Binary Heaps & Priority Queues

- Priority Queue ADT
- Binary Heaps

Priority Queue ADT

- Enforces some kind of “priority” on the items
- For general purposes, call the highest priority the “maximum” (or the “minimum”)
- Operations:
  - add something to the PQ: insert
  - remove the item of highest priority: delMin (min PQ) or delMax (max PQ)

- Often, we store (key, value) pairs where the key is the “priority”
- **Key**: used to “rank” items—must be comparable

Priority Queue API

public class MaxPQ<Key extends Comparable<Key>>		
	MaxPQ()	*create a priority queue*
	MaxPQ(int max)	*create a priority queue of initial capacity* max
	MaxPQ(Key[] a)	*create a priority queue from the keys in* a[]
void	insert(Key v)	*insert a key into the priority queue*
Key	max()	*return the largest key*
Key	delMax()	*return and remove the largest key*
boolean	isEmpty()	*is the priority queue empty?*
int	size()	*number of keys in the priority queue*

API for a generic priority queue

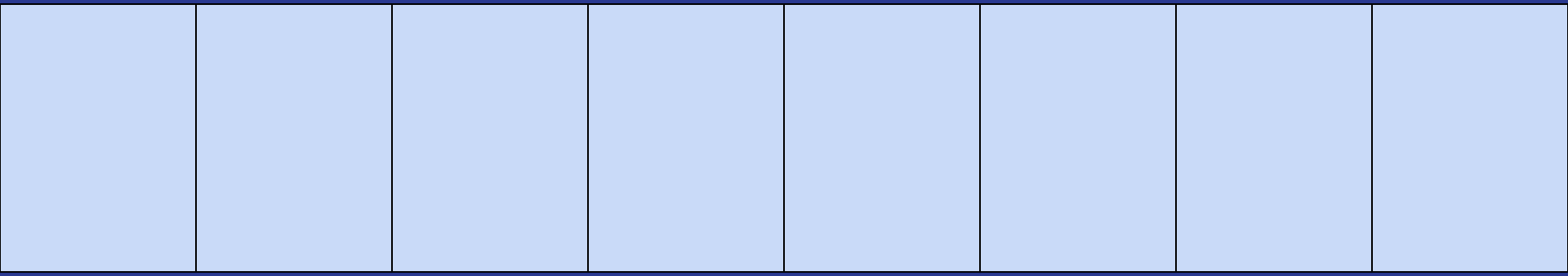
NOTE: a **MinPQ** would be basically the same except you would have **min**() to get the smallest key and **delMin**() to return and remove the smallest key.

Picture from [1]

How would you implement a Priority Queue?

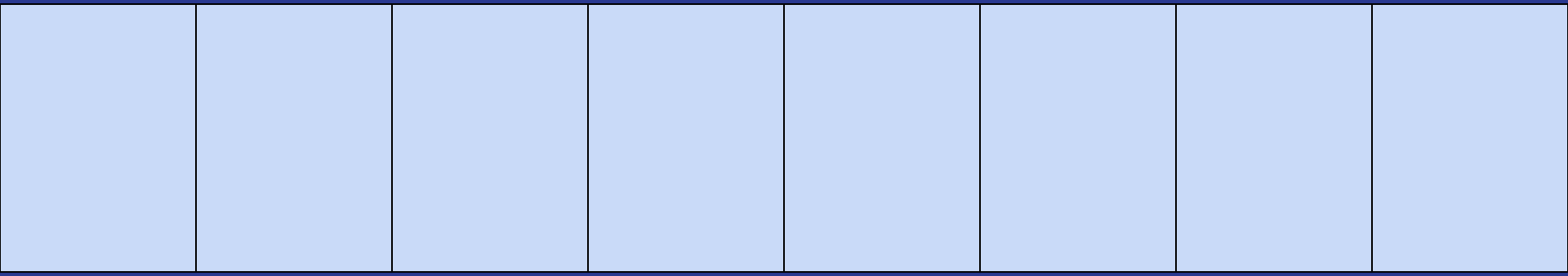
Unordered Array: Min PQ

insert: 23, 10, 17, 28, 34, 89, 22, 10
delMin, delMin



Ordered Array: Min PQ

insert: 23, 10, 17, 28, 34, 89, 22, 10
delMin, delMin



Key Operations: **insert** & **delMin**

Option 1

Unordered array

Worst-case runtime analysis:

- **insert** O(1)
- **delMin** O(N)

