Prof. Dr. Claudia Müller-Birn, Barry Linnert

Objektorientierte Programmierung, SoSe 17

Übung 04

TutorIn: Thierry Meurers
Tutorium 10

Stefaan Hessmann, Jaap Pedersen, Mark Niehues

20. Mai 2017

1 Rekursion in Python

Listing 1: Code zu Aufgabe 1

```python
"""
Aufgabe 1
"""


def rekursion(counter=0):
    """
    Calls itself until an error occurs.
    Prints the number of recursion-steps.

    Parameters
    ----------
    counter : int
        should not be set

    Returns
    -------
    None
    """
    error_occured = False
    counter += 1
    if not error_occured:
        try:
            rekursion(counter)
        except:
            end = True
            print("Nach {} Rekursionen ist die Rekursionstiefe erreicht".format(counter))


if __name__ == '__main__':
    rekursion()
```

2 Türme von Hanoi in Python

Listing 2: Code zu Aufgabe 2

```python
"""
http://www.python-kurs.eu/tuerme_von_hanoi.php
"""

def hanoi(n, source, helper, target):
    if n > 0:
        # Put thie print so somewhere, where it makes sense!
        print("Tower 1: {} \t Tower 2: {} \t Tower 3: {}".format(source, helper, target))
        # move tower of size n - 1 to helper:
        hanoi(n - 1, source, target, helper)
        # move disk from source peg to target peg
        if source:
            target.append(source.pop())
        # move tower of size n-1 from helper to target
        hanoi(n - 1, helper, source, target)

n = int(input("How many discs?:\n"))

source = list(range(1,n+1)) # Creates list from 1 to n
source = source[::-1]       # Invertes the order

target = []
helper = []
hanoi(n,source,helper,target)
```

3 Auswirkung der Rekursionstiefe in Python

Einige