1. (3 Punkte) a. Erläutere am Beispiel die folgenden Begriffe: globale Variable, lokale Variable, Parameter und Argument. b. Was wird ausgegeben?

```
k = 10
def doit(y):
    r = 5
    w = k + y + r
    return w

z = doit(3) + k
print(z)
```

2. (1 Punkt) Schreibe eine Funktion, die sich so verhält, wie im folgenden docstring beschrieben.

```
'''
returns: None,  Gibt die Zeichenfolge 'x.' aus

>>> func1()
x.
'''
```

3. (1 Punkt) Schreibe eine Funktion, die sich so verhält, wie im folgenden docstring beschrieben.

```
'''
returns: string, die Zeichenfolge 'x.'

>>> func2()
'x.'
'''
```

4. (1 Punkt) Schreibe eine Funktion, die sich so verhält, wie im folgenden docstring beschrieben.

```
'''
s: nicht leerer string
returns: s ohne erstes und letztes Zeichen.

>>> func01('Hallo')
'all'
'''
```

5. (1 Punkt) Schreibe eine Funktion, die sich so verhält, wie im folgenden docstring beschrieben.

```
'''
x, k: ints, k >= 0, k optional mit default 1
returns: x * k

>>> func(2, 5)
10
>>> func(2)
2
'''
```

6. (1 Punkt) Schreibe eine Funktion, die sich so verhält, wie im folgenden docstring beschrieben.

```
'''
x: int
returns 'A' falls x < 20
        'B' falls 20 <= x < 30
        'C' falls 30 <= x

>>> func(15)
'A'
>>> func(25)
'B'
>>> func(35)
'C'
'''
```

7. (1 Punkt) Schreibe eine Funktion, die sich so verhält, wie im folgenden docstring beschrieben.

```
'''
a, b: ints
returns: Tupel aus a//b und a%b

>>> func(20,3)
(6, 2)
```

8. (1 Punkt) Was wird ausgegeben?

```
x = 10
def doit(k):
    global x
    x = 5
    z = 7
    w = k + z
    return w

z = doit(3) + x
print(z)
```

9. (2 Punkte) Was wird ausgegeben?

```
def a(x):
    return x + 1
def b(x):
    return x + 2
def c(x,y):
    return x + y
def d(x,y):
    return x > y
z = c(a(3),b(5))
print(d(z,10),z)
```

10. (2 Punkte) Schreibe eine Funktion, die sich so verhält, wie im folgenden docstring beschrieben.

```
'''
s: String mit mindestens zwei Zeichen
returns: True, wenn die ersten beiden Zeichen von s gleich sind
         False, sonst

>>> pruef01('aab')
True
>>> pruef01('abb')
False
'''
```

11. (4 Punkte) Schreibe eine Funktion, die sich so verhält, wie im folgenden docstring beschrieben.

```
'''
n: int, n > 0
returns: die Summe der echten Teiler von n. Echte Teiler sind alle Teiler der Zahl,
         außer die Zahl selbst. Beispiel: die Summe der echten Teiler
         von 12 ist: 1+2+3+4+6 = 16

>>> echteTeilerSumme(12)
16
'''
```