HONORS ALGORITHMS | 2025-01-24

MERGE·SORT

$$A: \underbrace{\boxed{a_1} \boxed{\ } \boxed{\ }}_{A_1} \underbrace{\boxed{\ \cdots\ } \boxed{a_n}}_{A_2}$$

<u>If</u> $n=1$ <u>return</u> A

<u>else</u> $\text{MERGE}(\text{SORT}(A_1), \text{SORT}(A_2))$

recursive calls

$R(n)$: # comparisons

$T(n)$: total cost including bookkeeping

$$R(1) = 0$$

$$R(n) \le R\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + R\left(\left\lceil \frac{n}{2} \right\rceil\right) + n - 1$$

$$T(n) \le T\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + T\left(\left\lceil \frac{n}{2} \right\rceil\right) + O(n)$$

$R(n) \le n \log_2 n$ HW for $n=2^k$ CH for all n

$T(n) = O(n \log_2 n)$ HW for all n

p2

example

NAME	DOB	SS#	.

item

ID	key_1	…	key_t	$link_1$	…	$link_s$

LIST: NEXT, LAST, FIRST ← links

QUEUE: ENQUEUE(Q, x) $x = ID$

(ADD)

FIFO first-in-first-out

DEQUEUE(Q)

(DELETE)

(oldest item

dynamic data str.

STACK: PUSH(Q, x) FILO first-in-last-out

POP (Q)

(DELETE)

(youngest item)

implem: linked lists

Made with Goodnotes

PRIORITY QUEUE

INSERT(Q, x)

EXTRACT-MIN(Q)

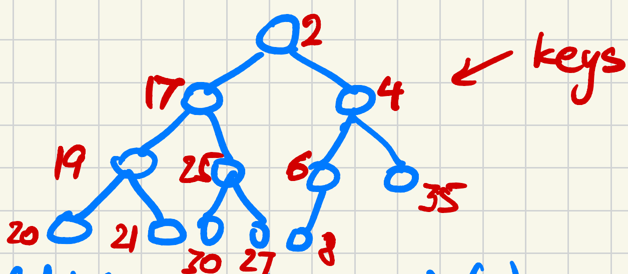
↑ item min(key(x))

AMORTIZED COMPLEXITY

overall cost of unpredictable sequence of requests

IMPLEMENTATION: HEAP

Link topology: balanced binary tree



12 nodes

item x held by node node(x)

HEAP RULE: key(parent(node(x))) ≤ key(x)

"heap-sorted data"

Repairing HEAP

If HEAP rule violated between

node x and one of its children, y

then swap $x \leftrightarrow y$

if both children y_1 y_2

then if $y_1 \le y_2$, swap $x \leftrightarrow y_1$

else swap $x \leftrightarrow y_2$

---

balanced binary tree of depth d

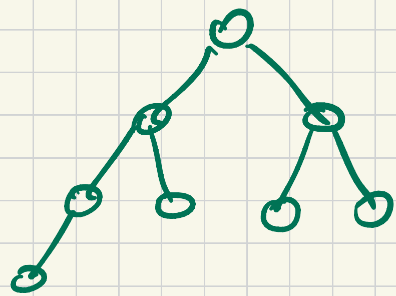
has

$$2^d \le n < 2^{d+1}$$

nodes

$$d \le \log_2 n < d+1$$

$d=3$



$$\therefore \boxed{d = \lfloor \log_2 n \rfloor} \le \log_2 n$$

cost of repair: $2 \cdot \log_2 n$ comparisons

EXTRACT-MIN :

<u>return</u> item(root)

move last item to <u>root</u>

<u>while</u> $\exists$ violation

fix it

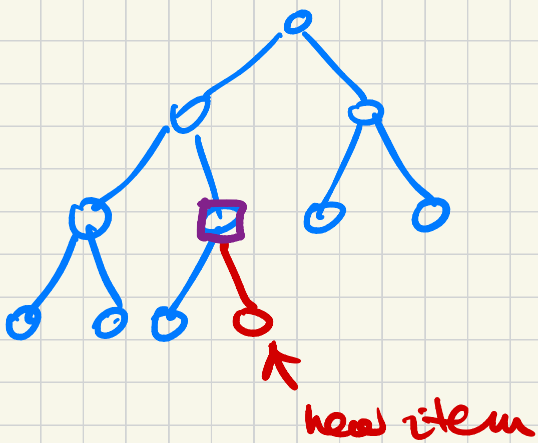
<u>bubble down</u>

INSERT

$\leq \log_2 n$

comparisons

<u>bubble up</u>



HEAPIFY

given n items

naive algorithm

$n \log_2 n$

START : empty heap

INSERT n times

$\#\text{compar.} \leq \log_2 1 + \log_2 2 + \cdots + \log_2 n = \log(n!) \sim n \log_2 n$

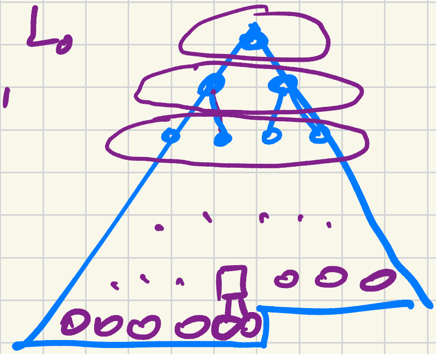
Thm HEAPIFY can be solved in $O(n)$ comparisons

layers L_0 L_1 L_2

LEMMA For $k \in \mathbb{N}_0$

$$|L_{d-k}| \leq \frac{n}{2^k}$$

L_d



Filling bottom-up, repairing:

$$\#\text{comparisons} \leq 2 \cdot \sum_{k=0}^{d} k \cdot |L_{d-k}| < 2n \underbrace{\sum_{k=0}^{\infty} \frac{k}{2^k}}_{2} = 4n$$

HEAP-SORT

1. HEAPIFY $O(n)$
2. WHILE $\neq \emptyset$ EXT-MIN

$2n \log_2 n$ comps