06d_Miniuebungen_Loesungen

0.1 Mini-Aufgaben zur Überprüfung des Verständnis: Comprehensions

Verwenden Sie Comprehensions, um die folgenden Objekte anzulegen. Entscheiden Sie sich jeweils begründet, ob Sie eine List, eine Set oder eine Dict Comprehension erstellen.

- Wertetabelle für die Funktionen math.sin(), math.cos() und math.tan() für die Argumente $x \in \{0, \frac{\pi}{2}, \pi, \frac{3\pi}{2}, 2\pi\}$ in Form von je vierelementigen tuple $(x, \sin(x), \cos(x), \tan(x))$
- Wertetabelle für das "kleine Einmaleins"
- Raumbelegungsplan ausgehend von einer Liste von Bewohnernamen. Die Raumnummern sind durchlaufend und sollen nicht vorab in einer Liste gespeichert sein, sondern inkrementell beim Durchlaufen der Liste der Bewohner vergeben werden: Ausgehend von [Peter, Paul, Mary] soll also Peter der Raum 1, Paul der Raum 2 und Mary der Raum 3 zugeordnet werden.

[1]:
```
import math

# erstes Beispiel: im Prinzip egal ob als Set oder List Comprehension,
# da die x-Argumente eindeutig sind, ist Set leicht zu favorisieren
arguments = { 0, 0.5*math.pi, math.pi, 1.5*math.pi, 2*math.pi }
# oder eleganter so:
arguments = { i*math.pi for i in {0,0.5,1,1.5,2} }
trigonometry = { (x, math.sin(x), math.cos(x), math.tan(x)) for x in arguments }
print(trigonometry)
```

```
{(0.0, 0.0, 1.0, 0.0), (4.71238898038469, -1.0, -1.8369701987210297e-16,
5443746451065123.0), (6.283185307179586, -2.4492935982947064e-16, 1.0,
-2.4492935982947064e-16), (3.141592653589793, 1.2246467991473532e-16, -1.0,
-1.2246467991473532e-16), (1.5707963267948966, 1.0, 6.123233995736766e-17,
1.633123935319537e+16)}
```

[2]:
```
# zweites Beispiel: hier bietet sich eine Liste an
onebyone = [ i*j for i in range(1,11) for j in range(1,11) ]
print(onebyone)
# sieht blöd aus, deshalb besser so
s = ""
for i in range(len(onebyone)):
    s += "{:4d}".format(onebyone[i])
    if (i+1) % 10 == 0:
        print(s)
```

```
        s = ""
```

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 3, 6, 9, 12,
15, 18, 21, 24, 27, 30, 4, 8, 12, 16, 20, 24, 28, 32, 36, 40, 5, 10, 15, 20, 25,
30, 35, 40, 45, 50, 6, 12, 18, 24, 30, 36, 42, 48, 54, 60, 7, 14, 21, 28, 35,
42, 49, 56, 63, 70, 8, 16, 24, 32, 40, 48, 56, 64, 72, 80, 9, 18, 27, 36, 45,
54, 63, 72, 81, 90, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100]
   1   2   3   4   5   6   7   8   9  10
   2   4   6   8  10  12  14  16  18  20
   3   6   9  12  15  18  21  24  27  30
   4   8  12  16  20  24  28  32  36  40
   5  10  15  20  25  30  35  40  45  50
   6  12  18  24  30  36  42  48  54  60
   7  14  21  28  35  42  49  56  63  70
   8  16  24  32  40  48  56  64  72  80
   9  18  27  36  45  54  63  72  81  90
  10  20  30  40  50  60  70  80  90 100
```

[3]:
```
# drittes Beispiel: hier brauchen wir einen Dictionary
input = ["Peter", "Paul", "Mary"]
plan = { i:input[i] for i in range(0,len(input)) }
print(plan)
```

```
{0: 'Peter', 1: 'Paul', 2: 'Mary'}
```

Implementieren Sie das Beispiel mit Andrea, Peter und Geraldine nur mit Listen und der zip() Funktion.

[25]:
```
names = ['Andrea', 'Peter', 'Geraldine']
ages = [29, 35, 48]

people = dict(zip(names,ages))
print(people)
```

```
{'Andrea': 29, 'Peter': 35, 'Geraldine': 48}
```

Implementieren Sie die Mini-Übungen des vorherigen Abschnitts mit Comprehensions.

[26]:
```
# Exemplarisch nur für Set Comprehensions
demo_string = "ldksfldsafnsdfsakdjgkfsajlfdlsakhfkbcxkgdskjdfslkdhflskd"
#demo_string = "abcdefghijklmnopqrstuvwxyz"

letters = { c for c in demo_string }

# Ergebnis
if len(letters)==26:
    print("Alle Kleinbuchstaben kommen vor.")
else:
    print("Nur die folgenden Kleinbuchstaben kommen vor")
```

```
    print(letters)
```

```
Nur die folgenden Kleinbuchstaben kommen vor
{'n', 'x', 'k', 'l', 'g', 'j', 'f', 's', 'a', 'd', 'h', 'c', 'b'}
```

0.2 Impressum

0.2.1 Programmierkurs Python, Dominik Göddeke https://www.ians.uni-stuttgart.de, Universität Stuttgart

Version vom April 2023

Gefördert durch die Stiftung Innovation in der Hochschullehre. (alle Rechte am Logo vorbehalten)

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

[ ]: