Grundlagen Algorithmen & Datenstrukturen Blatt 07

Aufgabe 7.4.

a) $T(n) = \begin{cases} a & \text{falls } n=1 \\ c \cdot n + T(\frac{n}{2}) & \text{sonst} \end{cases}$

Da $a = 1 < b = 2$ gilt nach dem Master Theorem

$T(n) = \Theta(n)$

b) Ja, da dadurch Laufzeitklasse von $O(\log n)$ für binäre Suche zu $\Theta(n)$ wird.

c) $T(n) = \begin{cases} a & \text{falls } n=1 \\ c \cdot T(\frac{n}{2}) & \text{sonst} \end{cases}$

Da nicht auf jeder Stufe der Rekursion $c \cdot n$ Arbeit verrichtet wird, kann das Master-Theorem nicht angewendet werden.

Aus der Vorlesung ist bekannt, dass Binäre Suche $O(\log n)$ ist. Somit ist zu erwarten, dass:

$T(n) = c \cdot \log_2(n) + a$

Beweis der Rekursionsformel durch Induktion:

Induktionsbasis: $n = 1$

$T(n) = a = c \cdot \log_2(1) + a$ ✓

Induktionsschritt: Sei $n \in \mathbb{N}$ beliebig fixiert

Induktionsannahme:

$\forall n' \leq n$ mit $n = 2^k$ und $n' = 2^{k'}$

mit $k, k' \in \mathbb{N}$

gilt: $T(n) = c \cdot (\log_2(n) + a$

Induktionsbehauptung:

$T(2n) = c \cdot (\log_2(2n) + a$

Beweis:

$T(2n) = 2 \cdot T(n) + c$

nach Induktionsannahme

$= c \cdot (\log_2(n) + a + c$

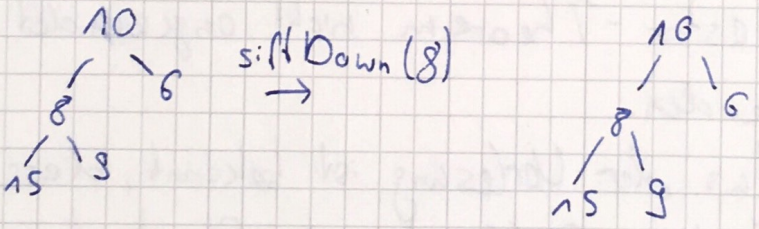
$= c \cdot ((\log_2(n) + 1) + a$

$= c \cdot ((\log_2(n) + (\log_2(2)) + a$

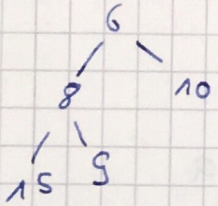
$= c \cdot (\log(2n) + a$ □

Aufgabe 7.5.

1. build:



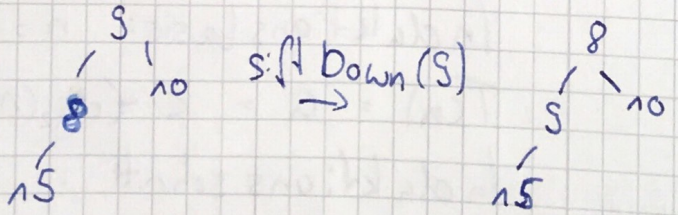
siftDown(10)



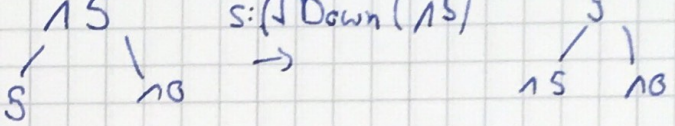
2: Sortieren:

swap (6, 9)

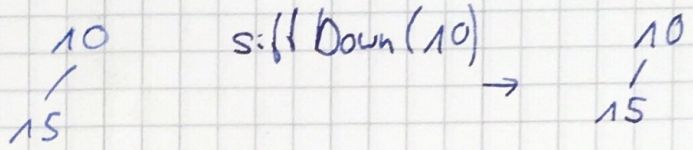
delete (6)



swap (8, 15)
delete (8)



swap (9, 10)
delete (9)



swap (10, 15)
delete (10)

15 siftDown(15) → 15

delete (15) / ⇒ Heap ist leer