# Priority Queue

ADT similar to a queue

but

* enqueue requires prio
* dequeue is in order of prio

< is more important

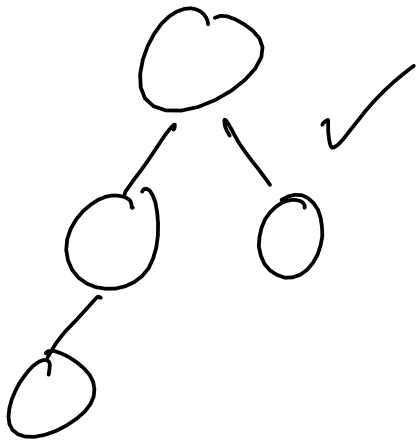
Min-Heap

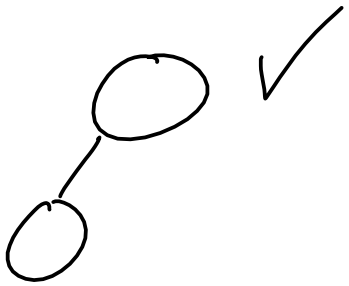
# Heap (Min)

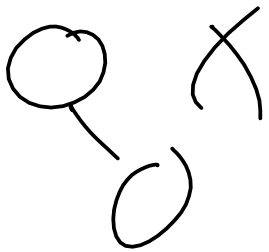
Kind of tree; $\forall$ nodes, children have $\geq$ values

Always complete

Complete tree: all levels except bottom are full
last level filled in left-to-right

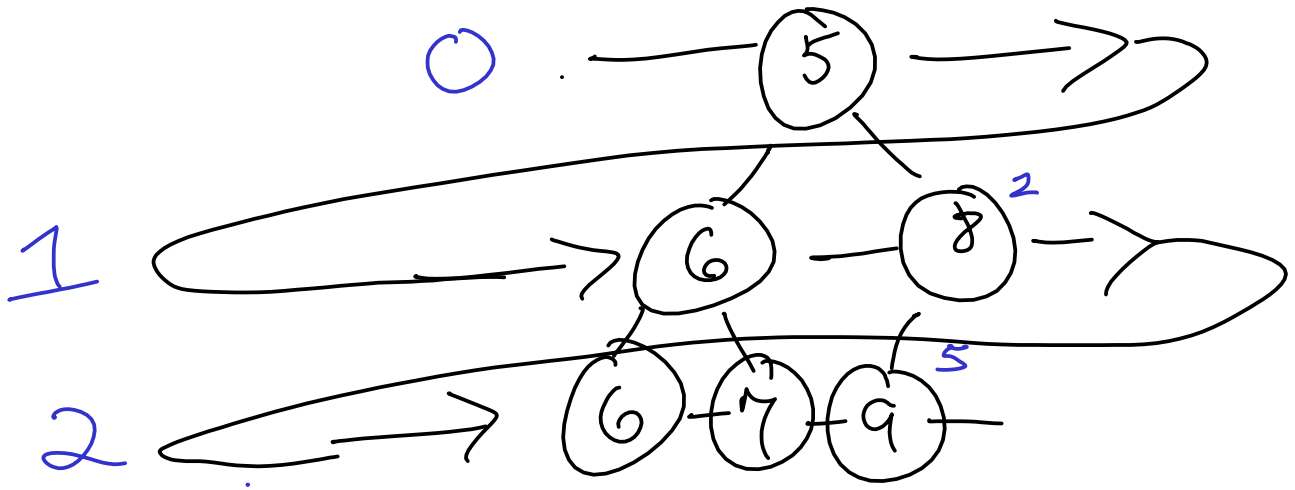






# Complete tree in array (list)

list matches level-order traversal




5, 6, 8, 6, 7, 9

isLeaf? i

is 2i+1 in bounds?

Left child of i is at pos $2i+1$

Right child: $2i+2$

Parent of i is at $(i-1)$ div 2 $\left\lfloor \frac{i-1}{2} \right\rfloor$

```
template <... P, ...T >
class MinHeap : public PQ {
    ⋮

    List<Pair<P, T>> contents;
```

O(log n)

Method enqueue(P prio, T value):

O(1) add (prio, value) to end of the list

O(log n) bubble_up (last index)

Time complexity? O(log n)

Method 'bubble_up(index):

