4. Wiederholungen

4.1. Grundlagen

1. **Schleifen**

   Die zweite wichtige Programmierstruktur ist die Wiederholungsschleife. Dabei wird eine gewisse Folge von Befehlen ausgeführt, so lange eine Bedingung erfüllt ist.
   Man unterscheidet (aus informatik-theoretischer Sicht) drei Arten von Wiederholungsschleifen, wobei eigentlich nur eine einzige Art nötig ist. Die andern beiden Arten kann man als Spezialfälle dieser ersten Art betrachten.

2. **Beispiel**

   Schreibe ein Programm, welches den Notendurchschnitt einiger einzugebender Zahlen berechnet. Dabei soll die Eingabe wiederholt werden, so lange man sinnvolle Werte eingibt. Die Wiederholung endet also, wenn man eine Zahl kleiner als 1 oder grösser als 6 eingegeben hat.
   Das folgende Programm illustriert die Struktur:

```
# Notendurchschnitt
summe = 0  #Summe der bisher eingegebenen Noten
anznoten = 0  #Anzahl der eingegebenen Noten
eing = float(input("Erste Note? "))
while (eing >= 1) and (eing <= 6):
    summe = summe + eing
    anznoten = anznoten + 1
    eing = float(input("Nächste Note? ")):
schnitt = summe/anznoten
print("Der Durchschnitt beträgt ",schnitt," .")
```

3. **Bemerkungen**

   a) *Vor* der Wiederholungsschleife sollten normalerweise alle verwendeten Variablen definiert sein.

   b) Nach dem `while` kommt die Bedingung, welche getestet wird. Die Bedingung kann in Klammern stehen. Der Befehl wird immer mit einem Doppelpunkt abgeschlossen.

   c) Nach dem Doppelpunkt folgt die Einrückung automatisch.

   d) Solange die Bedingung erfüllt ist, dann werden alle Befehle ausgeführt, die eingerückt wurden.

   e) Die letzten beiden Zeilen sind nicht mehr eingerückt und werden erst bearbeitet, wenn die Wiederholungsschleife verlassen wurde, d.h. wenn zum ersten Mal die Bedingung *nicht* erfüllt ist.

   f) Die Berechnung des Durchschnitts erfolgt, nachdem alle Noten eingegeben wurden.

4. **Die for-Schleife**

   Rein theoretisch gesehen ist die **while**-Schleife für alle Formen von Wiederholungen ausreichend. Für gewisse Situationen kann man auch eine andere Struktur verwenden, nämlich die so genannte **for**-Schleife.
   Der Befehl lautet:
   `for i in range(n):`
   Alles nachfolgende, was innerhalb der Schleife liegen soll, wird wieder eingerückt. Die Struktur ist also praktisch dieselbe, aber bei der for-Schleife weiss man, dass man *genau* n Durchläufe hat. Die Schleife wird also genau n mal wiederholt, wobei für die erste Wiederholung $i = 0$, für die zweite Wiederholung $i = 1$, usw., und für die letzte Wiederholung $i = n - 1$ ist.
   Das folgende Beispiel illustriert eine gute Anwendung der for-Schleife.

5. **Immobilie**

   Der Wert einer Immobilie steigt jedes Jahr um p % an.
   Schreibe ein Programm, welche den Zeitwert dieser Immobilie für die nächsten 10 Jahre berechnet.
   Eingabewerte sind der heutige Wert und der Prozentsatz p.

```
# Immobilie
zeitwert=float(input("Wert heute? "))
p=float(input("Wertzunahme in %? "))
for i in range(10):
    zeitwert=zeitwert+zeitwert*p/100
    jahr=i+1
    print("Wert im Jahr",jahr,": ",zeitwert)
```

   Weil man weiss, dass die Schleife genau 10 Mal wiederholt wird, ist eine for-Schleife empfehlenswert. Weil der Index i von 0 bis 9 geht, ist das Jahr einfach $i + 1$.
   Untenstehend ist das gleiche Programm (zum Vergleichen) auch noch mit einer while-Schleife programmiert.

```
# Immobilie
zeitwert=float(input("Wert heute? "))
p=float(input("Wertzunahme in %? "))
jahr=1
while jahr<=10:
    zeitwert=zeitwert+zeitwert*p/100
    print("Wert im Jahr",jahr,": ",zeitwert)
    jahr=jahr+1
```