Dekoratoren und Exceptions - Übung

Richard Müller, Tom Felber

20. Januar 2022

Python-Kurs

Gliederung

1. Übung 1 - Referenzen

2. Übung 2 - Dekoratoren / Exceptions

Übung 1 - Referenzen

Übung 1 - Referenzen

```
1 d = {"a": lambda x: x*2, "b": lambda y, z: y(z)}
2 print(d["b"](d["a"], 4))
```

Was wird ausgeprintet ?

Übung 1 - Referenzen

```
1 d = {"a": lambda x: x*2, "b": lambda y, z: y(z)}
2 print(d["b"](d["a"], 4))
```

```
8
```

Übung 1 - Referenzen

```
1 def rec(depth):
2     if depth <= 1:
3         return 1
4     return rec(depth-1) + rec(depth-2)
5
6 for i in range(5):
7     print(rec(i))
```

Was wird ausgeprintet ?

Übung 1 - Referenzen

```
1 def rec(depth):
2     if depth <= 1:
3         return 1
4     return rec(depth-1) + rec(depth-2)
5
6 for i in range(5):
7     print(rec(i))
```

1 1 2 3 5

Übung 2 - Dekoratoren / Exceptions

Übung 2 - Dekoratoren

Schreibe eine Funktion, die andere Funktionen dekorieren kann. In der Original Funktion auftretende Exceptions sollen von dem Dekorator abgefangen werden.

Übung 2 - Dekoratoren

Schreibe eine Funktion f, die als Parameter einen Fehlertyp und eine Reaktion auf den Fehler (ein String zum ausprinten) entgegen nimmt. f soll eine Funktion zurück geben, die als Dekorator verwendet werden kann. Der zurückgegebene Dekorator soll die Exception abfangen, falls sie in f angegeben wurde.

Beispiel Anwendung:

```
1 @handle_errors(RuntimeError, "RunTimeError")
2 def my_func():
3     raise RuntimeError("Ich bin ein Fehler o.o")
```