Python (Rekursion)

Prüfungsvorbereitung

Welche Ausgaben machen die folgenden rekursiven Python-Programme?

Anstelle einer Zeichnung, die jeweils den nächsten rekursiven Selbstaufruf der Funktion durch Pfeile symbolisieren, wird hier ein Funktionsaufruf [z.B. $f(7)$] durch den nächsten inneren Funktionaufruf [z.B. $f(7-1)+3$] ersetzt, bis der Base-Case erreicht wird [z.B. $f(1)=7$]. Danach kann der Term mit dem zurückgegebenen Wert aufgelöst, d.h. ausgerechnet werden.

Aufgabe 1

```
def f(n):
    if n == 1:
        return 7
    else:
        return f(n-1)+1

print(f(3))
```

Aufgabe 1

```
def f(n):
    if n == 1:
        return 7
    else:
        return f(n-1)+1

print(f(3))
```

$$
\begin{aligned}
f(3) &= f(3-1)+1 = f(2)+1 \\
&= (f(2-1)+1)+1 = f(1)+2 \\
&\overset{(*)}{=} 7+2 = 9
\end{aligned}
$$

$(*)$ Die Rekursion hat den Base-Case erreicht $[n = 1 \Rightarrow f(1) = 7]$ und kann aufgelöst werden.

Aufgabe 2

```
def f(n):
    if n < 2:
        return 5
    else:
        return 2*f(n-3)

print(f(9))
```

Aufgabe 2

```
def f(n):
    if n < 2:
        return 5
    else:
        return 2*f(n-3)

print(f(9))
```

$$
\begin{aligned}
f(9) &= 2 \cdot f(9-3) = 2 \cdot f(6) \\
&= 2 \cdot (2 \cdot f(6-3)) = 4 \cdot f(3) \\
&= 2 \cdot (4 \cdot f(3-3)) = 8 \cdot f(0) \\
&\overset{(*)}{=} 8 \cdot 5 = 40
\end{aligned}
$$

$(*)$ Die Rekursion hat einen Base-Case erreicht $[n = 0 < 2 \Rightarrow f(0) = 5]$ und kann aufgelöst werden.

Aufgabe 3

```
def f(n):
    if n > 6:
        return 3
    else:
        return f(n+2)+4

print(f(1))
```

Aufgabe 3

```
def f(n):
    if n > 6:
        return 3
    else:
        return f(n+2)+4

print(f(1))
```

$$
\begin{aligned}
f(1) &= f(1+2)+4 = f(3)+4 \\
&= (f(3+2)+4)+4 = f(5)+8 \\
&= (f(5+2)+4)+8 = f(7)+12 \\
&\overset{(*)}{=} 3+12 = 15
\end{aligned}
$$

$(*)$ Die Rekursion hat einen Base-Case erreicht $[n = 7 \Rightarrow f(7) = 3]$ und kann aufgelöst werden.

Aufgabe 4

```
def f(n):
    if n == 1:
        return 10
    else:
        return f(n-1) + n

print(f(4))
```

Aufgabe 4

```
def f(n):
    if n == 1:
        return 10
    else:
        return f(n-1) + n

print(f(4))
```

$$
\begin{aligned}
f(4) &= f(4-1) + 4 = f(3) + 4 \\
&= \big(f(3-1) + 3\big) + 4 = f(2) + 7 \\
&= \big(f(2-1) + 2\big) + 7 = f(1) + 9 \\
&\overset{(*)}{=} 10 + 9 = 19
\end{aligned}
$$

$(*)$ Die Rekursion hat den Base-Case erreicht $[n = 1 \Rightarrow f(1) = 10]$ und kann aufgelöst werden.

Aufgabe 5

```
def f(n):
    if n == 0:
        return 1
    else:
        return n * f(n-1)

print(f(4))
```

Aufgabe 5

```
def f(n):
    if n == 0:
        return 1
    else:
        return n * f(n-1)

print(f(4))
```

$$
\begin{aligned}
f(4) &= 4 \cdot f(4-1) = 4 \cdot f(3) \\
&= 4 \cdot \big(3 \cdot f(3-1)\big) = 12 \cdot f(2) \\
&= 12 \cdot \big(2 \cdot f(2-1)\big) = 24 \cdot f(1) \\
&= 24 \cdot 1 \cdot f(1-1) = 24 \cdot f(0) \\
&\overset{(*)}{=} 24 \cdot 1 = 24
\end{aligned}
$$

$(*)$ Die Rekursion hat den Base-Case erreicht $[n = 0 \Rightarrow f(0) = 1]$ und kann aufgelöst werden.