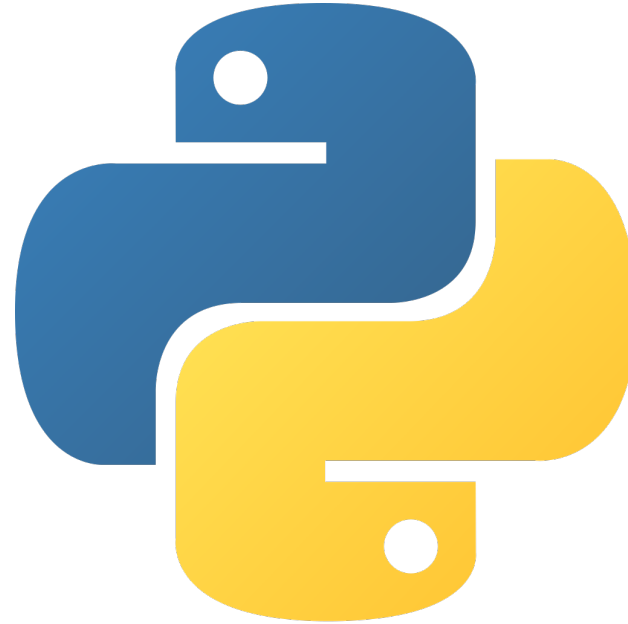
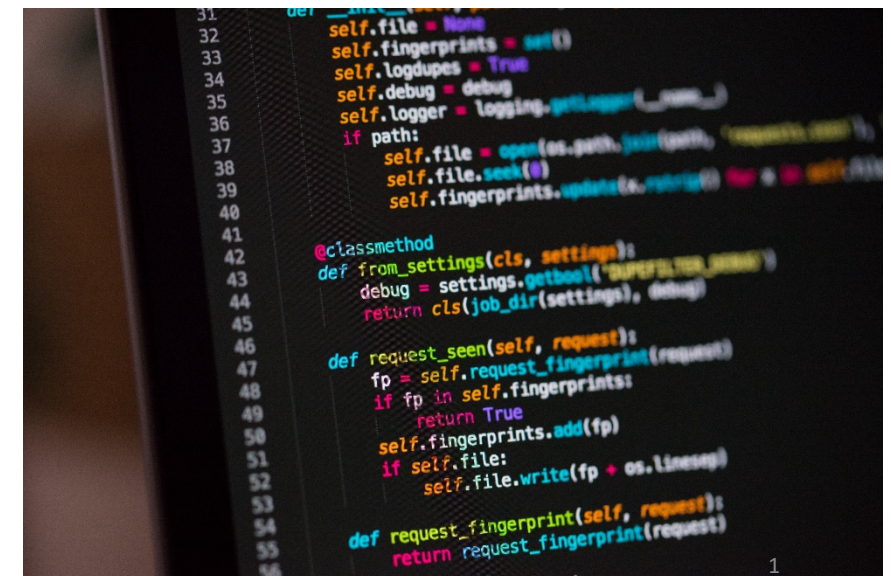


# Einführung in Python



Prof. Dr.-Ing. Prof. h.c. Detlef Jensen  
Dekan Fachbereich Technik

Tel.: +49 (0) 481 / 85 55 - 300  
mobil: +49(0)177 / 87 31 62 1  
Fax: +49 (0) 481 / 85 55 - 301  
Email: [jensen@fh-westkueste.de](mailto:jensen@fh-westkueste.de)



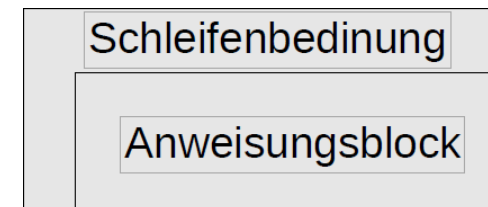
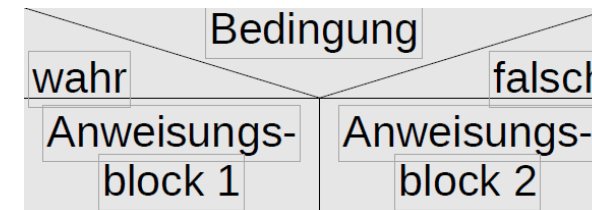
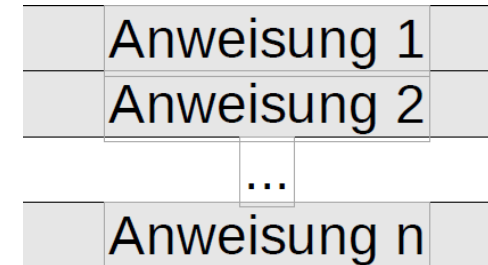


# Kontrollstrukturen

# Struktogramme

## Graphische Darstellung von Programmen

- **Anweisungen**
- **Bedingungen**
- **Schleifen**





# Bedingungen

# Bedingte Ausführung

- **Bisher hatten alle unsere Beispiele etwas gemeinsam.**
- **Der Computer hat die Anweisungen in der Reihenfolge abgearbeitet, in der wir sie eingegeben haben.**

```
"Erste Zeile"  
"Zweite Zeile"  
"Dritte Zeile"  
"Vierte Zeile"
```

# Bedingtes Ausführen

- Gewisse Dinge passieren aber nur unter Bedingungen. z.B. Wenn du zur dunklen Seite der Macht überläufst, bekommst du Kekse.
- In Python:

```
if you_come_to_the_dark_side:  
    you_get_cookies = True
```



- Wenn **you\_come\_to\_the\_dark\_side** nicht **True** ist, wird **you\_get\_cookies = True** nicht ausgeführt.

