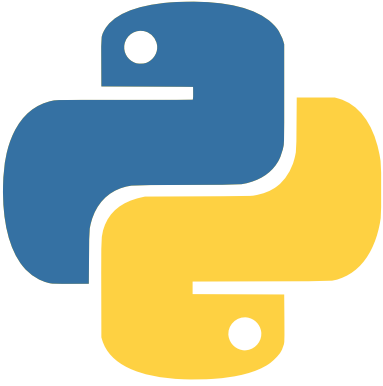
Universität Stuttgart
Projekt digit@L – BOOST. SKILLS. SUPPORT.



Programmierkurs Python

Benannte Funktionsargumente

Dominik Göddeke

IANS SimTech

Stiftung Innovation in der Hochschullehre

CC BY

Benannte Funktionsargumente

Benannte Funktionsargumente

- Vorgabe der **Zuordnung** zwischen Variable und Argument beim Aufruf einer Funktion

Benannte Funktionsargumente

- Vorgabe der **Zuordnung** zwischen Variable und Argument beim Aufruf einer Funktion
- **Benannte Argumente**: durch die Namen der Argumente vorgegeben

Benannte Funktionsargumente

- Vorgabe der **Zuordnung** zwischen Variable und Argument beim Aufruf einer Funktion
- **Benannte Argumente**: durch die Namen der Argumente vorgegeben
- Nicht durch ihre Position / Reihenfolge

Benannte Funktionsargumente

- Vorgabe der **Zuordnung** zwischen Variable und Argument beim Aufruf einer Funktion
- **Benannte Argumente**: durch die Namen der Argumente vorgegeben
- Nicht durch ihre Position / Reihenfolge
- Offenbar: Hilfreich bei Funktionen mit vielen (Default) Argumenten

Benannte Funktionsargumente

- Vorgabe der **Zuordnung** zwischen Variable und Argument beim Aufruf einer Funktion
- **Benannte Argumente**: durch die Namen der Argumente vorgegeben
- Nicht durch ihre Position / Reihenfolge
- Offenbar: Hilfreich bei Funktionen mit vielen (Default) Argumenten
- Starke **Erhöhung der Lesbarkeit** des Codes bei Funktionsaufrufen

Codebeispiel

Regeln für benannte Argumente

Regeln für benannte Argumente

- Alle **Pflichtargumente** müssen übergeben werden, egal ob per Position oder Name

Regeln für benannte Argumente

- Alle **Pflichtargumente** müssen übergeben werden, egal ob per Position oder Name
- **Reihenfolge** bei benannten Argumenten beliebig

Regeln für benannte Argumente

- Alle **Pflichtargumente** müssen übergeben werden, egal ob per Position oder Name
- **Reihenfolge** bei benannten Argumenten beliebig
- Mischformen sind möglich, konterkarieren Lesbarkeit

Regeln für benannte Argumente

- Alle **Pflichtargumente** müssen übergeben werden, egal ob per Position oder Name
- **Reihenfolge** bei benannten Argumenten beliebig
- Mischformen sind möglich, konterkarieren Lesbarkeit
- Optionale Argumente, d.h. Argumente mit Defaultwert, können weggelassen werden

Codebeispiele

Ein ausführlicheres Beispiel

Ein ausführlicheres Beispiel

- Erinnerung: **Vergleich zweier Gleitkomma-Zahlen** über

```
math.fabs(a-b) < eps*math.fabs(a)*math.fabs(b)
```

Ein ausführlicheres Beispiel

- Erinnerung: **Vergleich zweier Gleitkomma-Zahlen** über

```
math.fabs(a-b) < eps*math.fabs(a)*math.fabs(b)
```

- Offensichtlich Standardaufgabe

Ein ausführlicheres Beispiel

- Erinnerung: **Vergleich zweier Gleitkomma-Zahlen** über

```
math.fabs(a-b) < eps*math.fabs(a)*math.fabs(b)
```

- Offensichtlich Standardaufgabe
- Deshalb im Modul `math` bereitgestellt durch die Funktion `isclose()`

Ein ausführlicheres Beispiel

- Erinnerung: **Vergleich zweier Gleitkomma-Zahlen** über

  `math.fabs(a-b) < eps*math.fabs(a)*math.fabs(b)`

- Offensichtlich Standardaufgabe
- Deshalb im Modul `math` bereitgestellt durch die Funktion `isclose()`
- **Experiment**: verstehen wir nun die Python-Hilfe besser?

