1 Feinanalyse

a)  
**simplePower**Hier wird die Variable **p** einfach zu der Variable **x** fortlaufend multipliziert bis wir die gewünschte Potenz erreicht haben.

**quickPower**Hier wird das ganze mit dem log2(n) verfahren gemacht. Da wird solange durch eine Schleife durch iteriert bis **n** <= 0 ist. **N** wird bei jeder Iteration halbiert und **x** mit sich selber multipliziert, daher auch das logarithmische verfahren. Die Variable **p** wird wenn **n** mod 2 != 0 ist mit **x** multipliziert.

quickPower hat seinen namen verdient weil er logarithmisch wächst. Ab einer gewissen Problemgröße ist er um einiges schneller.

b)

Feinanalyse siehe beigelegtes excel file.

c)

Ab einer Problemgröße von n >= 12 sollte **quickPower** schneller sein. Bei n < 12 ist **simplePower** schneller, da weniger Operationen pro Durchgang ausgeführt werden. **quickPower** ist bei größeren Problemen schneller da er logarithmisch wächst und somit die Konstanten nicht mehr ins Gewicht fallen.

2 Grobanalyse

**findCharLeft**Hier wird von links nach rechts der string durch iteriert bis der passende Character gefunden wurde.

Testfälle:  
(“abc”, ‘b’) (“abc”, ‘a’) (“abc”, ‘c’) (“abc”, ‘d’) (“”, ‘i’) (“ewniobpmwEDasdgkRLÖPQAM“, ‘k’)

Best, average und worst case im c.file beschreiben.

**findCharRight**Hier wird zuerts die string länge gezählt, dannach wird von rechts nach links durch iteriert bis der passende Character gefunden wurde.

Testfälle:  
(“abc”, ‘b’) (“abc”, ‘a’) (“abc”, ‘c’) (“abc”, ‘d’) (“”, ‘i’) (“ewniobpmwEDasdgkRLÖPQAM“, ‘k’)

Best, average und worst case im c.file beschreiben.

**findCharRight\_2**Hier wird von links nach rechts durch iteriert bis der letzte passende Character gefunden wurde.

Testfälle:  
(“abc”, ‘b’) (“abc”, ‘a’) (“abc”, ‘c’) (“abc”, ‘d’) (“”, ‘i’) (“ewniobpmwEDasdgkRLÖPQAM“, ‘k’)

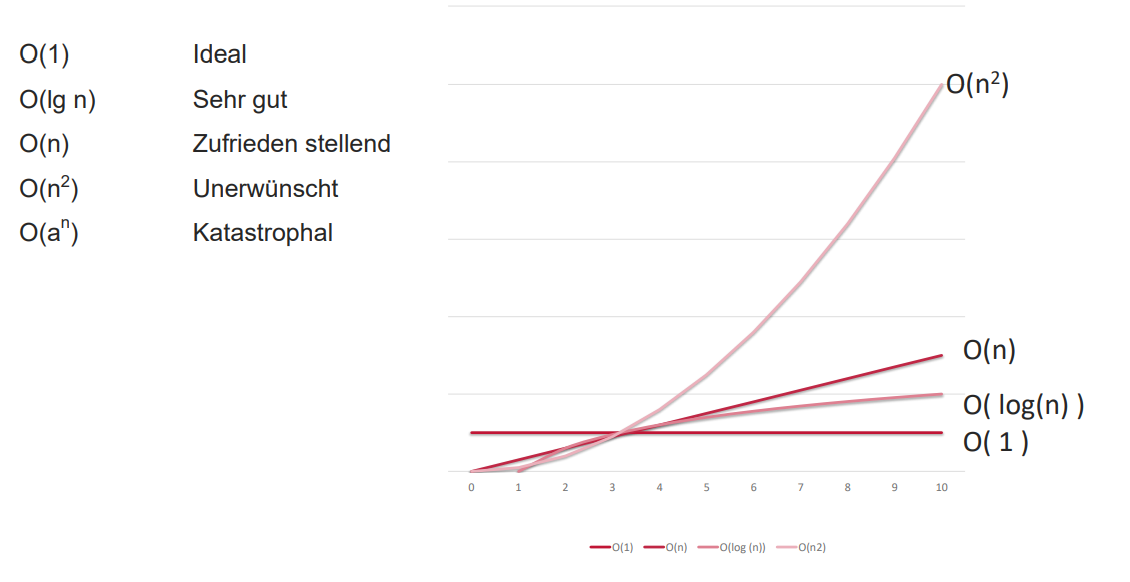
Best, average und worst case im c.file beschreiben.

