Lektion 2 : Bedingte Anweisung und Schleifen

Anwendung : Schleifen und Tests

Für diese Anwendung von Lektion 2 werden die in den bisherigen Lektionen gezeigten Konzepte in Bezug auf bedingte Anweisungen sowie begrenzte und unbegrenzte Schleifen angewendet.

Lernziele :

- Anwendung der **while** – und **for** – Schleife im Bereich Stochastik

---

In dieser Anwendung wird ein Skript vorgestellt, das Folgendes ermöglicht:

- **Simulation eines Würfels** durch eine **Wurffunktion**
- Diese Funktion wird dann in einem anderen Skript verwendet, um die Anzahl der Würfe zu bestimmen, die erforderlich sind, um beim Würfeln von 2 idealen Würfeln eine **Summe von 12** zu erhalten.
- Für den einzelnen Würfel sollen dann die **absoluten Häufigkeiten** für das Auftreten einer Zahl ermittelt werden, um sie mit den Wahrscheinlichkeiten vergleichen zu können.

Die Simulation des Würfelwurfes

- Um mit Zufallszahlen arbeiten zu können, muss das **Zufallsmodul** geladen werden, **bevor** die Funktion definiert wird.
- Erstellen Sie ein neues Skript und nennen Sie es „**WURF“**.
- Drücken Sie „**F1 (Fns…)**“ und dann „**Modul**“ und wählen Sie „**2: random**“
- Definieren Sie die Funktion **wurf()**, um eine ganzzahlige Zufallszahl zwischen 1 und 6 zu erhalten.
- „Würfeln“ Sie nun, indem Sie die Funktion in der Shell wiederholt durch ⍓ aufrufen.

Anzahl der erforderlichen Versuche.

Zwei ideale 6-seitige Würfel werden geworfen und die beiden erzielten Ergebnisse addiert. In einem weiteren Skript werden die Würfe dieser beiden Würfel modelliert und die Anzahl n der Würfe ermittelt, die erforderlich ist, bis die Summe von 12 erhalten wird.

- Es bietet sich die Wiederverwendung der vorherigen Funktion **wurf()** an.
- Die Variable ***s*** gibt die Summe der Würfe und ***n*** die Anzahl der Würfe an, die vor Erreichen von 12 erforderlich sind. Beide werden auf 0 initialisiert.
- In Python erhält man das Symbol „**≠**“ durch Drücken der Taste **“f2**“ oder über das Menü „**Fns ..**“ und dann „**Ops**“. Das Symbol selbst wird durch „**! =**“ dargestellt.
- Vervollständigen Sie das Skript und achten Sie darauf, die Einrückung zu berücksichtigen. Führen Sie es dann aus.
- Die erste Zahl gibt die Anzahl der Würfe an, die erforderlich sind, um die von der zweiten Zahl angezeigte Summe 12 zu erreichen.

EDITOR: ZWOELF
PROGRAM LINE 0011

```
from random import *
def wurf():
  d=randint(1,6)
  return d
def summe():
  s=0
  n=0
  while s!=12:
    s=wurf()+wurf()
    n=n+1
  return n,s_
```

Fns… | a A # | Tools | Run | Files

PYTHON SHELL

```
>>> # Running ZWOELF
>>> from ZWOELF import *
>>> summe()
(37, 12)
>>> summe()
(120, 12)
>>> summe()
(144, 12)
>>> summe()
(17, 12)
>>> |
```

Fns… | a A # | Tools | Editor | Files

Wie oft erscheinen die Zahlen beim Würfelwurf?

Das vorige Beispiel hat deutlich gemacht, dass die Anzahl der Versuche schwankt, die erforderlich sind, um eine 12 zu erhalten. Macht man nun eine sehr große Anzahl von Würfen (Tests), so sollte sich eine Gleichverteilung bei den einzelnen Wurfergebnissen einstellen.
Beim Skript werden Listen verwendet, die den Code erheblich verkürzen.

- Erstellen Sie ein neues Skript “**WURF2**“
- Es werden **n** Würfe durchgeführt.
- Das Ergebnis eines Wurfs wird in der Liste **l** gespeichert, die zuvor mit der Anweisung **l = []** leer initialisiert wurde.
- Liste **f** enthält die Zahlen auf dem Würfel.
- Liste **fl** enthält dann die absoluten Häufigkeiten. Auch diese Liste wird als zunächst leere Liste **fl=[]** angelegt.
- Die erste **for** – Schleife erstellt eine Würfelliste der Länge **n**.
- Die zweite **for** – Schleife zählt dann, wie oft die einzelnen Zahlen aus der Liste **f** in der Wurfliste **l** vorkommen.
- Lassen Sie das Programm für mehrere Werte von **n** laufen und beobachten Sie, ob sich eine (ungefähre) Gleichverteilung einstellt.

EDITOR: WURF2
PROGRAM LINE 0010

```
from random import *
f=[1,2,3,4,5,6]
def wurf(n):
  l=[]
  fl=[]
  for i in range(n):
    l.append(randint(1,6))
  for i in range(6):
    fl.append(l.count(f[i]))
  return fl
```

Fns… | a A # | Tools | Run | Files

PYTHON SHELL

```
>>> # Shell Reinitialized
>>> # Running WURF2
>>> from WURF2 import *
>>> wurf(10)
[0, 3, 1, 1, 4, 1]
>>> wurf(100)
[17, 11, 12, 23, 15, 22]
>>> wurf(1000)
[172, 167, 183, 169, 153, 156]
>>> |
```

Fns… | a A # | Tools | Editor | Files