützen:

- Das Programm kann direkt im IDE ausgeführt werden und muss nicht separat aufgerufen werden.
- Die Befehle werden automatisch vervollständigt, so dass man nicht die ganzen Befehle tippen muss.
- Gewisse Fehler im Programm werden schon während dem programmieren erkannt und als solche gekennzeichnet.

Wir werden mit IDLE, einer sehr einfachen Python-IDE, welche schon bei einer normalen Python-Installation dabei ist, arbeiten. Etwas besser unterstützt wirst du von der speziell für Python entwickelten IDE PyCharm (in der Community Edition). Diese kannst du selbstständig in deiner Ubuntu VM installieren.

PyCharm lässt dich nicht nur Python-Programme editieren, sondern auch mehrere Dateien zu einem Projekt zusammenfassen. Dadurch werden auch komplexe Programme leichter umsetzbar.

Erstelle aus diesem Grund beim ersten Starten von PyCharm ein Projekt. Achte darauf, dass du einen Speicherort innerhalb deiner eigenen Dateien wählst, etwa in einem Ordner mit dem Namen „Informatik“. Benenne das Projekt jeweils nach dem Kapitel, an welchem du gerade arbeitest. Innerhalb des Projekts kannst du für jede weitere Aufgabe eine neue Python-Datei anlegen, in dem du im Menu File -> New... anklickst und anschliessend New Python-File wählst.

	Name:	Klasse:	Datum:
	BS Scripte	Python - 1	BFS12 Imr

Die erste Aufgabe, in welcher du eine Programmdatei benötigst ist die Aufgabe 2. Nenne also deine erste Datei „Aufgabe 2“.

In der Konsole siehst du etwa den folgenden Text:

```
Python 3.6.1 (default, Jun 25 2017, 14:04:36)
[GCC 7.1.0] on linux
Type "help", "copyright", "credits" or "license" for more
information.
>>>
```

Dies ist der Python Befehlsprompt, hier kannst du Befehle nicht als Programm speichern, du kannst sie nur einzeln hintereinander ausprobieren. In diesem Skript werden wir in Aufgaben oft kurze Beispiele im Befehlsprompt ausprobieren. Sie sind immer durch »> markiert und haben keine Zeilennummern wie andere Programme. Du könntest also zum Beispiel

```
>>> print("Hallo Welt!")
```

oder

```
>>> 2 + 3
```

eingeben. Sowohl PyCharm als auch IDLE haben immer nur einen Befehlsprompt. Du kannst also keine zweite Konsole öffnen.

Es macht aber oft keinen Sinn, die Befehle für ein Programm jedes mal am Befehlsprompt von neuem einzutippen. Für diesen Fall benutzen wir die oben erstellte, neue Python-Datei. Dort kannst du mehrere Befehle hintereinander aufschreiben um sie dann der Reihe nach ausführen zu lassen. So kannst du das Programm später auch wieder öffnen, verändern oder erneut ausführen.

Um die Befehle auszuführen, kannst du in PyCharm auf das grüne Dreieck neben der ersten Zeile des Programms klicken oder die Shift und F10 Tasten zusammen drücken. In IDLE kann das Programm mit der F5 Taste ausgeführt werden.

1.2 Eingabe und Ausgabe

Computer arbeiten grundsätzlich nach dem EVA-Prinzip. Die Abkürzung EVA steht für **E**ingabe, **V**erarbeitung und **A**usgabe. Damit ist gemeint, dass wir dem Computer zu Beginn über eine Tastatur, eine Maus, einen Touchbildschirm oder über andere Eingabegeräte Daten oder Befehle zur Verarbeitung liefern.

Anschließend verarbeitet der Computer unsere Daten gemäß den erteilten Befehlen und gibt uns die berechnete Antwort über ein Ausgabegerät zurück. Dies kann ein Bildschirm, ein Drucker oder ein anderes, spezialisiertes Gerät zur Darstellung von Daten sein.

Wir werden uns in den folgenden Kapiteln mit der einfachsten Art der Ein- und Ausgabe begnügen, in dem wir den Programmen vorerst nur Text übergeben und sie uns auch wieder Text zurückgeben. Wir haben schon gesehen, dass wir mit

```
>>> print("Hallo Welt")
```

Text ausgeben können. Umgekehrt können wir mit der Funktion input() Text einlesen. Dabei kannst du in der Klammer eine Frage an den Benutzer eingeben. Die folgende Zeile führt dazu, dass Text eingeben kannst. Der Computer fordert dabei den Benutzer zunächst auf „Gib bitte etwas Text ein: „ und wartet anschließend auf die Eingabe, welche mit Drücken der Enter-Taste abgeschlossen wird.

```
>>> input("Gib bitte etwas Text ein: ")
```

Du kannst so gewissermaßen einen eigenen Befehlsprompt programmieren.

1.3 Aufgaben

1. Probiere die folgenden Befehle in der Python-Konsole aus und notiere dir, wozu sie benutzt werden.

```
>>> 1 + 2 * 4
>>> print("Hello World!")
>>> sorted( [4,3,2,5,10,9] )
>>> sum( [4,3,2,5,10,9] )
>>> (4 + 3) / 5
>>> (4 + 3) // 5
```

2. Erstelle, falls nicht schon erledigt, wie oben erklärt, eine neue Python-Datei.
 - (a) Schreibe in dieser Datei die nötigen Befehle für ein „Hello World“-Programm. Falls du nicht weiter weißt, kannst du noch einmal den Anfang dieses Kapitels lesen.
 - (b) Das Programm kannst du mit der F5-Taste ausführen und testen. Wo wird der Text „Hello World“ ausgegeben?
 - (c) Passe das Programm so an, dass es den Benutzer zuerst nach einer Eingabe fragt und anschließend mit „Hello World“ antwortet.
3. Speichere das folgende Programm in einer Python-Datei namens gui-beispiel.py.
 - (a) Was macht das Programm?
 - (b) Welche Zeilen kannst du aus dem Programm löschen, ohne dass sich seine Funktionsweise ändert oder es nicht mehr funktioniert?

```
1  # Tkinter ist ein Paket zum erstellen von GUIs
2  from tkinter import *
3
4  # Funktion, welche beim Klick des Buttons
5  # ausgeführt wird.
6  def action():
7      print("Aua!")
8
9  # Ein Fenster erstellen
10 fenster = Tk()
11
12 # Einen Button erstellen
13 button = Button(fenster, text="Hit me!", command=action)
14 button.pack()
15
16 mainloop()
```