LIF-Qualifikationskurs
zu
Intelligenten Suchverfahren
mit Python

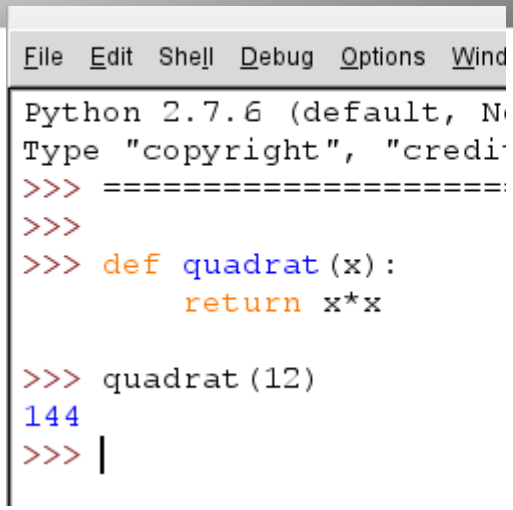
Grundlagen Python

Die Idee zur Reduzierung der Anforderungen im grundlegenden Niveau:

- kein Wechsel der Programmiersprache in der Oberstufe
- Wahl einer Programmiersprache, mit der die Schülerinnen und Schüler ggf. schon in der Mittelstufe arbeiten können

Achtung! Neu! Python kann auch im erweiterten Niveau eingesetzt werden.

Grundlagen Python

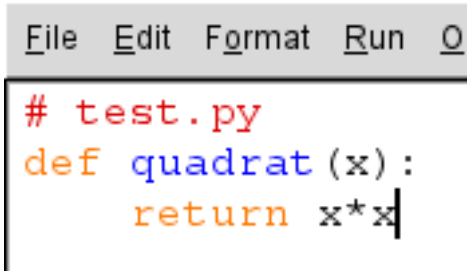


Python ist interaktiv

- Anweisungen kann man direkt auswerten und das Ergebnis direkt angezeigt bekommen
- Ein Python-Programm kann sofort ausgetestet werden
- Der Editor [z.B. IDLE] ist ein struktureller Editor, berücksichtigt die Python-Regeln

Python ist interaktiv

- Anweisungen kann man direkt auswerten und das Ergebnis direkt angezeigt bekommen
- Ein Python-Programm kann sofort ausgetestet werden



- Der Editor [z.B. IDLE] ist ein struktureller Editor, berücksichtigt die Python-Regeln, insbesondere für Einrückungen

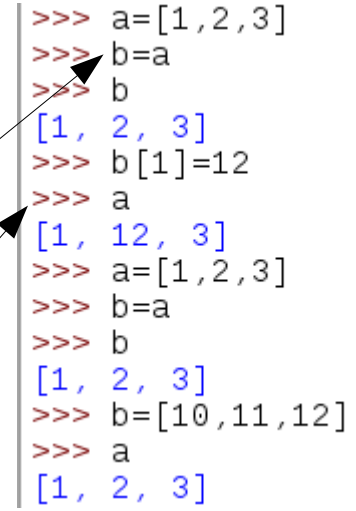
Grundlagen Python

Python ist typgebunden

- Variable kennen ihren Typ
- keine Deklaration von Typen
- a=10
  definiert eine Variable a und weist ihr den Wert 10 zu, der damit Zahltyp ist
- a="10" [oder a='10' mit derselben Wirkung]
  definiert einen String, also eine Zeichenkette

Grundlagen Python

Python verwendet *Referenzen* und Referenzzähler



- ergibt kein neues Objekt, nur eine zweite Referenz
- Veränderungen an einem der Objekte ändern den Wert bei beiden !
- Eine neue Zuweisung für die zweite Referenz erzeugt aber ein neues Objekt und die erste bleibt unverändert
- Achtung auch bei Prozeduren / Funktionen !

Grundlagen Python

Python ist prozedural

- Anweisungen werden sequenziell ausgeführt
- Mit der Anweisung
  `def <Name>(<par>):`
  `    <Anweisung 1>`
  `    <Anweisung 2>`
  können Prozeduren definiert werden

Python ist funktional

- Mit der Definition
  ```
  def <Name>(<par>):
      <...>
      return <Rückgabewert>
  ```
  können Funktionen definiert werden
- Funktionsschachtelungen sind möglich
- Rekursion ist möglich

Grundlagen Python

Python ist funktional

- return → liefert einen Wert zurück
- es gibt anonyme Funktionen: *(leider etwas kompliziert)*
  `lambda x,y: sqrt((x-zX)*(x-zX)+(y-zY)*(y-zY))`
  ist die Abstandsfunktion:

```
def Abstand_vom_Zentrum(zX,zY):
    return lambda x,y: sqrt((x-zX)*(x-zX)+(y-zY)*(y-zY))
```

für ein (global definiertes) Zentrum Z(3,4)

```
abstand_von_Z=Abstand_vom_Zentrum(3,4)

abstand_von_Z(19,16)

→ 20.0
```

- Funktionen können Parameter sein

Python ist voll funktional mit einem **Mangel**

- Python erkennt keine Endrekursion!
- Das führt leider unnötigerweise in einigen Fällen durch Rekursionsabbruch zu einem Fehler (*im Prinzip stack overflow*) wegen zu großer Rekursionstiefe.

Objektorientierung mit Python

Python ist objektorientiert

- Ein großer Teil der Sprache ist objektorientiert definiert
- Das hat auch Probleme zur Folge (s.o.):
  - Listen sind Objekte
  - call-by-reference bei Aufrufen
  - Methoden verändern das Objekt dauerhaft
  - Zuweisungen (a=b) erzeugen keine neuen Objekte