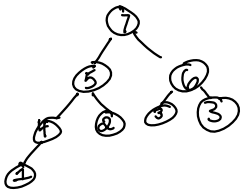
Heap Insert

* A heap is a complete binary tree where each node has priority ≥ its children.

parent(n): produces the index of the parent of node w/ index n
leftChild(n): produces the index of the left child of node w/ index n
rightChild(n): produces the index of the right child of node w/ index n
isLeaf(n): determines if node w/ index n is a leaf



amortized $O(\log n)$

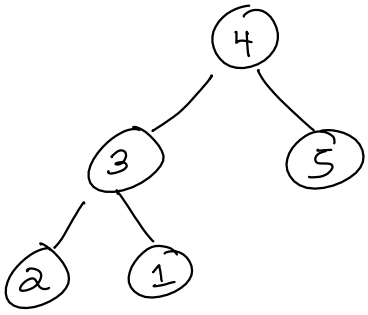
```
Method insert(P priority, V value):
    add (priority, value) to end of list
    bubbleUp(list size - 1)
End Method
```

$O(\log n)$

```
Method bubbleUp(int index):
    If index == 0:
        Return
    If list[index].prio > list[parent(index)].prio:
        Swap list[index] and list[parent(index)]
        bubbleUp(parent(index))
End Method
```

7	5	6	4	2	3	3	1

Heap Removal




```
Method remove():
    Swap list[0] with list[size-1]
    Remove the last element from list
    bubbleDown(0)
    Return value of node removed
End Method
```


```
Method bubbleDown(int index):
    Set left to leftChild(index)
    Set right to rightChild(index)
    If isLeaf(index):
        Return
    End If
    If right ≥ size:
        If list[index] < list[left]:
            Swap list[index] with
                list[left]
        End If
        Return
    End If
    If list[index] < list[left]
        or list[index] < list[right]:
            If list[left] < list[right]:
                Swap list[index]
                with list[right]
                bubbleDown(right)
            Else Swap list[index]
                with list[left]
                bubbleDown(left)
            End If
    End If
End Method
```

$O(\log n) = O(h)$