Aufgabe 1)

Welche asymptotische Laufzeitkomplexität besitzt dieser Algorithmus?

- O(n!). Jeder Aufruf des Algorithmus mit einem Wort der Länge n läuft n -mal durch die for-Schleife und ruft den Algorithmus erneut mit n-1 auf. $\prod_{i=1}^{n} i = n!$

Welche Art von Rekursion verwendet dieser Algorithmus?

- Nichtlineare Rekursion, da jeder Aufruf des Algorithmus mehrere (gleich große) Subaufrufe verursacht.

Aufgabe 2)

Ergebnisse:

- someFunction(2,5) = 32
- someFunction(10,10) = 1000

Was macht die gezeigte Funktion?

- someFunction(a,b) berechnet a^b. a^0 = 0, a^1 = a
- Wenn b%2 == 0, a^b = (a^2)^(b/2)
- Wenn b%2 != 0, (int)(b/2) = (b-1)/2. Dadurch a^b = ((a^2)^(b/2)) * a

Laufzeitkomplexität:

- O(log b) bzw O(log b). Laufzeit wächst schwächer als b und flacht mit steigendem b ab. Vergleiche hier auch binäre Suche.