

Aufgabe 1

1.1

- Laufzeit: Der Algorithmus verhält sich wie der Standard Insertion Sort und unterscheidet sich nur beim Vergleich der aktuellen Zahlen (aus der Vorlesung folgt also):

Best Case:

$$T(n) = \sum_{i=1}^{n-1} c \in \Theta(n) \text{ mit } c \in \mathbb{N}$$

Average Case:

$$T(n) = \sum_{i=1}^{n-1} \frac{i}{2} = \frac{1}{2} \sum_{i=1}^{n-1} i = \frac{n(n-1)}{4} \in \Theta(n^2)$$

Worst Case:

$$T(n) = \sum_{i=1}^{n-1} i = \frac{n(n-1)}{2} \in \Theta(n^2)$$

- Korrektheit: Behauptung: Nach j-ten Schleifendurchlauf gilt: $a_0 \leq a_1 \leq \dots \leq a_j$

IA: $j = 1$

- $\text{key} = a[j] = a[1]$ und $i = 0$

- case 1: $a_i = a_0 \leq \text{key}$ so wird while-Schleife nicht betreten und es gilt: $a_0 = a_i \leq a_j$
- case 2: $a_i = a_0 > \text{key}$, so wird die while-Schleife betreten und $a[i]$ und $a[j]$ getauscht also gilt nach dem Tausch: $a_0 = a_i < a_j = a_1$

IV: es gibt ein $j-1$ für das die Aussage gilt

IB: für dieses $j-1$ gilt die Aussage auch für j

IS: $j-1 \rightarrow j$

- nach IV gilt $a_0 \leq a_1 \leq \dots \leq a_{j-1}$
- bei der j-ten Iteration wird a_j zwischengespeichert in key und durch das bereits sortierte Teilarray. Ist nun die untersuchte Zelle kleiner als key so wird diese Zelle nachgerückt. Ist diese Zelle nun größer so wird die Iteration abgebrochen, da dann alle Elemente die an kleineren Position stehen auch kleiner sind (IV). Zuletzt wird a_j an dem letzten Index geschrieben, sodass $a_0 \leq \dots \leq \text{key} \leq \dots \leq a_{j-1+1} = a_j$ gilt.

1.2

- Laufzeit Analog zu anderen einfachen Sortieralgorithmen:

$$\text{WC} = \text{BC} = \text{AC} = T(n) \in \Theta(n^2)$$

- Korrektheit: Verhält sich analog zum normalen Bubblesort nur anstatt zum Minimum hin zu sortieren, sortiert dieser zum Maximum

1.3

- Laufzeit $\text{WC} = \text{BC} = \text{AC} = T(n) \in \Theta(n^2)$
- Korrektheit: Äquivalent zum normalen Selection Sort. Unterschied: es wird nach dem Max index gesucht an diesem Index hin getauscht

1.4

- Laufzeit

Best Case: (Halbierung jeden Zug) $T(n) = 2T\left(\frac{n}{2}\right) + n \in \Theta(n \log n)$

Worst Case:(Pivot ist immer ganz links oder rechts) $T(n) = T(n-1) + T(0) + n \in \Theta(n^2)$
Average Case:(Annahme alle Positionen sind gleich wahrscheinlich)

$$T(n) = \left(\frac{T(1)}{2} - \frac{1}{2} + 2 \sum_{i=3}^n \frac{1}{i} + \frac{3}{n+1} \right) (n+1)$$

Und da: $\sum_{i=1}^n \frac{1}{i} \in \Theta(\log n)$

$$T(n) = \left(\frac{T(1)}{2} - \frac{1}{2} + 2 \sum_{i=3}^n \frac{1}{i} + \frac{3}{n+1} \right) (n+1) \in \Theta(n \log n)$$

- Korrektheit: Siehe Vorlesung

Aufgabe 2

siehe implementation

Aufgabe 3

siehe implementation