Option 2

Ordered array

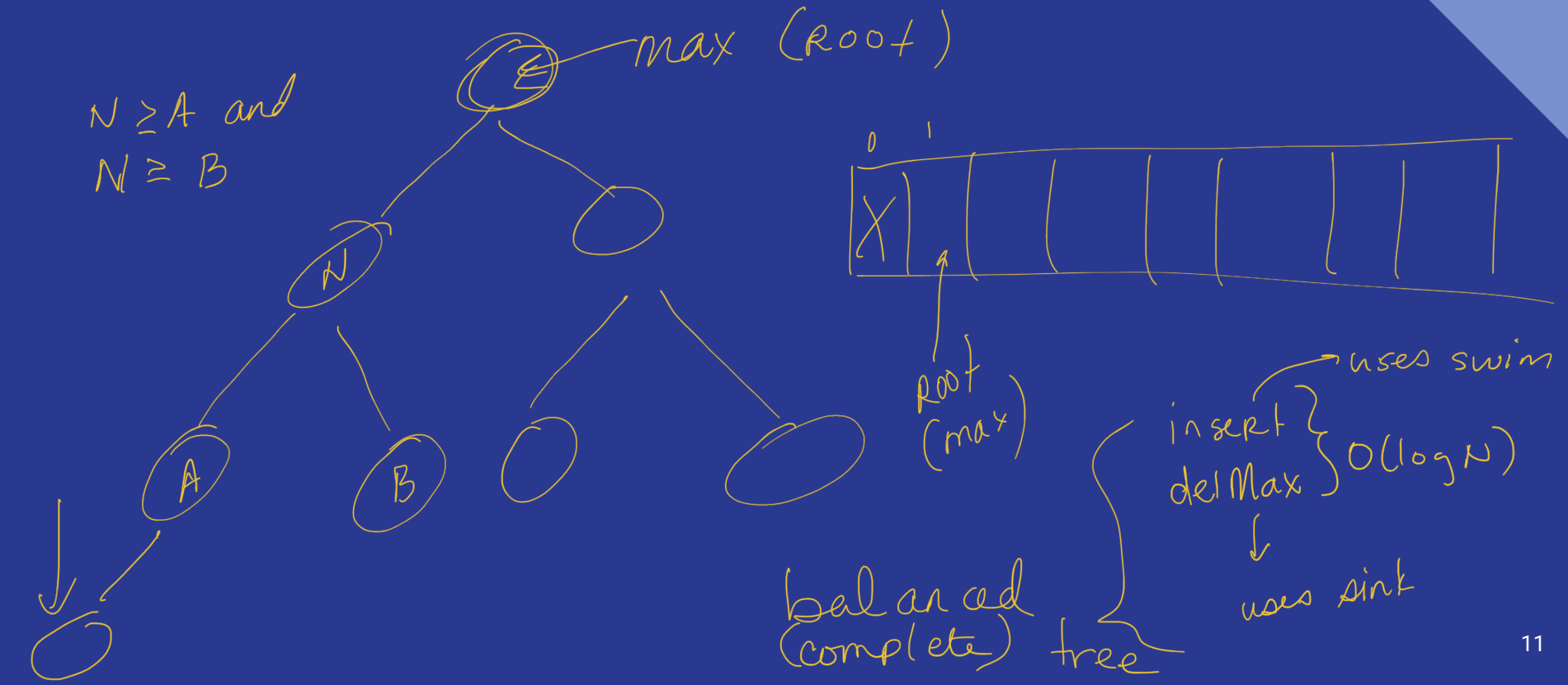
Worst-case runtime analysis:

- **insert** O(N)
- **delMin** O(1)

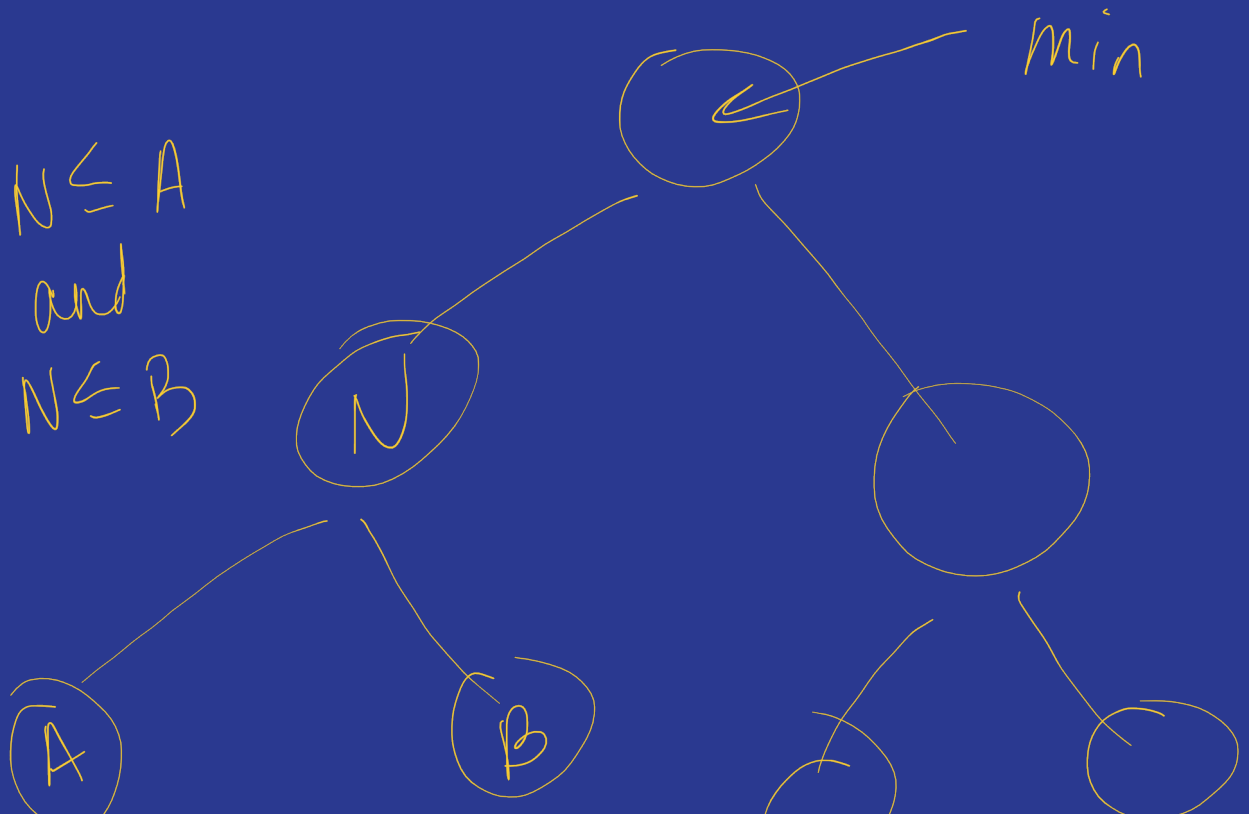
Imagine a company...

