Parameter bei Python

Einführung von Konstruktoren
mit variabler Anzahl von Parametern

Parameter bei Python

- Anlass ist die Aufgabe zum Erstellen einer Tischgruppe aus einem Tisch mit 6 Stühlen.
- Das Arbeiten mit dem Ausgangsprogramm ist unkomfortabel, da alle Stuhlobjekte nach dem Erzeugen dieselbe Position und Orientierung haben.

Parameter bei Python

- Ursache ist der unflexible Konstruktor, der keine Übergabe von Parametern (bis auf den Parameter sichtbar) zulässt.

```
def __init__(self, sichtbar=False):
    """einfacher Konstruktor"""
    self.x=20
    self.y=20
    self.b=40
    self.t=40
    self.w=270
    self.f="blue"
    self.s=sichtbar
    if sichtbar: self.Zeige()
```

Parameter bei Python

- Wünschenswert sind Aufrufe des Konstruktors von Stuhl in der Testanwendung wie beispielsweise hier angegeben:

```
def TestAnwendung(self):
    """Allein fuer die Testanwendung:"""
    global tisch, s1, s2, s3, s4, s5, s6
    tisch=Tisch(70,70,120,60,0,'red',True)
    s1=Stuhl(20,80,40,40,270,'blue',True)
    s2=Stuhl(80,20,40,40,0,'blue',True)
    s3=Stuhl(140,20,40,40,0,'blue',True)
    s4=Stuhl(200,80,40,40,90,'blue',True)
    s5=Stuhl(140,140,40,40,180,'blue',True)
    s6=Stuhl(80,140,40,40,180,'blue',True)
```

Parameter bei Python

- Dazu muss der Konstruktor alle Parameter anbieten.
- Im hier dargestellten Beispiel verlangt er alle:

```
def __init__(self, xPos, yPos, breite, tiefe, winkel, farbe, sichtbar):
    """Konstruktor mit zwingender Parameteruebergabe"""
    self.x=xPos
    self.y=yPos
    self.b=breite
    self.t=tiefe
    self.w=winkel
    self.f=farbe
    self.s=sichtbar
    if sichtbar: self.Zeige()
```

Parameter bei Python

- Alternativ kann man vordefinierte Parameter (*keyword-arguments*) anbieten.

```
def __init__(self,
             xPos=20,
             yPos=20,
             breite=40,
             tiefe=40,
             winkel=270,
             farbe='blue',
             sichtbar=False):
    self.x=xPos
    self.y=yPos
    self.b=breite
    self.t=tiefe
    self.w=winkel
    self.f=farbe
    self.s=sichtbar
    if sichtbar: self.Zeige()
```

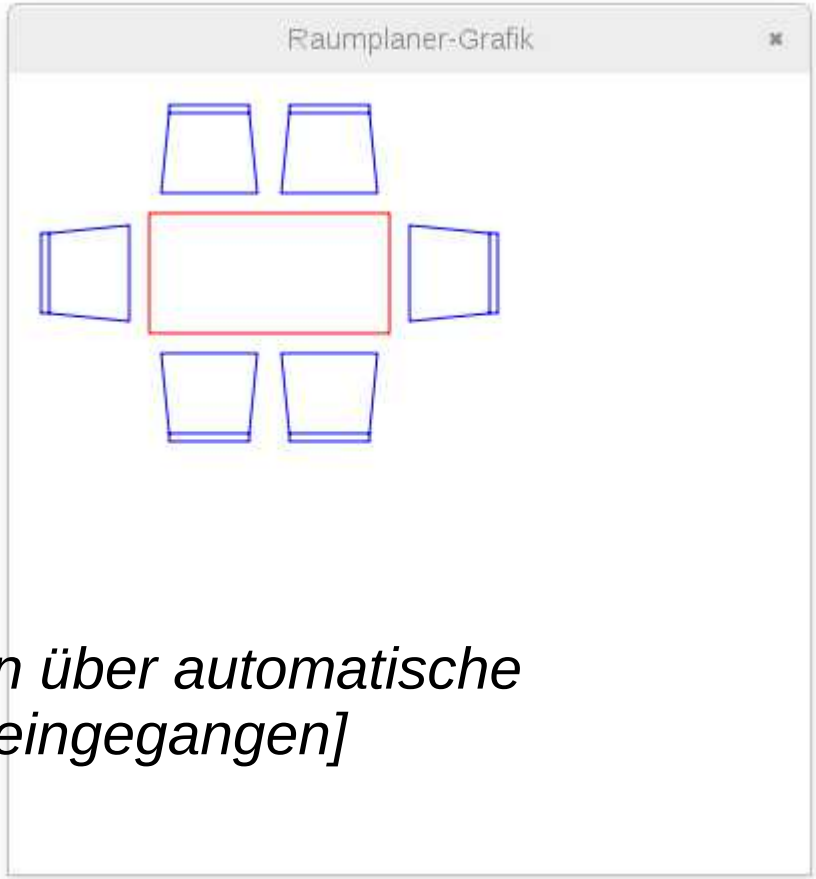
Parameter bei Python

- Das ermöglicht Aufrufe in der hier dargestellten Form:

```
def TestAnwendung2(self):
    """Allein fuer die Testanwendung:"""
    global tisch, s1, s2, s3, s4, s5, s6
    print('global: tisch, s1, s2, s3, s4, s5, s6 ; vordefinierte Parameter')
    tisch=Tisch(70,70,120,60,0,'red',True)
    s1=Stuhl(20,80,winkel=270,sichtbar=True)
    s2=Stuhl(80,20,winkel=0,sichtbar=True)
    s3=Stuhl(140,20,winkel=0,sichtbar=True)
    s4=Stuhl(200,80,winkel=90,sichtbar=True)
    s5=Stuhl(140,140,winkel=180,sichtbar=True)
    s6=Stuhl(80,140,winkel=180,sichtbar=True)
```

Parameter bei Python

- Aber auch die oben angegebenen Aufrufe sind dann immer noch zulässig und erzeugen ebenso das gewünschte Bild.
- Eine Mischung von Positionsparametern und Schlüsselwortparametern ist im Konstruktor ebenso zulässig, allerdings immer in dieser Reihenfolge.



[Auf eine flexible Zahl von Parametern über automatische Erzeugung von Listen wird hier nicht eingegangen]