

## a) Insertion in Max Heap

### Algorithm:

1. Increase the heap size by 1.
2. Place the new element at the last position.
3. Set  $i = \text{last index}$ .
4. While  $i > 1$  and  $\text{heap}[i] > \text{heap}[\text{parent}(i)]$ :  
    Swap  $\text{heap}[i]$  with  $\text{heap}[\text{parent}(i)]$ .  
    Set  $i = \text{parent}(i)$ .
5. Stop when heap property is satisfied.

**Time Complexity:**  $O(\log n)$

---

## b) Deletion in Max Heap

### Algorithm:

1. Remove the root element (maximum).
2. Move the last element to the root position.
3. Reduce heap size by 1.
4. Set  $i = 1$ .
5. While  $i$  has children:  
    Find the larger child.  
    If  $\text{heap}[i] < \text{heap}[\text{larger\_child}]$ , swap them.  
    Set  $i = \text{larger\_child}$ .
6. Stop when heap property is satisfied.

**Time Complexity:**  $O(\log n)$