ZUSAMMENFASSUNG Module in Python

EINFÜHRUNG

Für eine bessere Übersicht, vor allem bei komplexem Code, kann man ein Programm auf mehrere Files aufteilen. Das eignet sich vor allem, wenn man ein großes Problem in mehrere Teilprobleme aufteilen kann, die man dann einzeln programmiert.

Dadurch kann man zum einen Aufgaben innerhalb eines Teams besser aufteilen. Einzelne Module können so unabhängig voneinander entwickelt werden. Später können die Module dann ins Hauptprogramm geladen werden, um dort auf die ganzen Funktionalitäten zuzugreifen.
Außerdem kann man dadurch einzelne Module sehr gut wiederverwenden, wenn man sie öfter verwenden muss.
Zudem können Fehler einfacher gefunden werden und Abhängigkeiten werden verringert. Denn jedes Modul bildet eine abgeschlossene Einheit und kann separat getestet werden.

MODULE PROGRAMMIEREN UND EINBINDNGEN

Wichtig ist, dass die Module die man erstellt, auch Python files sind, also mit .py enden.

Ein programmiertes Modul bindet man in die Haupt file ein, indem man eine import-Anweisung schreibt.
Zum Beispiel:

```
import(calc)
import (random)
```

oder aber auch: import (calc, random)

Man kann dabei so viele Module importieren wir man möchte.
Wenn ein Modul erfolgreich geladen wurde, hat man einen neuen Namensraum erzeugt, über den man jetzt alle Variablen, Funktionen und Klassen des Moduls ansprechen kann. Das geht, indem man zuerst den Modulnamen schreibt, gefolgt von einem Punkt und der gewünschten Funktion/Variable/Klasse
Zum Beispiel:

```
print(calc.addition(5,10))
```

Das liegt daran, dass die Funktion über den jeweiligen Namensraum aufgerufen werden muss.

Name des Modulraum ändern

Man kann zudem durch die import Anweisung den Namen des Modulraum ändern.

Zum Beispiel:
import calc as calculation

Somit hat man den Namen des Moduls calc in calculation umgewandelt.
Natürlich muss dann bei einem Funktionsaufruf auch der neue Name verwendet werden.

Modul ohne Namensraum

Falls man die Funktionen eines Moduls einfach mit dem normalen Namen aufrufen möchte, kann man für Module auch keinen Namensraum erzeugen.
Das funktioniert so:

import calc import *

Hier nimmt man den Namen calc vor dem Funktionsaufruf weg und ruft die Funktion so auf, als wäre sie innerhalb dieses files definiert worden. Diese Schreibweise sollte man aber besser vermeiden!

Eine weitere Möglichkeit ist, statt dem Stern den Namen der gewünschten Funktion hinzuschreiben. Also:
from calc import faculty

Bei mehreren Funktionen:
from calc import faculty, addition, …

DAS BUILT-IN MODUL MATH

Python besitzt zahlreiche built-in Module. Diese sind in der Programmiersprache C geschrieben und werden in den Python Interpreter integriert.

Das math Modul enthält viele Funktionen, die man in der Mathematik braucht. In der Python Dokumentation findet man einen Überblick über alle Funktionen des Moduls:
https://docs.python.org/3/library/math.html

Eine Übersicht über ALLE built-in Module in Python findest du unter:
(https://docs.python.org/3/py-modindex.html

DAS BUILT-IN MODUL MATH

Ein wichtiges built-in Modul ist das random Modul. Mit diesem kann man sich beispielsweise Dummy Daten generieren lassen um Dinge zu testen, oder es kann in der Spieleprogrammierung genutzt werden.
Es wird eingebunden über den Befehl import random.

random()
Mit der Funktion random() kann man einen Zufallswert vom Typ float() zwischen 0 und 1 ausgeben lassen. 0 ist dabei inkludiert und 1 nicht.

```
test = random.random()
```

uniform()
Hier können wir zwei Werte angeben die Den Bereich beschreiben, zwischen dem eine Zufallszahl ausgegeben werden soll.

```
test = random.uniform(1, 10)
print(test)
```

randint()
Mit dieser Funktion kann mein eine Zahl vom Typ integer erstellen lassen.
Das funktioniert wieder mit zwei Werten die man übergibt:

```
test = random.randint(2,7)
```

Hier sind diesmal der Start- und Endwert inkludiert.

choice()
Diese Funktion ist nützlich, wenn man aus einer Liste oder einem anderen sequentiellen Datentyp einen zufälligen Wert ausgeben möchte.
Zum Beispiel:

```
winner = random.choice(lottery)
```

choices()
Diese Funktion hat das gleiche Prinzip wie oben, nur hier kann man sich mehrere Werte ausgeben lassen. Wie viele Werte das sind, muss man vorher festlegen über den Parameter k.

```
winner = random.choices(lottery, k=3)
```

WICHTIG Es kann auch sein, dass ein Wert mehrmals gezogen wird!

Eine weitere Besonderheit ist die Gewichtung. Denn mit choices() kannst du auch angeben welche Gewichtung die einzelnen Werte deiner Liste haben.
Zum Beispiel:

```
winner = random.choices(lottery, weights=[1, 2, 3, 4, 5], k=3)
```

sample()
Diese Funktion hat das gleiche Prinzip wie choices(). Allerdings kann hier jeder Wert maximal einmal gezogen werden.

shuffle()

Mit dieser Funktion kann man eine übergebene Liste mischen. Die Reihenfolge der Werte verändert sich dadurch.

MODULE SIND AUSFÜHRBARE DATEIEN

Die __main__file ist genauso eine Python file wie calc. Daher ist es möglich, dass wir nicht nur main.py sondern auch calc.py direkt ausführen.

Man kann also main.py auch als Modul verwenden und beispielsweise in calc.py über eine import-Anweisung einbinden. Dadurch kann der Inhalt einer Variable ausgeben werden, die im main.py definiert ist.

Modulreferenz __name__

Diese referenziert auf einen String, der den Namen des Moduls enthält. Dadurch referenziert man auf den String __main__.

Über die if-Abfrage
If__name__ == „__main__“:
Kann man bestimmen, dass das jeweilige Programm als Hauptprogramm ausgeführt wird, wenn der Name main ist.