Heaps

October 26, 2023 8:47 AM

A set of n elements $H = \{H_1, H_2, \ldots, H_n\}$ is a max heap if $H_i \geq H_{2i}$ and $H_i \geq H_{2i+1}$ for $i \geq 1$.

1	2	3	4	5	6	7
11	7	3	4	6	5	1

$H =$ the array above ✗ $H_3 < H_6$.

$H =$ | 11 | 9 | 6 | 7 | 4 | 3 | 1 | ✓

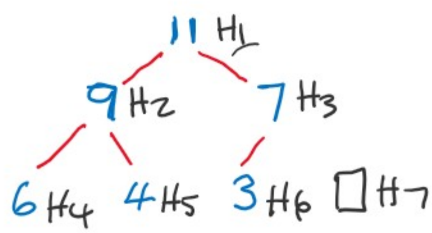
① If H is sorted in descending order then H is a heap.
② If H is a heap then H_1 is the maximum.

We represent a heap in an array with empty slots allowing for insertions and put $n = |H|$ in H_0. E.g.

0	1	2	3	4	5	6	7
6	11	9	7	6	4	3	—

$H :=$ Array(0..7);

We visualize H as a binary tree



① The children of node i are at positions $2i$ and $2i+1$

② The parent of a child at node i is at position $\lfloor i/2 \rfloor$.

A heap H with $n = |H|$ elements supports the following operations

$H :=$ MakeHeap(<, M) Create a new heap with space for M entries. $O(M)$

$X :=$ GetMax(H) $= H_1$ $O(1)$.

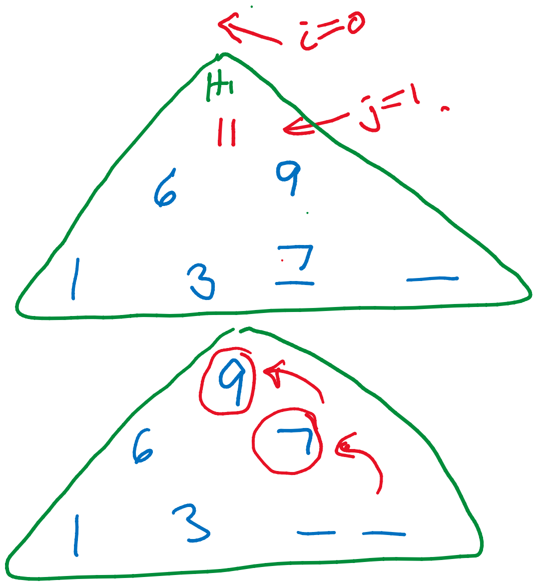
Insert(x, H) Insert x into H. $O(\log n)$ comparisons.

$X :=$ ExtractMax(H); Return H_1 and fix H. $O(\log n)$ comps.

$n :=$ Size(H) $= H_0$ $O(1)$

IsEmpty(H) $O(1)$.

Inserting into a heap.



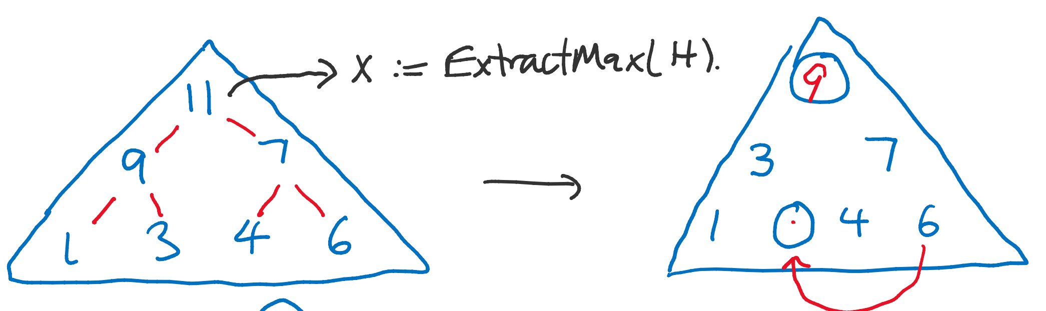
	H_1	2	3	4	5			
H=	5	9	6	7	1	3	_	_

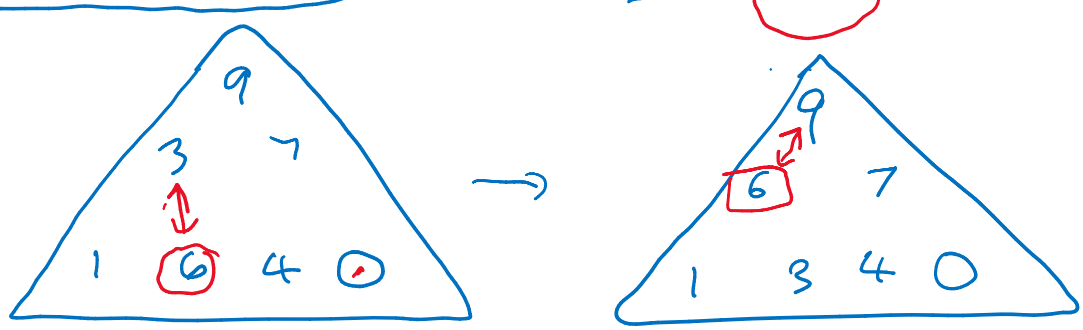
Insert (11, H).

```
Insert(x, H)
    n = H0 = 5
    j = n+1 = 6.
    i = ⌊j/2⌋ = 3
    while i>0 and x > Hi do
        H[j] = H[i];
        j = i
        i = ⌊i/2⌋
    end.
    H[j] = x;
    H[0] = n+1.
```

The # of comparisons is $\leq \lfloor \log_2(n+1) \rfloor \in O(\log n)$.

Extracting the largest element H_1 from the heap.





comparisons $\leq \underbrace{\lfloor \log_2 n \rfloor}_{\text{comparing}} + \underbrace{\lfloor \log_2 n \rfloor}_{\text{comparing}} = 2\lfloor \log_2 n \rfloor \in O(\log n)$.

comparisons $\leq \lfloor \log_2 n \rfloor + \lfloor \log_2 n \rfloor = 2\lfloor \log_2 n \rfloor \in O(\log n)$.

comparing two children — comparing with a parent

Average # comparisons $= \lfloor \log_2 n \rfloor + 2$

$\lfloor \log_2 n \rfloor$ ↑ comparing children; 2 ↑ comparing parents.