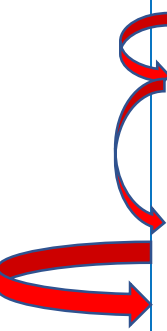
# If... then...

- **Blöcke** markieren den Bereich der bedingt ausgeführt wird:

```
"Vor dem Block,,  
if True:  
    "Wir sind im Block"  
"Nicht mehr im Block"
```

- Eingeleitet wird ein Block von einem :  
Der Block selbst ist mit vier Leerzeichen ( ) eingerückt.  
Er endet, wenn diese Einrückung endet.

# Blöcke



```
"Arbeit, Arbeit!"  
if you_train_me:                # False  
    you_get_a_solider  
    "Seid ihr der König?"  
if you_are_the_king:            # True  
    print("Also ich hab euch nicht gewählt.")
```

- Man nennt diese roten Pfeile den **Kontrollfluss**. Anweisungen, die den Kontrollfluss ändern, heißen **Kontrollstrukturen**.



# else...

- **Trifft eine Bedingung nicht zu, soll evtl. eine alternative Anweisung ausgeführt werden. Dafür gibt es das Schlüsselwort **else**:**

```
if you_come_to_the_dark_side:                                # False
    you_get_cookies
else:
    you_will_be_killed
```

- Gibt es mehrere Bedingungen, können wir **elif** verwenden:

```
x = 3
if x < 0:
    print("Negative Zahl")
elif x > 0:
    print("Positive Zahl")
else:
    print("Die Zahl ist Null")
    print("Block ist zuende")
```



# Zusammenfassung: Kontrollstrukturen

- **Blöcke (siehe Struktogramme) müssen im Code markiert werden**
- **In vielen Sprachen üblich: geschweifte Klammern**
- **Python verwendet stattdessen Einrückungen**
- **Einrücktiefe entscheidet über Blockzugehörigkeit**
- **Doppelpunkt leitet Block ein**

Fallunterscheidung: **if**

```
if x < y:
    print ("x is smaller ")
    print (x)
elif x == y:
    print (" both are equal ")
else :
    print ("y is smaller ")
    print (y)
```

# Wiederholtes Ausführen

- Stellen wir uns einmal vor, wir wollen die Zahlen von 0 bis 99 ausgeben:

```
print(0)  
print(1)  
print(2)  
print(3)  
# usw...  
print(99)
```

**Geht das nicht einfacher?**



# While-Schleife

- Einfacher geht es mithilfe einer while-Schleife:

```
x = 0  
while x < 100:  
    print(x)  
    x = x + 1
```



- Wurde der Block ausgeführt, wird die Bedingung wieder geprüft. Solange sie wahr ist, wird er erneut ausgeführt.

- **Es gibt nur zwei Schleifenarten: for und while. for ist eigentlich ein foreach und durchläuft ein beliebiges iterierbares Objekt elementweise. Möchte man einen Index durchlaufen, so kann dies mit der range-Funktion simuliert werden:**

```
for i in range(17, 42):  
    print i                # gibt die Zahlen 17 bis 41 aus
```

# range()

- Mit Hilfe der for-Schleife und der **range()** Funktion können wir das Zählen bis 100 noch eleganter gestalten:
- **range()** erzeugt eine Liste aller Zahlen in einem Bereich:
  - Syntax: **range(start, stop, step)**
    - **start**: Anfang der Liste (optional)
    - **stop**: Ende der Liste (nicht eingeschlossen!)
    - **step**: Schrittweite (optional)

# Kontrollstrukturen - Zählschleife

- **for** iteriert über die Elemente einer Sequenz.

```
for i in [1,2,3]:  
    print ("Zahl: ",i)
```

Zahl: 1

Zahl: 2

Zahl: 3

```
for i in " Hallo ":  
    print (" Buchstabe : " , i)
```

Buchstabe :

Buchstabe : H

Buchstabe : a

Buchstabe : l

Buchstabe : l

Buchstabe : o

Buchstabe :



# Kontrollstrukturen - Zählschleife

```
for i in (1 , 'a ' , [4 ,3 ,2] , " Hallo " ):  
    print (" Element : ", i)
```

Element : 1

Element : a

Element : [4, 3, 2]

Element : Hallo

- **continue** startet den nächsten Schleifendurchlauf (**for** und **while**).
- **break** bricht eine Schleife ab (**for** und **while**).

```
a = (" let ", " us ", " find ", " Bilbo ", " again ")
for word in a:
    if word == " Bilbo ":
        print " found Bilbo "
        break
print word
```

- Eine **while** Schleife iteriert, solange ihre Bedingung **True** ist.

```
a = 2048; exp = 0
while a > 1:
    a /= 2                # a = a / 2
    exp += 1             # exp = exp + 1
print a , "=", 2, "^" , exp
```