Python (Funktionen)

Prüfungsvorbereitung

Wenn nichts anderes steht, ist die Ausgabe des Python-Programmfragments anzugeben.

Aufgabe 8.1

```
def f(x):
    return 2*x + 4

print(f(9))
```

Aufgabe 8.1

22

Aufgabe 8.2

```
def f(x):
    return 19

print(f(3))
```

Aufgabe 8.2

19

Aufgabe 8.3

```
def f(x):
    x + 1

print(f(6))
```

Aufgabe 8.3

```
None
```

Aufgabe 8.4

```
def f(a, b):
    return 2*a + b

print(f(5, 7))
```

Aufgabe 8.4

17

Aufgabe 8.5

```
def f(a, b):
    return 2*a + b

print(f(b=5, a=7))
```

Aufgabe 8.5

19

Aufgabe 8.6

```
def f(x):
    return 2*x+1

print(f(f(f(1))))
```

Aufgabe 8.6

15

Aufgabe 8.7

```
def f(x):
    return 2*x - 1

def g(x):
    return x*x

print(f(5) + g(3))
```

Aufgabe 8.7

18

Aufgabe 8.8

```
def f():
    return 9

print(f())
```

Aufgabe 8.8

9

Aufgabe 8.9

```
def f(x):
    if x < 0:
        return 1
    else:
        return 0

print(f(8))
```

Aufgabe 8.9

0

Aufgabe 8.10

```
def f(x):
    a = 4
    print(a+x)

a = 3
f(5)
print(a)
```

Aufgabe 8.10

9
3

Variablen, die in einer Funktion definiert sind, sind nur im betreffenden Funktionsrumpf (eingerückten Bereich) gültig. Gleichzeitig „überschatten“ sie globale Variablen mit gleichem Namen. Nach der Ausführung werden die lokalen Variablen wieder gelöscht und globale Variablen mit gelichem Namen werden wieder sichtbar.

Aufgabe 8.11

Schreibe eine syntaktisch korrekte Funktion mit dem Namen `dreieckUmfang(...)`, die aus den drei Seitenlängen a, b und c den Dreiecksumfang berechnet und als Wert zurückgibt.

Aufgabe 8.11

```
def dreieckUmfang(a, b, c):
    return a + b + c
```

Aufgabe 8.12

Schreibe eine syntaktisch korrekte Funktion mit dem Namen `trapezInhalt(...)`, die aus den gegebene parallelen Seiten a und c sowie der Höhe h den Flächeninhalt des Tapzes berechnet und als Wert zurückgibt.

Hinweis: $A_{\text{Trapez}} = \frac{a+c}{2} \cdot h$

Aufgabe 8.12

```
def trapezInhalt(a, c, h):
    m = (a+c)/2
    return m*h
```

Aufgabe 8.13

```
def f(x, y, z):
    x * y + z

print(f(2, 3, 4))
```

Aufgabe 8.13

```
None
```

Aufgabe 8.14

```
def f(x=2, y=1):
    return 2*x + 3*y

print(f(3))
```

Aufgabe 8.14

9

Aufgabe 8.15

```
def f(x):
    return 7

print(f(8))
```

Aufgabe 8.15

7

Aufgabe 8.16

```
def f(x):
    return 2*x - 1

print(f(f(2)))
```

Aufgabe 8.16

5

Aufgabe 8.17

```
def f(x, y):
    return x + y

def g(a, b):
    return a * b

print(f(1, 2) + g(3, 4))
```

Aufgabe 8.17

15

Aufgabe 8.18

```
def f(x):
    y = 2*x

y = 8
f(3)
print(y)
```

Aufgabe 8.18

8

Aufgabe 8.19

```
def f(L, x):
    L.append(x)

L = [1, 2, 3]
f(L, 4)
print(L)
```

Aufgabe 8.19

```
[1, 2, 3, 4]
```

Aufgabe 8.20

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

print(f(4))
```

Aufgabe 8.20

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

print(f(4))
```

$$
\begin{aligned}
f(4) &= f(3) + 4 \\
&= (f(2) + 3) + 4 \\
&= ((f(1) + 2) + 3) + 4 \\
&= (((f(0) + 1 + 2) + 3) + 4 \quad \text{(Base Case erreicht)} \\
&= (((1 + 1) + 2) + 3) + 4 \\
&= 11
\end{aligned}
$$

```
11
```

Aufgabe 8.21

```
def f(n):
    if n < 2:
        return n
    else:
        return 2*f(n-2) + 1

print(f(5))
```

Aufgabe 8.21

```
def f(n):
    if n < 2:
        return n
    else:
        return 2*f(n-2) + 1

print(f(5))
```

$$
\begin{aligned}
f(5) &= 2 * f(3) + 1 \\
&= 2 * (2 * f(1) + 1) + 1 \quad \text{(Base Case erreicht)} \\
&= 2 * (2 * 1 + 1) + 1 \\
&= 2 * 3 + 1 = 7
\end{aligned}
$$

7

Aufgabe 8.22

```
def f(x, y, z):
    return x+y, z-y

a, b = f(7, 2, 5)
print(a+b)
```

Aufgabe 8.22

12