Aufgabe 6

Der Hauptsatz der Laufzeitfunktionen ist bekanntlich folgendermaßen definiert:

> **Exkurs: Master-Theorem**
>
> $$T(n) = a \cdot T\left(\frac{n}{b}\right) + f(n)$$
>
> $a =$ Anzahl der Unterprobleme in der Rekursion
>
> $\frac{1}{b} =$ Teil des Originalproblems, welches wiederum durch alle Unterprobleme repräsentiert wird
>
> $f(n) =$ Kosten (Aufwand, Nebenkosten), die durch die Division des Problems und die Kombination der Teillösungen entstehen
>
> Dann gilt:
>
> **1. Fall:** $T(n) \in \Theta\left(n^{\log_b a}\right)$
>
> falls $f(n) \in \mathcal{O}\left(n^{\log_b a - \varepsilon}\right)$ für $\varepsilon > 0$
>
> **2. Fall:** $T(n) \in \Theta\left(n^{\log_b a} \cdot \log n\right)$
>
> falls $f(n) \in \Theta\left(n^{\log_b a}\right)$
>
> **3. Fall:** $T(n) \in \Theta(f(n))$
>
> falls $f(n) \in \Omega\left(n^{\log_b a + \varepsilon}\right)$ für $\varepsilon > 0$ und ebenfalls für ein c mit $0 < c < 1$ und alle hinreichend großen n gilt: $a \cdot f(\frac{n}{b}) \leq c \cdot f(n)$

(a) Betrachten Sie die folgende Methode `m` in Java, die initial mit `m(r, 0, r.length)` für das Array `r` aufgerufen wird. Geben Sie dazu eine Rekursionsgleichung $T(n)$ an, welche die Anzahl an Rechenschritten von `m` in Abhängigkeit von der Länge `n = r.length` berechnet.

```java
public static int m(int[] r, int lo, int hi) {
  if (lo < 8 || hi <= 10 || lo >= r.length || hi > r.length) {
    throw new IllegalArgumentException();
  }

  if (hi - lo == 1) {
    return r[lo];
  } else if (hi - lo == 2) {
    return Math.max(r[lo], r[lo + 1]); // O(1)
  } else {
    int s = (hi - lo) / 3;
    int x = m(r, lo, lo + s);
    int y = m(r, lo + s, lo + 2 * s);
    int z = m(r, lo + 2 * s, hi);
    return Math.max(Math.max(x, y), 2); // O(1)
  }
}
```

Code-Beispiel auf Github ansehen: `src/main/java/org/bschlangaul/examen/examen_66115/jahr_2018/fruehjahr/MasterTheorem.java`

Allgemeine Rekursionsgleichung:

$$T(n) = a \cdot T\left(\frac{n}{b}\right) + f(n)$$

Anzahl der rekursiven Aufrufe (a):

3

Anteil Verkleinerung des Problems (b):

um $\frac{1}{3}$ also $b = 3$

Laufzeit der rekursiven Funktion ($f(n)$):

$\mathcal{O}(1)$

Ergibt folgende Rekursionsgleichung:

$T(n) = 3 \cdot T\left(\frac{n}{3}\right) + \mathcal{O}(1)$

(b) Ordnen Sie die rekursive Funktion $T(n)$ aus (a) einem der drei Fälle des Mastertheorems zu und geben Sie die resultierende Zeitkomplexität an. Zeigen Sie dabei, dass die Voraussetzung des Falles erfüllt ist.

1. Fall: $f(n) \in \mathcal{O}\left(n^{\log_b a - \varepsilon}\right)$:

$f(n) \in \mathcal{O}\left(n^{\log_3 3 - \varepsilon}\right) = \mathcal{O}\left(n^{1-\varepsilon}\right) = \mathcal{O}\left(1\right)$ für $\varepsilon = 1$

2. Fall: $f(n) \in \Theta\left(n^{\log_b a}\right)$:

$f(n) \notin \Theta\left(n^{\log_3 3}\right) = \Theta\left(n^{1}\right)$

3. Fall: $f(n) \in \Omega\left(n^{\log_b a + \varepsilon}\right)$:

$f(n) \notin \Omega\left(n^{\log_3 3 + \varepsilon}\right) = \Omega\left(n^{1+\varepsilon}\right)$

Also: $T(n) \in \Theta\left(n^{\log_b a}\right)$