Funktionen

Auswerten eines
funktionalen Ausdrucks
ist das Grundkonzept
der
funktionalen Programmierung

Funktionen

- Funktionen der Mathematik sind die Vorlage
- Beispiel: *sin(30°) → 0,5*
  Funktionsname(Parameter) → Funktionswert
- Ein spezielles Problem von sinus:
  sin gehört nicht zum Standardvorrat, muss in Python importiert werden.

```
>>> from math import *
>>> sin(30)
-0.988031624092861 8
>>> sin(radians(30))
0.4999999999999994
```

Funktionen

- *Schachtelungen* von Funktionen sind möglich, werden durch Klammern gekennzeichnet und von innen nach außen ausgewertet
  $(3 – 5)^2 \rightarrow (-2)^2 \rightarrow 4$
- Schreibweise bei Python für Klammerausdrücke ist die gewohnte *Infix-Notation*
  quadrat (3 - 5) → quadrat (-2) → 4

Funktionen

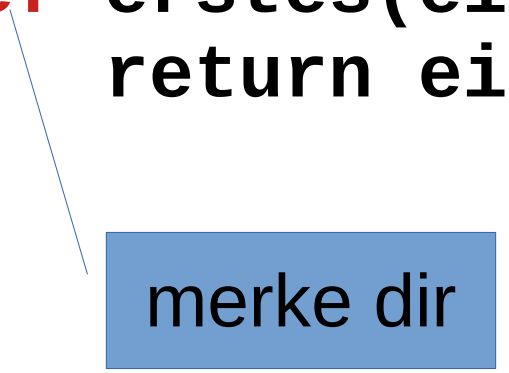
- Eine neue Funktion schreiben

```
def erstes(eineListe):
    return eineListe[0]
```

Funktionen

- Eine neue Funktion schreiben

```
def erstes(eineListe):
    return eineListe[0]
```



Funktionen

- Eine neue Funktion schreiben

unter dem Namen

```
def erstes(eineListe):
    return eineListe[0]
```

Funktionen

- Eine neue Funktion schreiben

verwende dabei

```
def erstes(eineListe):
    return eineListe[0]
```

Funktionen

- Eine neue Funktion schreiben

```
def erstes(eineListe):
    return eineListe[0]
```

Eine Funktion muss etwas zurückgeben

Hinweis: Da hinter **def** auch eine Prozedur stehen kann, fehlt eine Prüfung, so dass erst bei der Anwendung festgestellt wird, dass ggf eine Rückgabe fehlt.

Funktionen

- Eine neue Funktion schreiben

```
def erstes(eineListe):
    return eineListe[0]
```

was dafür auszuwerten ist

Funktionen

- Eine neue Funktion schreiben

```
def erstes(eineListe):
    return eineListe[0]
```

Funktionskopf

Funktionsrumpf

Funktionen

Achtung!

- Python macht call-by-reference
- Übergibt man einer Funktion ein Objekt (*also keine einfache Zahl*), wirken Veränderungen in der Funktion zurück auf die aufrufende Umgebung!
- Siehe nächste Folie

  (*Aber das geht natürlich viel einfacher!*)

Funktionen

```
meineListe=[1,2,3]
def gibEinzelnAus(liste):
    if liste==[]:
        return liste
    print(liste[0])
    liste.remove(liste[0])
    return gibEinzelnAus(liste)

print(gibEinzelnAus(meineListe))
print(meineListe)
meineListe=[1,2,3]
print(gibEinzelnAus([]+meineListe)) # echte Kopie
print(meineListe)
```

Funktionen

```
1
2
3
[]
[]
1
2
3
[]
[1, 2, 3]
```