Merkblatt Listen

January 29, 2022

1 Listen

Listen stellen eine Art dar, um in Python mehrere – gleiche wie verschiedene – Objekte zu einem Objekt zusammenzufassen. In Listen erfolgt dies in geordneter Reihenfolge.

Eine Liste wird wie folgt erstellt:

[6]:
```
fruits = ["Apfel", "Banane", "Birne", "Nektarine", "Orange", "Pfirsich"]
```

1.1 append()

Mit `append()` wird ein Element an eine Liste angehängt. Als Parameter wird das einzufügende Element angegeben.

[7]:
```
fruits.append("Kiwi")
print(fruits)
```

```
['Apfel', 'Banane', 'Birne', 'Nektarine', 'Orange', 'Pfirsich', 'Kiwi']
```

1.2 index()

Mit `index()` kannst du den Index (Position) von einem Element herausfinden. Wichtig: Die Zählung fängt bei 0 an!

[8]:
```
print(fruits.index("Banane"))
```

```
1
```

1.3 insert()

Mit `insert()` wird ein Element in einer Liste eingefügt. Dabei gibt der erste Parameter die Position an, der zweite Parameter ist das einzufügende Element. Beachte, dass die Zählung der Elemente einer Liste mit `0` beginnt.

[9]:
```
fruits.insert(3,"Marone")
print(fruits)
```

```
['Apfel', 'Banane', 'Birne', 'Marone', 'Nektarine', 'Orange', 'Pfirsich',
'Kiwi']
```

1.4 join()

Mit `join()` werden alle Elemente einer Liste zu einem String zusammengefügt. Der Parameter für `join()` ist eine Liste. Die Zeichenkette, an der `join()` hängt, dient im String als Zeichen zwischen den Elementen der Liste.

[10]:
```
s= " --- "
print(s.join(fruits))
```

```
Apfel --- Banane --- Birne --- Marone --- Nektarine --- Orange --- Pfirsich ---
Kiwi
```

oder auch

[11]:
```
print(" --- ".join(fruits))
```

```
Apfel --- Banane --- Birne --- Marone --- Nektarine --- Orange --- Pfirsich ---
Kiwi
```

Tipp: Auch Zeilenumbrüche können so ergänzt werden. Diese werden in Strings mit dem Rückschrägstrich angegeben: `"\n"`

[12]:
```
print("\n".join(fruits))
```

```
Apfel
Banane
Birne
Marone
Nektarine
Orange
Pfirsich
Kiwi
```

1.5 pop()

Mit 'pop()' wird das letzte Element einer Liste ausgegeben ***und*** aus der Liste entfernt.

[13]:
```
print(fruits.pop())
print(fruits)
```

```
Kiwi
['Apfel', 'Banane', 'Birne', 'Marone', 'Nektarine', 'Orange', 'Pfirsich']
```

1.6 remove()

Mit `remove()` wird **das erste** Element aus der Liste entfernt, das dem als Parameter angegebenen Wert entspricht.

[14]:
```
fruits.remove("Birne")
print(fruits)
```

```
['Apfel', 'Banane', 'Marone', 'Nektarine', 'Orange', 'Pfirsich']
```

1.7 split()

Mit `split()` wird ein String in eine Liste umgewandelt. Dabei wird als Parameter angegeben, welches Zeichen oder welche Zeichenkette zur Trennung der Elemente benutzt wird. `split()` wird an die Zeichenkette angehängt. Wird kein Parameter angegeben, dann trennt Python den String an einem White-Space (z.B. Leerzeichen, Zeilenumbruch, Tabulator,...).

[15]:
```
t = "Apfel, Banane, Birne, Kirsche, Kiwi, Nektarine, Marone, Pfirsich, Pflaume"
fruits = t.split(', ')
print(fruits)
```

```
['Apfel', 'Banane', 'Birne', 'Kirsche', 'Kiwi', 'Nektarine', 'Marone',
'Pfirsich', 'Pflaume']
```

1.8 Liste auf ein Element testen

Häufg muss gestestet werden, ob ein bestimmtes Element in einer Liste enthalten ist. Dies erfolgt mit dem Vergleichsoperator `in` :

[16]:
```
"Kiwi" in fruits
```

[16]:
```
True
```

`in` liefert einen Wahrheitswert, wie er zum Beuspiel für if-Anweisungen verwendet werden kann. Existiert das Element in der Liste, dann liefert der Test wahr/True.