**Rekursionstiefe und Anzahl der Funktionsaufrufe**
am Beispiel der Hofstadter-Funktion

Quellcode in Python:

```
def hof(x):
    if x == 1:
        return 1
    elif x == 2:
        return 1
    elif x > 2:
        return hof(x - hof(x - 1)) + hof(x - hof(x - 2))

endwert = int(input('Bis zu welchem Wert soll Hofstadter berechnet werden? '))
n = 1
while n <= endwert:
    y = hof(n)
    print('hof(',n,') =',y)
    n += 1
```

```
h(1) = 1

1 Aufruf    Rekursionstiefe = 1
```

```
h(2) = 1

1 Aufruf    Rekursionstiefe = 1
```

```
h(3) = h(3 - h(2)) + h(3 - h(1))      ←  2-fache Verschachtelung
 1           1             1          ←  # Aufrufe
     = h(3 - 1) + h(3 - 1)
     = h(2) + h(2)
        1      1                      ←  # Aufrufe
     = 1  +  1
     = 2

5 Aufrufe   Rekursionstiefe = 2
```

```
h(4) = h(4 - h(3)) + h(4 - h(2))
 1           5             1          ←  # Aufrufe
     = h(4 - 2)    + h(3)

     = h(2) + h(3)
         1     5                     ←  # Aufrufe

13 Aufrufe insgesamt

     = h(4 - h(3))                          + h(4 - h(2))
     = h(4 - (h(3 - h(2)) + h(3 - h(1)))) + h(3)
     = h(4 - (h(3 - h(2)) + h(3 - h(1))) + h(3 - h(2)) + h(3 - h(1))
       \_____________________________________/
                3-fache Verschachtelung

     = h(4 - (h(3 - 1) + h(3 - 1)) + h(3 - 1) + h(3 - 1)
     = h(4 - (h(2) + h(2)) + h(2) + h(2)
     = h(4 - (1 + 1)) + 1 + 1
     = h(2) + 1 + 1
     = 1 + 1 + 1
     = 3

Rekursionstiefe = 3
```