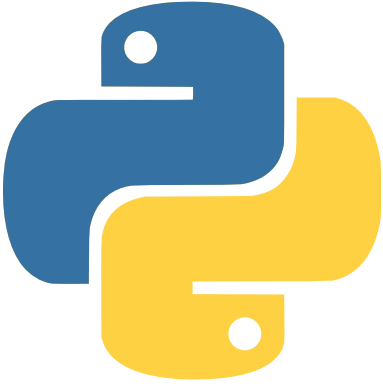
Universität Stuttgart
Projekt digit@L – BOOST. SKILLS. SUPPORT.



Programmierkurs Python

Elementares Rechnen

Dominik Göddeke

IANS SimTech

Stiftung Innovation in der Hochschullehre

Elementares Rechnen

Elementares Rechnen in Python

- Mit Zahlen können wir schon rechnen in Python

Elementares Rechnen in Python

- Mit Zahlen können wir schon rechnen in Python
  - `print(4+2)`

Elementares Rechnen in Python

- Mit Zahlen können wir schon rechnen in Python
  - `print(4+2)`
- Das funktioniert genau so mit Variablen

Elementares Rechnen in Python

- Mit Zahlen können wir schon rechnen in Python
  - `print(4+2)`
- Das funktioniert genau so mit Variablen
  - `a=4; b=2; print(a+b)`

Elementares Rechnen in Python

- Mit Zahlen können wir schon rechnen in Python
  - `print(4+2)`
- Das funktioniert genau so mit Variablen
  - `a=4; b=2; print(a+b)`
- **Wir schauen uns das in Jupyter an für die Grundrechenarten...**

Reihenfolge der Rechenoperationen

Reihenfolge der Rechenoperationen

- **Beispiel**: für $a = 3$ Auswertung von

$$\frac{5}{2} + 2 \cdot a^{\frac{4}{2}}$$

Reihenfolge der Rechenoperationen

- **Beispiel**: für $a = 3$ Auswertung von

$$\frac{5}{2} + 2 \cdot a^{\frac{4}{2}}$$

- Kurzes Kopfrechnen: $4/2 = 2$, also $3^2 = 9$ mal 2 gibt 18 plus 2.5 gibt 20.5

Reihenfolge der Rechenoperationen

- **Beispiel**: für $a = 3$ Auswertung von

$$\frac{5}{2} + 2 \cdot a^{\frac{4}{2}}$$

- Kurzes Kopfrechnen: $4/2 = 2$, also $3^2 = 9$ mal 2 gibt 18 plus 2.5 gibt 20.5
- Was macht Python? **Probieren Sie es gerne aus!**

Reihenfolge der Rechenoperationen

- **Beispiel**: für $a = 3$ Auswertung von

$$\frac{5}{2} + 2 \cdot a^{\frac{4}{2}}$$

- Kurzes Kopfrechnen: $4/2 = 2$, also $3^2 = 9$ mal 2 gibt 18 plus 2.5 gibt 20.5
- Was macht Python? **Probieren Sie es gerne aus!**
- Ohne Berücksichtigung der Reihenfolge: Ergebnis 83.5 :(

Reihenfolge der Rechenoperationen

- **Beispiel**: für $a = 3$ Auswertung von

$$\frac{5}{2} + 2 \cdot a^{\frac{4}{2}}$$

- Kurzes Kopfrechnen: $4/2 = 2$, also $3^2 = 9$ mal 2 gibt 18 plus 2.5 gibt 20.5
- Was macht Python? **Probieren Sie es gerne aus!**
- Ohne Berücksichtigung der Reihenfolge: Ergebnis 83.5 :(
  - Naive Implementierung `5/2+2*3**4/2`

Reihenfolge der Rechenoperationen

- Python berücksichtigt die üblichen Regeln aus der Schule

Reihenfolge der Rechenoperationen

- Python berücksichtigt die üblichen Regeln aus der Schule
- **„Punktrechnung vor Strichrechnung“**: Multiplikation/Division vor Addition/Subtraktion

Reihenfolge der Rechenoperationen

- Python berücksichtigt die üblichen Regeln aus der Schule
- **„Punktrechnung vor Strichrechnung“**: Multiplikation/Division vor Addition/Subtraktion
- **Potenzen vor allen anderen**

Reihenfolge der Rechenoperationen

- Python berücksichtigt die üblichen Regeln aus der Schule
- **„Punktrechnung vor Strichrechnung“**: Multiplikation/Division vor Addition/Subtraktion
- **Potenzen vor allen anderen**
- In der naiven Implementierung `5/2+2*3**4/2` also

$$3^4 = 81 \quad \rightarrow \quad 2 * 81/2 = 81 \quad \rightarrow \quad 81 + 2.5 = 83.5$$

