1.)

Big O

* Maß für die Anzahl der Elementarschritte/ Speichereinheiten in Abh. Von der Größe des Problems
* Gibt die Komplexität eines Problems an bzw. klassifiziert dieses
* Rechenzeit steigt mit der Anzahl der Rechenschritte
* Benötigter Speicherplatz steigt mit der Eingabelänge

2.)

Der Unterschied ist die Komplexität als benötigter Speicherplatz bzw Anwendung

Data structure – je größer desto mehr Speicherplatz wird benötigt

Operation, algorithms, … - je mehr Schritte programmiert werden desto höher die Rechenzeit

Je nach Operation steigt die Komplexität

X+1 -> sehr einfach σ (1)

X+Y -> schwerer σ(x)

X \* X -> schwerer σ(x²) , usw.

3.)

Vector/ array -> 𝒪 (1)

Matrix/ Table -> 𝒪(m x n )

Tree -> 𝒪(n²)

Graph with n vertices and e edge -> ?

4.) nur Durchschnitt angegeben

Merge sort -> 𝒪(n\*log(n))

Quick sort -> 𝒪(log²(n))

Bubble sort -> 𝒪(n)

Matrix multiplication -> 𝒪(n³) für quadratische Matrizen

Exercise

Siehe: „erster\_mergesort“, „erster\_quicksort“