1. (3 Punkte) a. Weise der Variablen a ein leeres Tupel zu
   b. Weise der Variablen b ein Tupel mit dem Element 'B' zu.
   c. Weise der Variablen c ein Tupel mit den booleschen Werten True, True, False zu.

**Lösung:**

```
a = ()
b = ('B',)
c = (True, True, False)
```

2. (3 Punkte) Zu was werten sich die Ausdrücke nach der ersten Zuweisung aus?

```
a = ((1,2,3),('a',('b',4)))
a. a[1][2]
b. a[0][2]
c. a[1][1]
```

**Lösung:**

```
a. error
b. 3
c. ('b', 4)
```

3. (2 Punkte) Was wird an der Konsole ausgegeben?

```
a = (1,4)
b = a * 2
c = b + (5,4,7)
d = c[2:5] + b[3:4]
print(d)
```

**Lösung:**

```
(1, 4, 5, 4)
```

4. (2 Punkte) Schreibe eine Funktion `minmax`, der ein nicht leeres Tupel aus Zahlen übergeben wird und die das Mininum und Maximum der Zahlen zurückgibt.

**Lösung:**

```
def minmax(t):
    return min(t),max(t)
```

5. (4 Punkte) Implementiere die Funktion `maxchar`

```
def maxchar(s):
    '''
    s: nicht leerer String
    returns: Tupel (c,k)
       c: Zeichen, das am häufigsten in s vorkommt, bei mehreren Zeichen das erste.
       k: Anzahl der Vorkommen von c

    Beispiel:
    >>> s = 'abbaacccccaa'
    >>> maxchar(s)
    ('a', 5)
    '''
```

**Lösung:**

```
def maxchar(s):
    maxZeichen = s[0]
    maxCount = s.count(maxZeichen)
    for c in s:
        if s.count(c) > maxCount:
            maxZeichen = c
            maxCount = s.count(c)
    return maxZeichen, maxCount
```

6. (2 Punkte) Gegeben ein Tupel `hugo` aus Strings. Gehe durch das Tupel und gib die Längen der Strings aus.

**Lösung:**

```
for s in hugo:
    print(len(s))
```