Reihenfolge der Rechenoperationen

- Python berücksichtigt die üblichen Regeln aus der Schule
- **„Punktrechnung vor Strichrechnung“**: Multiplikation/Division vor Addition/Subtraktion
- **Potenzen vor allen anderen**
- In der naiven Implementierung `5/2+2*3**4/2` also

$$3^4 = 81 \quad \rightarrow \quad 2 * 81/2 = 81 \quad \rightarrow \quad 81 + 2.5 = 83.5$$

- **Lösung: Klammersetzung**

Reihenfolge der Rechenoperationen

- Python berücksichtigt die üblichen Regeln aus der Schule
- **„Punktrechnung vor Strichrechnung“**: Multiplikation/Division vor Addition/Subtraktion
- **Potenzen vor allen anderen**
- In der naiven Implementierung `5/2+2*3**4/2` also

$$3^4 = 81 \quad \rightarrow \quad 2 * 81/2 = 81 \quad \rightarrow \quad 81 + 2.5 = 83.5$$

- **Lösung: Klammersetzung**
- Klammern haben Priorität vor allen anderen Operatoren

Reihenfolge der Rechenoperationen

- Python berücksichtigt die üblichen Regeln aus der Schule
- **„Punktrechnung vor Strichrechnung“**: Multiplikation/Division vor Addition/Subtraktion
- **Potenzen vor allen anderen**
- In der naiven Implementierung `5/2+2*3**4/2` also

$$3^4 = 81 \quad \rightarrow \quad 2 * 81/2 = 81 \quad \rightarrow \quad 81 + 2.5 = 83.5$$

- **Lösung: Klammersetzung**
- Klammern haben Priorität vor allen anderen Operatoren
- `5/2+2*3**(4/2)` liefert korrektes Ergebnis

Reihenfolge der Rechenoperationen

- Python berücksichtigt die üblichen Regeln aus der Schule
- **„Punktrechnung vor Strichrechnung“**: Multiplikation/Division vor Addition/Subtraktion
- **Potenzen vor allen anderen**
- In der naiven Implementierung `5/2+2*3**4/2` also

$$3^4 = 81 \quad \rightarrow \quad 2 * 81/2 = 81 \quad \rightarrow \quad 81 + 2.5 = 83.5$$

- **Lösung: Klammersetzung**
- Klammern haben Priorität vor allen anderen Operatoren
- `5/2+2*3**(4/2)` liefert korrektes Ergebnis
- **Quiz**: Warum muss der Ausdruck `5/2` nicht geklammert werden?

Besonderheit bei der Division

Elementares Rechnen in Python

- Besonderheit bei der Division: sogenannte **Links-Assoziativität**

Elementares Rechnen in Python

- Besonderheit bei der Division: sogenannte **Links-Assoziativität**
- Auswertungsreihenfolge ist linker Operand vor rechtem Operand

Elementares Rechnen in Python

- Besonderheit bei der Division: sogenannte **Links-Assoziativität**
- Auswertungsreihenfolge ist linker Operand vor rechtem Operand
- Erinnerung an die Schule: Warum kein Problem bei Multiplikation oder Subtraktion?

Elementares Rechnen in Python

- Besonderheit bei der Division: sogenannte **Links-Assoziativität**
- Auswertungsreihenfolge ist linker Operand vor rechtem Operand
- Erinnerung an die Schule: Warum kein Problem bei Multiplikation oder Subtraktion?
- **Zeit für eine kleine Demo . . .**

Komplexere Operationen

Komplexere Operationen

- **Komplexere Operationen** natürlich auch möglich

Komplexere Operationen

- **Komplexere Operationen** natürlich auch möglich
- Beispiele: Sinus, Cosinus, Exponentialfunktion, Logarithmus, ...

Komplexere Operationen

- **Komplexere Operationen** natürlich auch möglich
- Beispiele: Sinus, Cosinus, Exponentialfunktion, Logarithmus, ...
- Bereitgestellt in der **Bibliothek** `math`

Komplexere Operationen

- **Komplexere Operationen** natürlich auch möglich
- Beispiele: Sinus, Cosinus, Exponentialfunktion, Logarithmus, ...
- Bereitgestellt in der **Bibliothek** `math`
- Nutzung: `import math` einmal am Anfang des Programms

Komplexere Operationen

- **Komplexere Operationen** natürlich auch möglich
- Beispiele: Sinus, Cosinus, Exponentialfunktion, Logarithmus, ...
- Bereitgestellt in der **Bibliothek** `math`
- Nutzung: `import math` einmal am Anfang des Programms
- Später: genaue Funktionsweise von Python-Paketen

Probieren wir es aus!

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

ZOERR

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