Laufzeitanalyse Big O

Datenmenge n O(n)

- Worst Case

- Best Case

- Average Case

Grundregeln

Konstante ignorieren

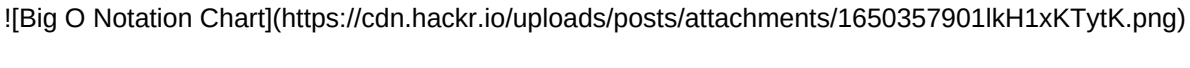
Beispiel:

5n -> O(n)

Dominante Terme

Die folgenden Laufzeitkomplexitäten in aufsteigender Reihenfolge:

O(1) < O(logn) < O(n) < O(nlogn) < O(n^2) < O(2^n) < O(n!)



Konstantzeit (Constant Time)

Beispiel:

```python
x = (5 + 15 * 20)  # O(1)
```

Ein komplexeres Beispiel:

```python
```

```
x = (5 + 15 * 20)  # O(1)
y = 15 - 2         # O(1)
print(x + y)       # O(1)
```

Gesamtzeit:

O(1) + O(1) + O(1) = 3 * O(1) -> O(1)

Lineare Zeit (Linear Time)

Beispiel:

```python
for x in range(0, n):  # Läuft von 0 bis n also n mal
    print(x)           # O(1)
```

Gesamtzeit:

n * O(1) = O(n)

Ein weiteres Beispiel:

```python
y = 5 + (15 * 20)     # O(1)
for x in range(0, n): # O(n)
    print(x)          # O(1)
```

Gesamtzeit:

O(1) + O(n) = O(n) (weil wir niedrigere Ordnungsterme vernachlässigen)

Quadratische Zeit (Quadratic Time)

Beispiel:

```python
for x in range(0, n):    # O(n)
    for y in range(0, n): # O(n)
        print(x * y)     # O(1)
```

Gesamtzeit:

O(n) * O(n) * O(1) = O(n^2)

Mastertheorem

Siehe Skript Müller.