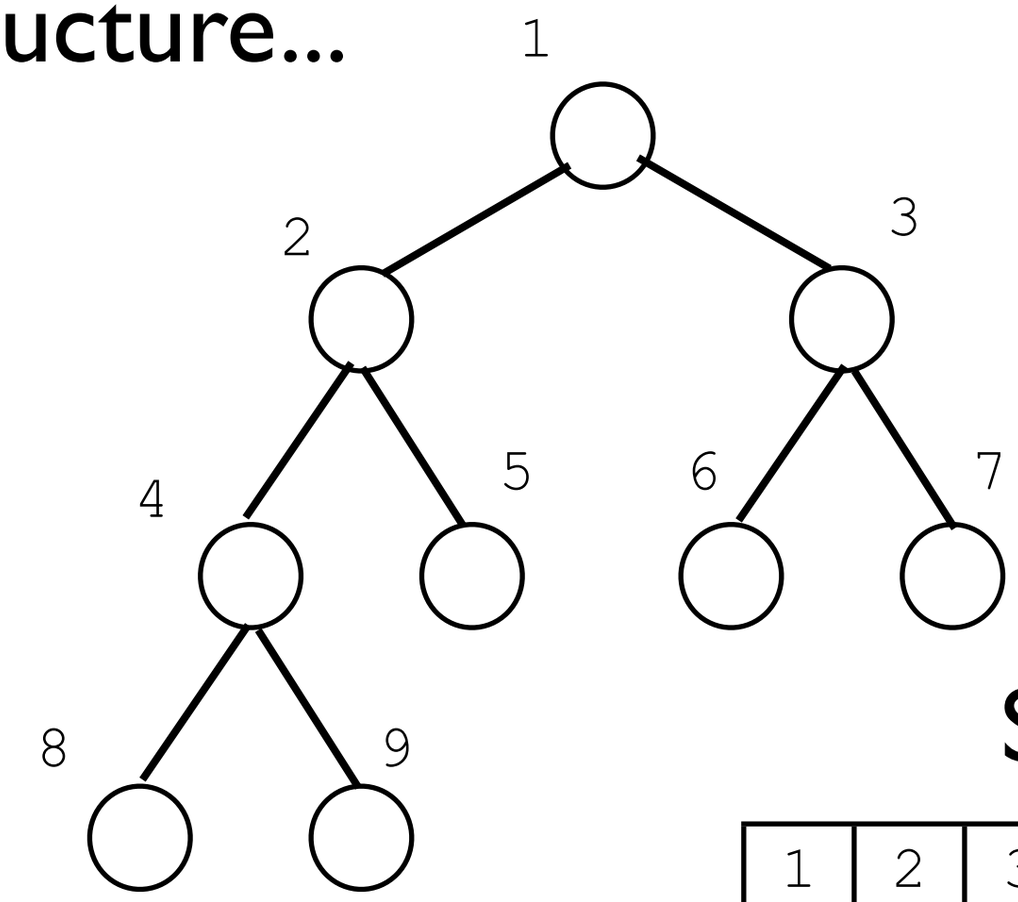
Heapsort

Max Heaps

- Typically stored in an array
- fast running time
- low memory needs (all swaps done in place)

Max Heaps

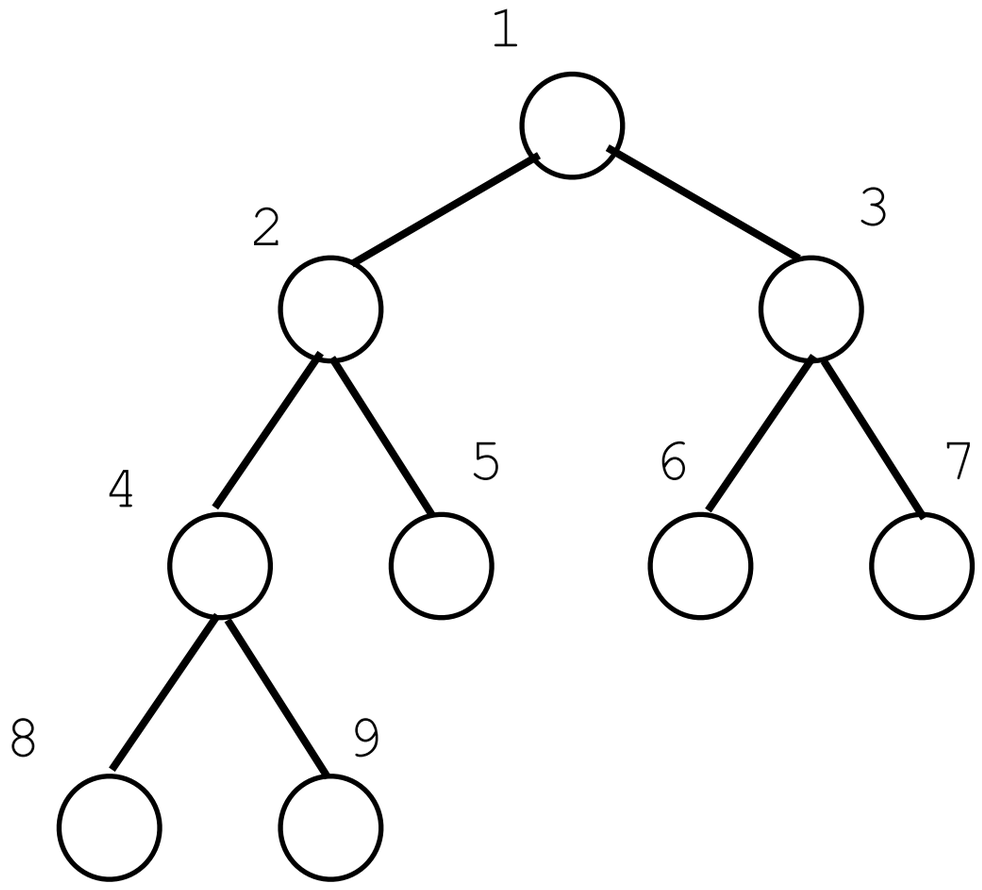
Tree structure...



Stored in an array

1	2	3	4	5	6	7	8	9

Max Heaps



Parent(i): A[⌊i/2⌋]

LeftChild(i): A[2*i]

Parent(i): A[2*i +1]

Stored in an array

1	2	3	4	5	6	7	8	9

MaxHeapify

```
MaxHeapify(A,i)
    if A[i] less than either of its children
        swap A[i] larger of its children, A[j]
        MaxHeapify(A,j)
```

- subroutine that maintains maxheap property if a value in the array has been changed
- the left and right subtrees of A[i] must be max heaps
- running time O(log n)

BuildMaxHeap

```
BuildMaxHeap(A)
    for i = length(A)/2 down to 1
        Maxheapify(A,i)
```

- subroutine that turns an array A into a max heap
- recall that maxheapify requires subtrees to be max heaps, hence starting with the parent of the last leaf of the deepest layer of the tree
- runs in O(n log n)

HeapSort

```
HeapSort(A)
    BuildMaxHeap(A)
    for i = length(A) down to 2
        swap A[i] and A[1]
        print value of A[i]
        set length of A to length(A)-1
        MaxHeapify(A,1)
```

- first builds a max heap
- takes advantage of the fact that the root always contains the largest value in the array
- prints reverse ordered list
- runs O (n log n)