Binary Heaps

September 17, 2021 10:10 AM

A set of n elements $H = \{h_1, h_2, \dots, h_n\}$ is a $\overset{max}{\wedge}$ heap if
$H_i \geq H_{2i}$ and $H_i \geq H_{2i+1}$ for $i = 1, 2, \dots$

	1	2	3	4	5	6
$H =$	11	7	3	4	6	5

$3 = H_3 \not\geq H_6 = 6$

$H =$	11	9	6	7	4	3	1

1. If H is sorted in descending order it's a heap.
2. If H is a heap H_1 is the maximum element.

We represent a heap in an array with empty slots to allow for insertions. Put $n = |H|$ is H_0. E.g.

	0	1	2	3	4	5	6	7
$H =$	6	11	9	7	6	4	3	_

We visualize H as a binary tree.

1. The children of node i are at positions $2i$ and $2i+1$.
2. The parent of node i is $\lfloor i/2 \rfloor$

A heap H with $n = |H|$ elements supports the following operations which make a priority queue

$H :=$ MakeHeap$(<, m)$ $O(1)$

$x :=$ GetMax$(H) = H_1$ $O(1)$

$\to$ Insert(x, H) $O(\log n)$

$\to$ $x :=$ ExtractMax(H) $O(\log n)$

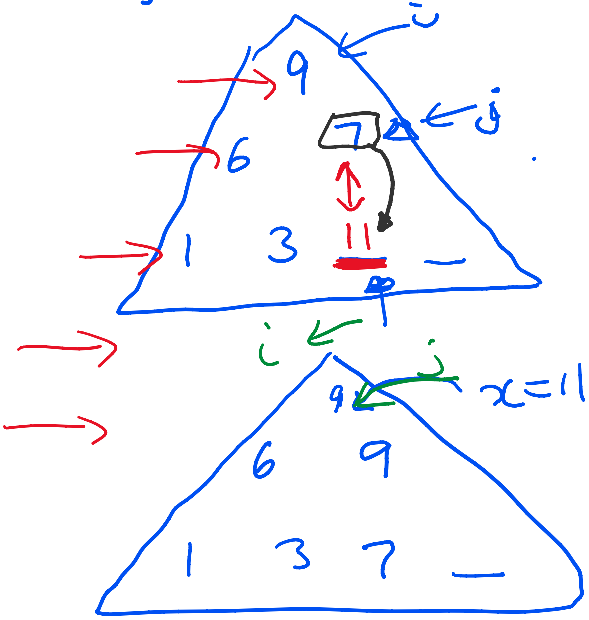
IsEmpty(H) $O(1)$

$n := |H| = H_0$ $O(1)$.

Inserting into a heap. Insert$(11, H)$ $H =$ | 5 | 9 | 6 | 7 | 1 | 3 |

Inserting into a heap.

Insert(11, H)   H = | 5 | 9 | 6 | 7 | 1 | 3 |

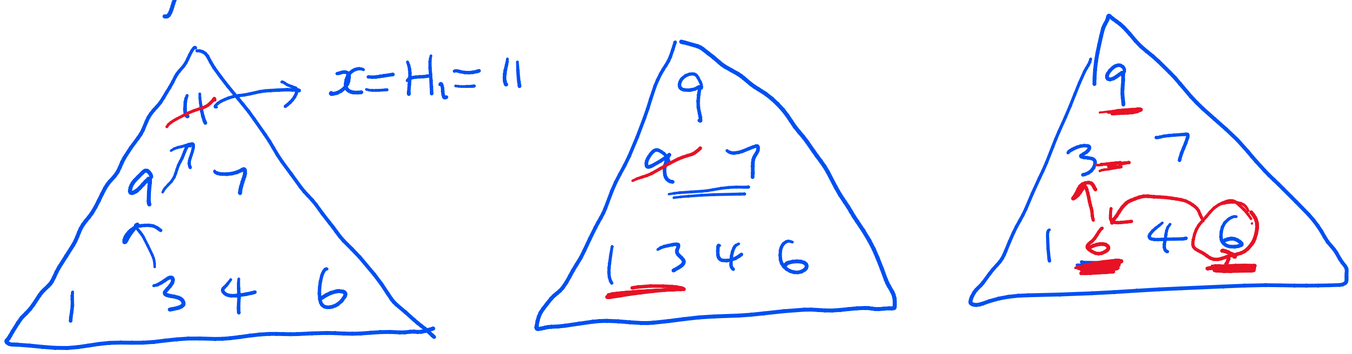


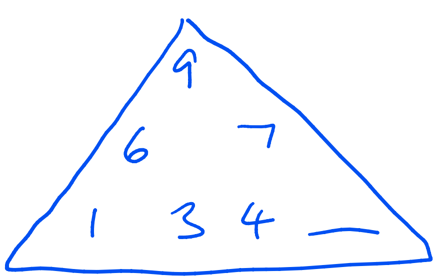

```
Insert(x,H) { int n,i,j;
    n = |H|; = 5
    j = n+1; = 6
    i = j/2; = 3
    while (i>0 && x > H[i]) {
        H[j] = H[i];
        j = i;
        i = i/2;
    }
    H[j] = x;
}
```

& moves
\# comparisons $= O(\log n)$

$\leq \lfloor 1 \cdot \log_2(n+1) \rfloor$

Extracting the largest element from a heap.



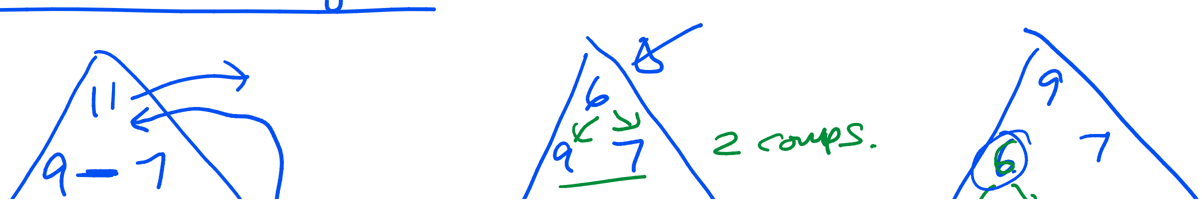


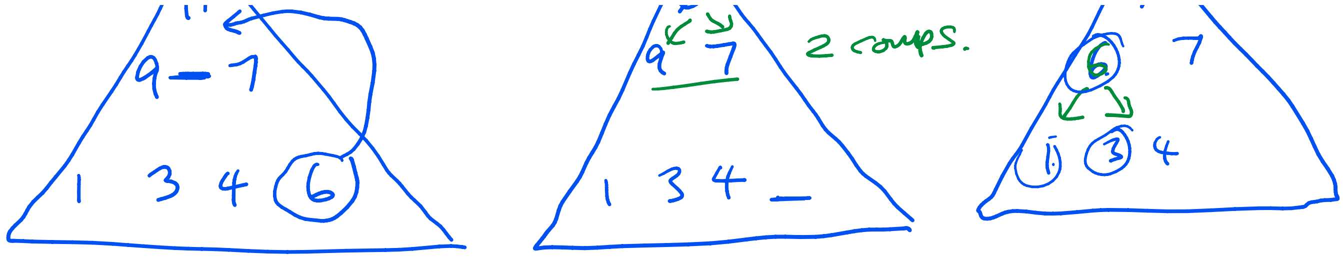
\# comparisons & moves

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

On average $= \lfloor \log_2 n \rfloor + 2.$

The bad extract algorithm.





On average one can show we do $\geq \lfloor \log_2 n \rfloor$ comps.