Lektion 2 : Bedingte Anweisung und Schleifen — Übung 3 : Die while-Schleife

In dieser dritten Übung lernen Sie, wie man einen Prozess oder eine Gruppe von Befehlen wiederholen kann, indem man eine **while** - Schleife verwendet.

Lernziele :

- Syntax der **while** - Schleife
- Anwendung der **while** – Schleife in einfachen Beispielen

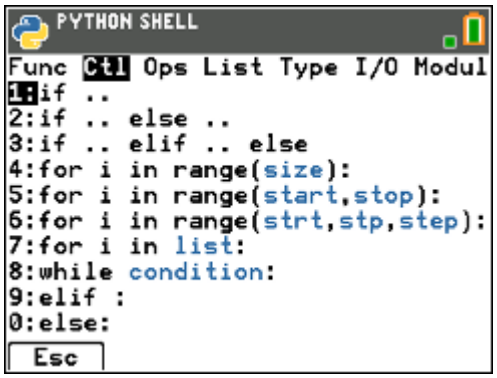
In einem Programm ist es manchmal nützlich, eine oder mehrere Anweisungen mehrmals wiederholen zu lassen, ohne dass die Anzahl der Wiederholungen bekannt ist. In diesem Fall verwendet man eine **while**-Schleife.
Die Schleife wird mehrmals durchlaufen, bis eine **Bedingung** nicht mehr erfüllt ist. Solange diese Bedingung erfüllt ist, wird die Schleife fortgesetzt.
Also **umgangssprachlich**

Solange die Bedingung erfüllt ist, tue folgendes ……

In **Python** :

```
while condition :
    Instructions
```

Es gibt keinen Befehl, der das Ende der Schleife anzeigt. Die zur Schleife gehörenden Befehle werden wieder durch die Einrückung um 2 Stellen nach rechts zusammen gehalten.

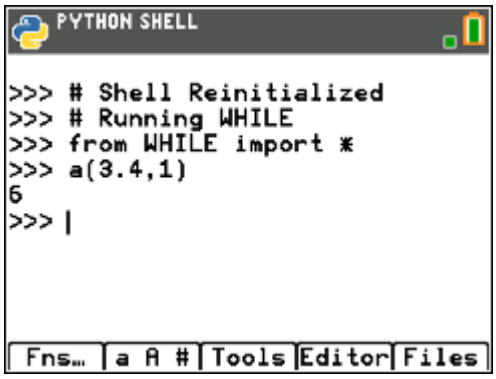


Ein einfaches Beispiel

Für die Folge $c_{n+1} = 0{,}8\ c_n$ mit $c_0 = 3{,}4$ ist der kleinste Index n gesucht, so dass $c_n < 1$.

- Legen Sie ein neues Skript mit dem Namen « **WHILE** » an.
- Die Aufgabe wird ein wenig erweitert, denn die Funktion erfordert zwei Eingaben : *c0* und die jetzt variable Grenze *g* statt der festen Grenze *1*.
- Startwert ist *n=0*. Bei jedem Schleifendurchlauf wird *n* um *1* vergrößert, solange bis die Bedingung nicht mehr erfüllt ist.
- Das nebenstehende Bild zeigt das Programm, das folgende Bild das Ergebnis eines Durchlaufs.

n←0
c←3.4
Solange wie $c \geq 1$
  n←n+1
  c←0.8c
Ende



Eine Anwendung : Der springende Ball

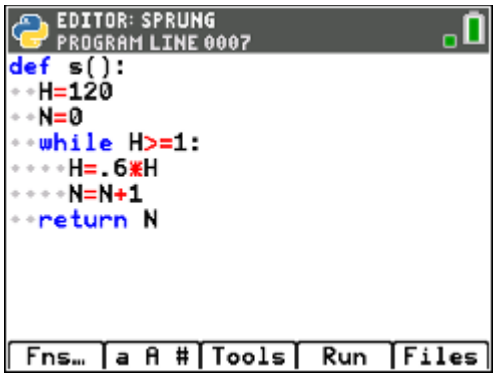
Ein Ball wird aus einer Höhe von 1,20 m fallen gelassen. Er springt wieder hoch und erreicht dabei genau 60% seiner ursprünglichen Höhe.

Wie oft springt der Ball, bis die Sprunghöhe unter 1 cm bleibt ?

Das **Programm** :

H ← 120
N ←0
Solange wie $H \geq 1$
  H← $0.6 \times H$
  N←N+1
Ende

- Legen Sie ein neues Skript mit dem Namen « **SPRUNG** » an.
- Benennen Sie die Funktion als **s()** (also ohne Variable !).
- Schreiben Sie das abgebildete Programm
- Lassen Sie es mit « **Run** » laufen. Es entsteht das untere Bild.
- Man kann das Skript auch noch mit einem Text ergänzen, etwa
  return(« Anzahl der Sprünge : « ,N)
  der dann mit angezeigt wird.



Nur zum Nachdenken :

Wie viele Sprünge macht der Ball, bis er zur Ruhe kommt ? Kann man das überhaupt berechnen ?