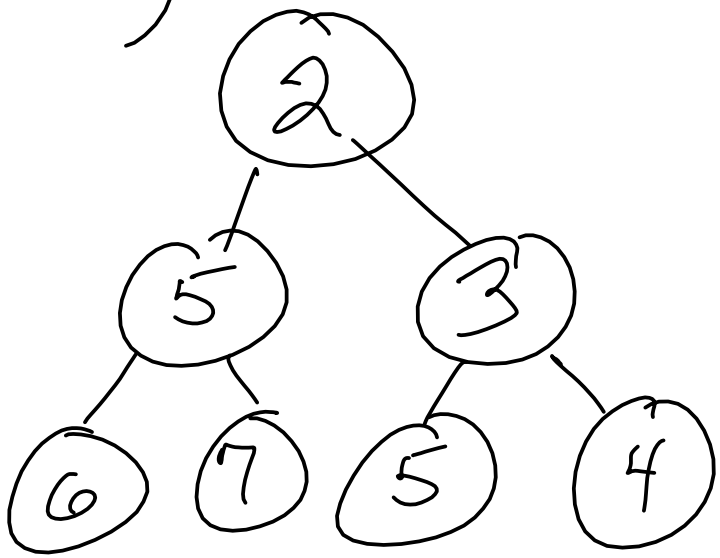
While index > 0 and

contents[(index-1)/2].first >

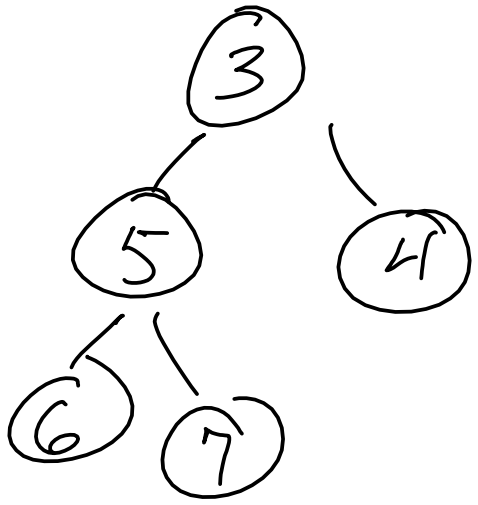
contents[index].first:

swap(contents[(index-1)/2], contents[index])

index = (index-1)/2



Method dequeue(): O(log n)




if size == 0:

value = contents[0].second

copy last to first pos (size--)

bubble_down(0)

Leaf

Left-only

Both

Method bubble_down(index):

If right_child(index) < size:

If prio(left_child(index)) < prio(right_child(index)):

If prio(index) > prio(left_child(index)):

swap index, left_child(index)

bubble_down(left_child(index))

Else...

| 3, 4, 3, 2, 7, 8, 6, 5 |
|---|

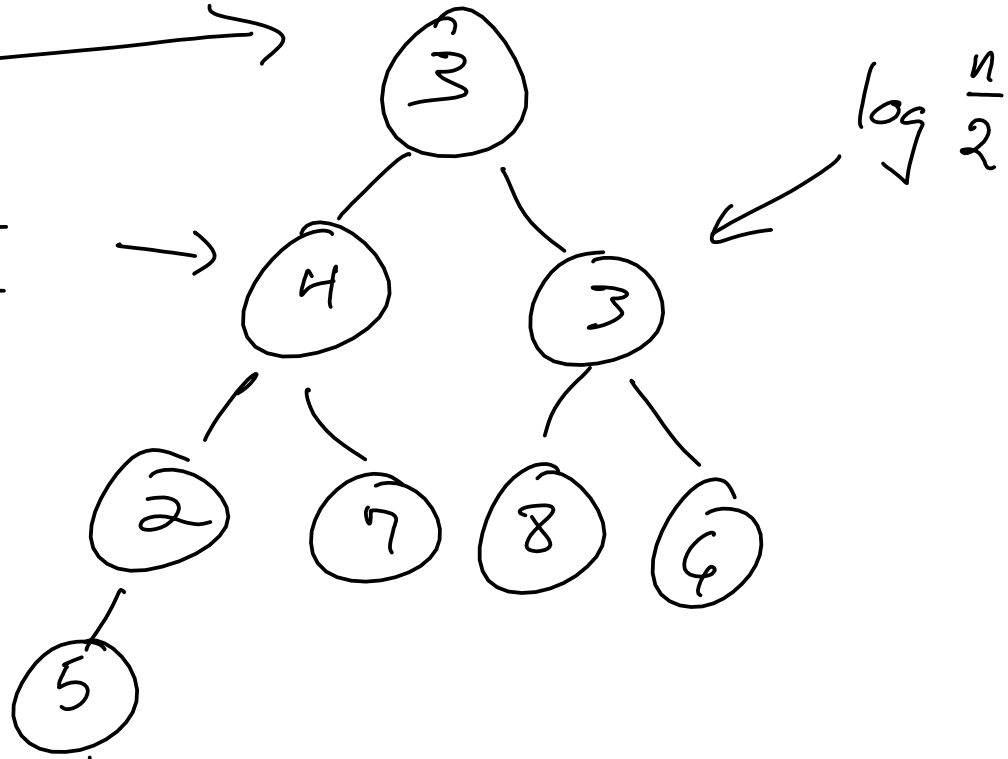
for each item, enqueue it! } $O(n \log n)$

## Heapify

| | |
|---|---|
| 1 | $\log n$ |
| 2 | $\log \frac{n}{2}$ |
| 4 | $\log \frac{n}{4}$ |
| ⋮ | ⋮ |
| $n/2$ | $O(1)$ |

$n$:

$\log n$ → $\log \frac{n}{2}$ → $\log \frac{n}{4}$




$$\sum_{i=0}^{\log n} 2^i \cdot \log \frac{1}{2^i} \Rightarrow O(n$$