

## Listen

hilft bei

Aufgabe 1, 3, 5 und 6

verwendet

| Was                  | Beispiel                   | Ergebnis |
|----------------------|----------------------------|----------|
| anlegen              | werte = [12.4, 15.1, 18.9] |          |
| einzelnen Wert lesen | werte[0]                   | 12.4     |
| letzten Wert lesen   | werte[-1]                  | 18.9     |
| Anzahl               | len(werte)                 | 3        |
| anhängen             | werte.append(21.8)         |          |
| zusammenzählen       | sum(werte)                 | 46.4     |
| größter, kleinster   | max(werte), min(werte)     |          |

Gezählt wird ab null: `werte[0]` ist der erste Wert.

## Verzweigen

hilft bei

Aufgabe 2, 5 und 6

verwendet

So ist eine Verzweigung gebaut:

```
if <Bedingung>:  
    <if-Rumpf>  
elif <Bedingung>:  
    <elif-Rumpf>  
else:  
    <else-Rumpf>
```

`elif` und `else` sind freiwillig. Doppelpunkt und Einrückung sind es nicht.

## Wiederholen mit while

hilft bei

Aufgabe 3 und 5

verwendet

```
zaehler = 1  
while zaehler <= 5:  
    <Rumpf>  
    zaehler = zaehler + 1
```

Drei Stellen müssen zusammenpassen: **Startwert** davor, **Bedingung** in der `while`-Zeile, **Erhöhen** im Rumpf.

Fehlt das Erhöhen, wird die Bedingung nie falsch – die Schleife läuft endlos. Anhalten mit dem roten **Stop**-Knopf.

`continue` springt zum nächsten Durchlauf, `break` verlässt die Schleife ganz.

ID  
S021



## Listen

Hilfskärtchen • Workshoptag 2

ID  
S022



## Verzweigen

Hilfskärtchen • Workshoptag 2

ID  
S023



## Wiederholen mit `while`

Hilfskärtchen • Workshoptag 2

## Wiederholen mit for

hilft bei  
Aufgabe 4, 5 und 6

verwendet

```
for <Name> in <Liste>:  
    <Rumpf>
```

for, in und der Doppelpunkt stehen fest. Den **Namen** suchst du dir aus; er trägt bei jedem Durchlauf den nächsten Wert.

Der Rumpf läuft einmal je Wert – die **Liste** bestimmt also, wie oft. Mitzählen musst du nichts.

continue springt zum nächsten Durchlauf, break verlässt die Schleife ganz.

## Warten

hilft bei  
Aufgabe 3 und 4

verwendet

```
import time  
  
time.sleep(0.5)
```

import time steht einmal ganz oben im Programm, time.sleep überall dort, wo eine Pause hingehört.

Die Zahl sind Sekunden – 0.5 ist eine halbe.

## Wenn etwas schiefgeht

hilft bei  
allen Aufgaben

verwendet

| Meldung          | Meistens heißt das                   |
|------------------|--------------------------------------|
| IndentationError | Eintrückung fehlt oder passt nicht   |
| SyntaxError      | Doppelpunkt vergessen, Klammer offen |
| NameError        | Name vertippt oder noch nicht belegt |
| IndexError       | Die Liste hat so viele Werte nicht   |
| TypeError        | Text und Zahl gemischt               |

**Gar keine Meldung, und es hört nicht auf?** Endlosschleife – mit dem roten **Stop**-Knopf anhalten und prüfen, ob sich die Zählvariable wirklich ändert.

ID  
S024



## Wiederholen mit for

Hilfskärtchen • Workshoptag 2

ID  
S025



## Warten

Hilfskärtchen • Workshoptag 2

ID  
S026



## Wenn etwas schiefgeht

Hilfskärtchen • Workshoptag 2

## Ablaufplan • die Symbole

hilft bei  
Gruppenpuzzle • Plakat

verwendet

Start

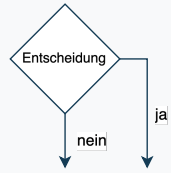
Operation

Ausgabe  
Variable/Wert

Eingabe  
Variable/Wert

Unterprogramm

Ende





# Ablaufplan • die Symbole

Hilfskärtchen • Workshoptag 2