Sarah Ertel
Patrick Greher
Eugen Ljavin

1	2	3	4	5	Σ
1	0	6	4	3	14

Übungsblatt Nr. 5

(Abgabetermin 31.05.2018)

Aufgabe 1

Die Zahlen befinden sich zwischen $[0, ..., n^k - 1]$, was $[n^k - n^k, ..., n^k - 3, n^k - 2, n^k - 1]$ entspricht. Dabei ist Vorausgesetzt, dass es sich hierbei um ein festes $k \epsilon \mathbb{N}$ handelt.
Es wird für jede Zahl im angegebenen Bereich ein Bucket erstellt. Der Aufwand dafür beträgt $\mathcal{O}(n)$. Warum ist das O(n) und nicht O(n^k)?
Da jeder Bucket einen einzelnen einzigartigen Wert darstellt müssen die Zahlen innerhalb eines Buckets nicht weiter sortiert werden. Es ist folglich egal, ob jede Zahl ein einem eigenen Bucket liegt, alle Zahlen in einem Bucket liegen oder sie zufällig verteilt sind. Es entfällt der Aufwand der Sortierung innerhalb der Buckets. Für das Zusammenfügen der Buckets ist nur noch ein Aufwand von $\mathcal{O}(1)$ notwändig.
Der gesamte Aufwand ist folglich auch im Worst-Case $\mathcal{O}(n) + \mathcal{O}(1)$ was $\mathcal{O}(n)$ entspricht.

Aufgabe 2

a)

b)

i)

Sei $T(n) = T(\frac{n}{5}) + T(\frac{7}{10}n) + \mathcal{O}(n)$ die Rekursionsgleichung. Angenommen es gilt $T(n) < c \cdot n$. Folglich gilt:

$$
\begin{aligned}
T(n) &= T(\frac{n}{5}) + T(\frac{7}{10}n) + d \cdot n \\
c \cdot n &\geq T(\frac{n}{5}) + T(\frac{7}{10}n) + d \cdot n \\
c \cdot n &\geq c \cdot \frac{n}{5} + c \cdot \frac{7}{10}n + d \cdot n \\
c &\geq \frac{9c}{10} + d \\
\frac{c}{10} &\geq d \\
c &\geq 10 \cdot d
\end{aligned}
$$

ii)

Wird die Liste in siebener- statt fünfergruppen geteilt, lautet die Rekursionsgleichung $T(n) = T(\frac{n}{7}) + T(\frac{10}{14}n) + \mathcal{O}(n)$

Sei $T(n) = T(\frac{n}{7}) + T(\frac{10}{14}n) + \mathcal{O}(n)$ die Rekursionsgleichung. Angenommen es gilt $T(n) <$

$c \cdot n$. Folglich gilt:

$$
\begin{aligned}
T(n) &= T(\frac{n}{7}) + T(\frac{10}{14}n) + d \cdot n \\
c \cdot n &\geq T(\frac{n}{7}) + T(\frac{10}{14}n) + d \cdot n \\
c \cdot n &\geq c \cdot \frac{n}{7} + c \cdot \frac{10}{14}n + d \cdot n \\
c &\geq \frac{12c}{14} + d \\
\frac{c}{14} &\geq d \\
c &\geq 14 \cdot d
\end{aligned}
$$

Das ist nicht was gesucht ist.

Aufgabe 3

a)

Zunächst wird aus beiden Arrays A und B der Median bestimmt. Ist der Median von A kleiner als der Median von B, muss in A nur noch der hintere Teil $[\frac{n}{2}...n]$ betrachtet werden und in B der vordere Teil $[1...\frac{n}{2}]$. Anderenfalls wird in A der vordere Teil $[1...\frac{n}{2}]$ betrachtet und in B der hintere Teil $[\frac{n}{2}...n]$. Dies erfolgt so lange rekursiv, bis beide zu betrachtende Arrays A und B die Länge 2 haben. Der Median wird dann vom Durchscnitt der Elemente gebildet, die in beiden Arrays am Index 0 liegen.

korrekt

b)

Ist der Median von A kleiner als der Median von B, kann davon ausgegangen werden, dass die Elemente, die vor dem Median von A kommen, nicht der gesuchte Median sein können. Analog kann in diesem Fall davon ausgegangen werden, dass die Elemente, die nach dem Median von B kommen, nicht der gesuchte Median sein können. Die selbe Beziehung gilt, falls der Median von B kleiner ist als der Median von A (dann nimmt B die Rolle von A und A die Rolle von B in obiger Beschreibung an). Nach endlich vielen Rekursionsschritten, werden alle Elemente, die nicht der gesuchte Median sein können "rausgestrichen". Übrig bleiben die Elemente, die den Median bilden.

Der Arrayzugriff hat eine konstante Laufzeit von $\mathcal{O}(1)$. In jedem Rekursionsschritt halbiert sich die zu durchsuchende Größe beider Arrays, wodurch sich je Array zur Bestimmung des Medians eine Laufzeit von $\mathcal{O}(\log n)$ ergibt. Folglich beträgt die Gesamtlaufzeit $\mathcal{O}(1) + \mathcal{O}(\log n) + \mathcal{O}(\log n) = \mathcal{O}(\log n)$.

korrekt

Aufgabe 4

z.Z: Die untere Schranke zur Erzeugung einer konvexen Hülle beträgt $\Omega(n \log n)$
Beweis:
Gegeben ist die Menge $M = \{(x_i, y_i) \epsilon \mathbb{Q}^2 | 1 \leq i \leq n\}$
Mit einem Algorithmus werden in der Zeit T(n) alle Eckpunkte der konvexen Hülle ausgegeben (p_i). Abschließend werden alle Eckpunkte linear durchlaufen, wobei mit dem kleinsten p_i gestartet wird.
Wäre der vorhandene Algorithmus schneller als $\mathcal{O}(n \log n)$, könnte man schneller sortieren

→ Widerspruch zur unteren Schranke der Sortieralgorithmen!
Daher benötigt ein Algorithmus zur Erstellung einer konvexen Hülle mindestens $\Omega(n \log n)$.

korrekt

Aufgabe 5

a)

Ein trivialer Algorithmus würde das Problem wie folgt lösen: Da das Array bereits sortiert ist, muss jedes Element n mit seinem Nachfolger $n+1$ verglichen werden. Das letzte Element hat keinen Nachfolger und muss nur mit seinem Vorgänger verglichen werden. Es sind also $n-1$ Vergleiche notwendig, womit der Algorithmus hat folgich eine Laufzeit von $\mathcal{O}(n-1)$ hat.

korrekt

Angenommen es gäbe einen analogen, besseren Algorithmus. Dieser müsste das Problem mit höchstens $n-2$ Vergleichen lösen können.

Es sei A ein Array mit n eindeutigen Werten. Da der bessere Algorithmus $n-2$ Vergleiche durchführt, existiert ein Element, das mit seinem Vorgänger nicht verglichen wurde. Entspricht der Wert des nicht verglichenen Elements dem seines Vorgängers (es sind nun nicht mehr alle Elemente des Arrays eindeutig), liefert der Algorithmus zurück, dass es keine Duplikate gibt, was nicht korrekt ist. Folglich gibt es keinen Algorithmus, der das Problem in weniger als $n-1$ Vergleichen lösen kann.

b)