**findCharRandom**Hier wird zuerst die string länge gezählt, dannach wird der zu prüfende Index mit einer random funktion bestimmt, dies wird solange gemacht bis der gesuchte Character gefunden ist. Bei 1000 versuchen wird aber abgebrochen und eine Fehlermeldung ausgegeben.

Testfälle:  
(“abc”, ‘b’) (“abc”, ‘a’) (“abc”, ‘c’) (“abc”, ‘d’) (“”, ‘i’) (“ewniobpmwEDasdgkRLÖPQAM“, ‘k’)

Best, average und worst case im c.file beschreiben.

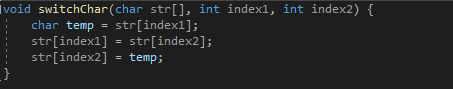
3 O-Notation  
In der Vorlesung wurden die unterschiedlichen Komplexitätsklassen besprochen. Suchen Sie in „echten“ Quellen für die Komplexitätsklassen O(1), O(n), O(log(n)) und O(n²) jeweils zwei Algorithmen. Überlegen Sie sich auch eine Begründung für die Einordnung in die jeweilige Klasse.

****

**O(1)**Mit O(1) ist eine konstante laufzeit eines Algorithmus gemeint.

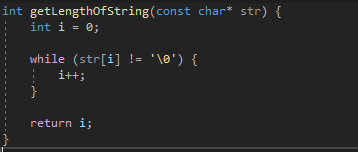
Beispiele:

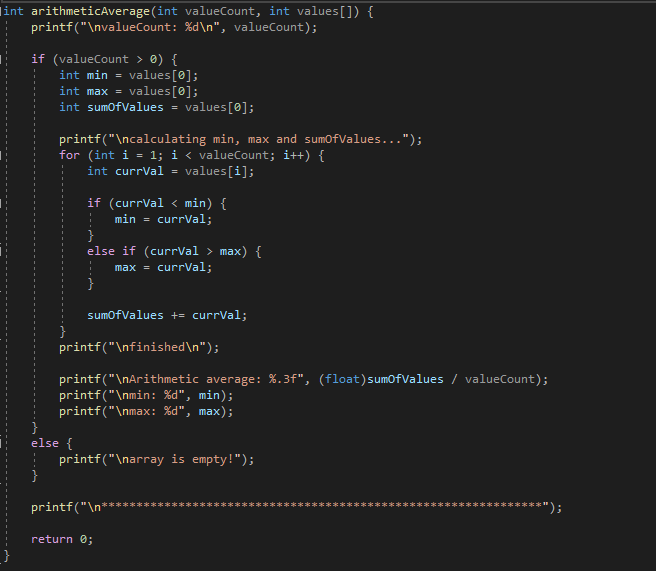




**O(n)**Mit O(n) ist eine lineare laufzeit eines Algorithmus gemeint.

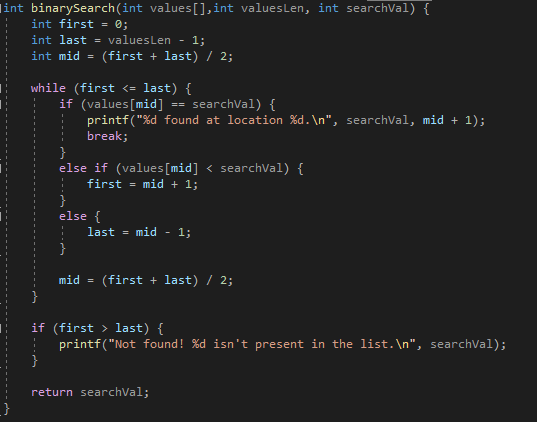
Beispiele:

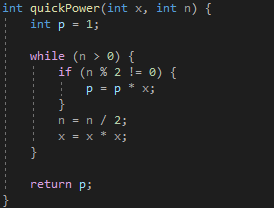




**O(log(n))**Hier nimmt die Laufzeit des Algorithmus Logarithmisch zur Problemgröße zu.

Beispiele:





**O(n²)**Hier nimmt die Lauftzeut des Algorithmus zur Problemgröße Quadratisch zu.  
  
