Fallunterscheidungen

- I........Input
- $|I|$....Größe des Inputs (z.B.: Anzahl der Elemente)
- $T(I)$....Anzahl der elementaren Schritte die ein Algorithmus für Input I benötigt.

- **Laufzeit Worst Case:**

$$T(n) = \max\{T(I) : |I| = n\}$$

- **Laufzeit Best Case:**

$$T(n) = \min\{T(I) : |I| = n\}$$

- **Laufzeit Average Case:**

$$T(n) = \frac{1}{|I_n|} \sum_{I \in I_n} T(I) \qquad mit\ I_n = \{I : |I| = n\}$$

- Gründe für **worst-case** Analyse:
  - Laufzeit im schlimmsten Fall – kei
  - Kommt häufig vor (z.B.: Suchen na
  - Mittlerer Fall oft ordnungsmäßig
  - Oft sehr einfach am Pseudocode a
- Gründe für **best-case** Analyse:
  - Laufzeit im besten Fall
  - Sinnvoll, wenn der beste Fall sehr
- Gründe für **average-case** Analyse
  - Um eine durchschnittliche Perfor
  - Wenn der worst case sehr selten i
  - Zusätzlich Wissen über die Verteil
  - Meist komplizierter in der Analyse

Best Case: Array ist bereits aufsteigend sortiert

$t_i = 1 \quad \forall i$

$T_{best}(n) = (c_0 - c_2 - c_3 - c_7) + (c_1 + c_2 + c_3 + c_7) \cdot n + c_4 (n-1)$

$= k \cdot n + d$ für irgendwelche Konstanten k, d

$\Rightarrow$ lineare Laufzeit

Worst Case: Arr

$t_i = i + 1 \quad \forall i$

$\sum_{i=1}^{n-1} (i+1) = \sum_{i=2}^{n} i =$

$\sum_{i=1}^{n-1} i = \frac{(n-1) \cdot n}{2}$

$T_{worst}(n) = l n^2 +$

$\Rightarrow$ qua

Insertionsort

Average Case

$E[t_i] = \frac{i}{2} + 1$

$\sum_{i=1}^{n-1} \frac{i}{2} + 1 - 1 = \sum_{i=1}^{n-1} \frac{i}{2} = \frac{1}{2} \sum_{i=1}^{n-1} i = \frac{1}{2} \frac{(n-1)n}{2}$

$T_{avg}(n) = l \cdot n^2 + k \cdot n + d$ für Konstanten l, k, d;