  `help(math.isclose)`

Ausgabe von help(math.isclose)

```
Help on built-in function isclose in module math:

isclose(a, b, *, rel_tol=1e-09, abs_tol=0.0)
    Determine whether two floating point numbers are close in value.

      rel_tol
        maximum difference for being considered "close", relative to the
        magnitude of the input values
      abs_tol
        maximum difference for being considered "close", regardless of the
        magnitude of the input values

    Return True if a is close in value to b, and False otherwise.

    For the values to be considered close, the difference between them
    must be smaller than at least one of the tolerances.

    -inf, inf and NaN behave similarly to the IEEE 754 Standard.  That
    is, NaN is not close to anything, even itself.  inf and -inf are
    only close to themselves.
```

Ein ausführlicheres Beispiel

- Zwei Pflichtargumente `a`, `b`

Ein ausführlicheres Beispiel

- Zwei Pflichtargumente `a`, `b`
- Zwei optionale Defaultargumente `rel_tol`, `abs_tol`

Ein ausführlicheres Beispiel

- Zwei Pflichtargumente `a`, `b`
- Zwei optionale Defaultargumente `rel_tol`, `abs_tol`
- Sternchen in der Argumentliste: vorerst ignoriert

Ein ausführlicheres Beispiel

- Zwei Pflichtargumente `a`, `b`
- Zwei optionale Defaultargumente `rel_tol`, `abs_tol`
- Sternchen in der Argumentliste: vorerst ignoriert
- Erinnerung Gleitkomma-Arithmetik: sinnvolle Defaultwerte sehr anwendungsabhängig

Ein ausführlicheres Beispiel

- Zwei Pflichtargumente `a`, `b`
- Zwei optionale Defaultargumente `rel_tol`, `abs_tol`
- Sternchen in der Argumentliste: vorerst ignoriert
- Erinnerung Gleitkomma-Arithmetik: sinnvolle Defaultwerte sehr anwendungsabhängig
- Verdeutlichung der Mächtigkeit optionaler Argumente für lesbaren Code

Ein ausführlicheres Beispiel

- Zwei Pflichtargumente `a`, `b`
- Zwei optionale Defaultargumente `rel_tol`, `abs_tol`
- Sternchen in der Argumentliste: vorerst ignoriert
- Erinnerung Gleitkomma-Arithmetik: sinnvolle Defaultwerte sehr anwendungsabhängig
- Verdeutlichung der Mächtigkeit optionaler Argumente für lesbaren Code
- Aber auch: Notwendigkeit, Funktionen vernünftig zu dokumentieren

Ein ausführlicheres Beispiel

- Zwei Pflichtargumente `a`, `b`
- Zwei optionale Defaultargumente `rel_tol`, `abs_tol`
- Sternchen in der Argumentliste: vorerst ignoriert
- Erinnerung Gleitkomma-Arithmetik: sinnvolle Defaultwerte sehr anwendungsabhängig
- Verdeutlichung der Mächtigkeit optionaler Argumente für lesbaren Code
- Aber auch: Notwendigkeit, Funktionen vernünftig zu dokumentieren
- Dokumentation: später mehr

Impressum, Danksagung und Quellen

Stiftung Innovation in der Hochschullehre

Gefördert durch die Stiftung Innovation in der Hochschullehre im Rahmen des Projekts digit@L, https://stiftung-hochschullehre.de

SimTech

Gefördert mit Mitteln der Deutschen Forschungsgemeinschaft (EXC 2075 - 390740016) im Rahmen der Exzellenzstrategie

---

Autor: Dominik Göddeke, IANS, Universität Stuttgart

Weitere Quellen:

- Logos Universität Stuttgart, IANS, SimTech: Universität Stuttgart, alle Rechte vorbehalten
- Logo Python: https://freesvg.org/387, CC-0
- Logo Stiftung: Stiftung Innovation in der Hochschullehre, alle Rechte vorbehalten
- Logo ZOERR: Universität Tübingen, alle Rechte vorbehalten

ZOERR

Veröffentlicht auf dem Zentralen OER Repositorium Baden-Württemberg, https://www.zoerr.de