Algorithmen und Datenstrukturen

Aufgabe 4
Gegeben: die Heapify- Operation an der Wurzel eines Array-basierten k-nären Heaps mit n Elementen im worst-case [k log (n)] Schritte
(a)
Suche: Minimum der Funktion $f(x) = x*\log_x(n)$ für alle reellen, positiven x
1. Ableitung:

$$f'(x) = d/dx\ x*\log_x(n) = d/dx\ x*\log(n)/\log(x)$$
$$= [\log(n)\log(x) - \log(n)] / \log(x)^2$$

$0 = [\log(n)\log(x) - \log(n)] / \log(x)^2$
$\rightarrow$ x = e, da log(x)-1 = 0
Da die Funktion f(x) stetig ist und da folgendes gilt: lim (x->1) f(x) = ∞ = lim (x-> ∞) f(x), also die Funktion an den Grenzen des Definitionsbereiches gegen unendlich geht, muss an der Stelle x = e ein Minimum sein.
(b)
Da e~2.72 kein Element der natürlichen Zahlen ist, kann entweder k=2 oder k=3 die richtige Wahl sein.
f(2) = 2log2(n)
f(3) = 3log3(n)
Für alle n>1 ist f(2) > f(3), also ist k = 3 das Minimum in den natürlichen Zahlen.

l	k = 3 → $3\log_3(10^l)$	k = 2 → $2\log_2(10^l)$
1	7	7
2	13	14
3	19	20
4	26	27
5	32	34
6	38	40
7	44	47
8	51	54
9	57	60

Da es in der Tabelle um die Anzahl der Schritte im worst-case geht, wurden die Werte in Spalte 2 und 3 alle aufgerundet.
Man sieht, dass sich die beiden Laufzeiten sehr ähneln.

(c)
Bei der Heapify Operation auf den Gesamtheap muss bei jedem Knoten auf dessen binären Baum heapify angewendet werden. Dies benötigt jeweils $2\log_2(k)$ und ergibt eine gesamt Laufzeit von $2\log_2(k)*\log_k(n)=2\log_2(n)$

(d)
Beim binären Baum aus Präsenzaufgabe 3 müssen zwei Vertauschungen durchgeführt werden, sodass man (in level-order)

8	7	6	1	3	5	2

erhält.

Beim ternären Baum muss sogar nur eine Vertauschung durchgeführt werden, bis man

8	7	5	1	3	6	2

erhält.