**Recurrence Relations Running Time**

linear search: T(n)=T(n-1)+c O(n)

Insertion sort: T(n)=T(n-1)+n-1 O()

binary search tree: T(n)=1+log(n) O(log(n))

tree traversal: T(n)=2T(n/2)+1 O(n)

insert to max heap: T(n)= O(log(n))

extract from max heap O(log(n))

heapify: T(n) = ʘ(log(n))

heapsort: T(n) = T(nlog(n)) O (n \* lg (n))

**Heap Information**

Heap is excellent for a priority queue.

Min-heap is opposite of max heap

Height of heap is where d is the depth and first node is depth 0 (height)

Heap maximum is ʘ(1) and O(log(n)

Heap property: for every node i, other than the root, A[PARENT(i)] >= A[i]

Parent of I in array A[i] = floor(i/2)

Left child node of i Left[A[i]] = i\*2

Right child node of I right[A[i]] = (i\*2) + 1

Height of heap is: floor(log2n)

**Binary Tree Information**

A full binary tree one in which each node is either a leaf or has degree exactly 2.

Minimum height is: floor(log2n)