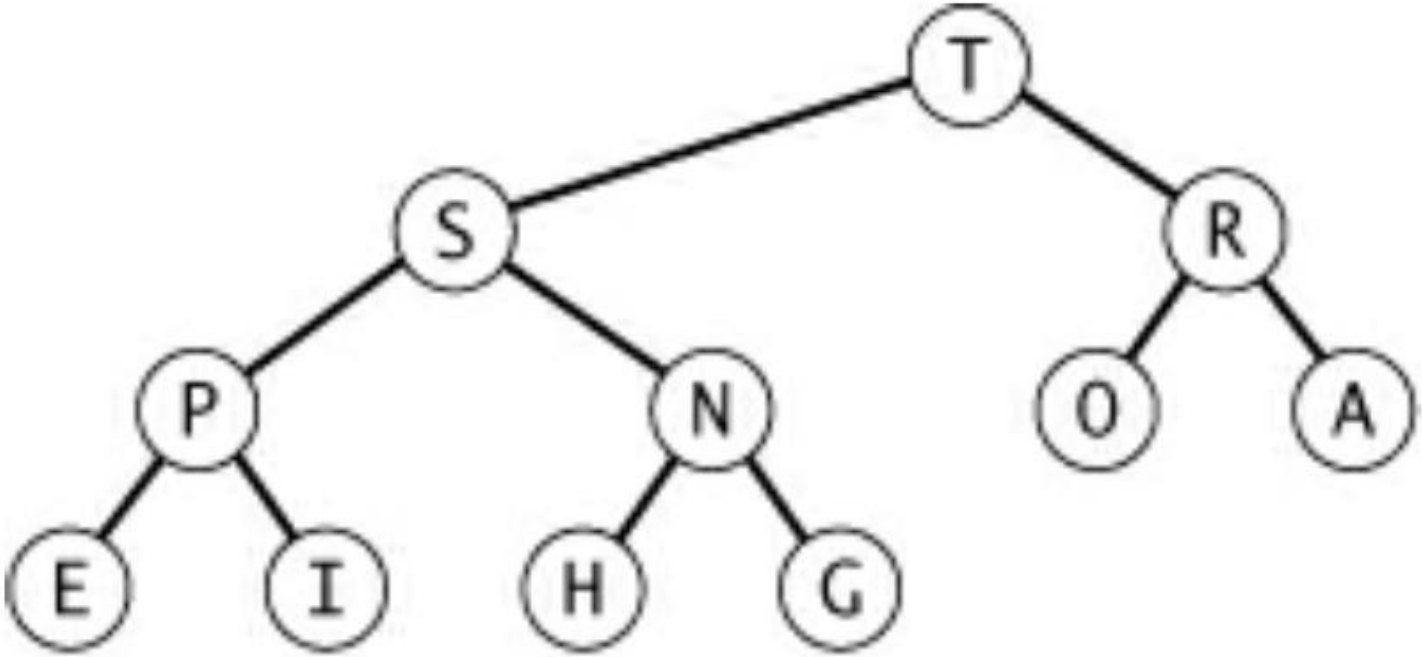
- New people are always hired at the lowest level with an open position
- Only the CEO ever quits
- Final ranking is determined by an IQ test where any given employee can only be managed by another employee with a higher IQ.
- Any given person can only directly manage two people.

So...what's a max binary heap?



So...what's a min binary heap?

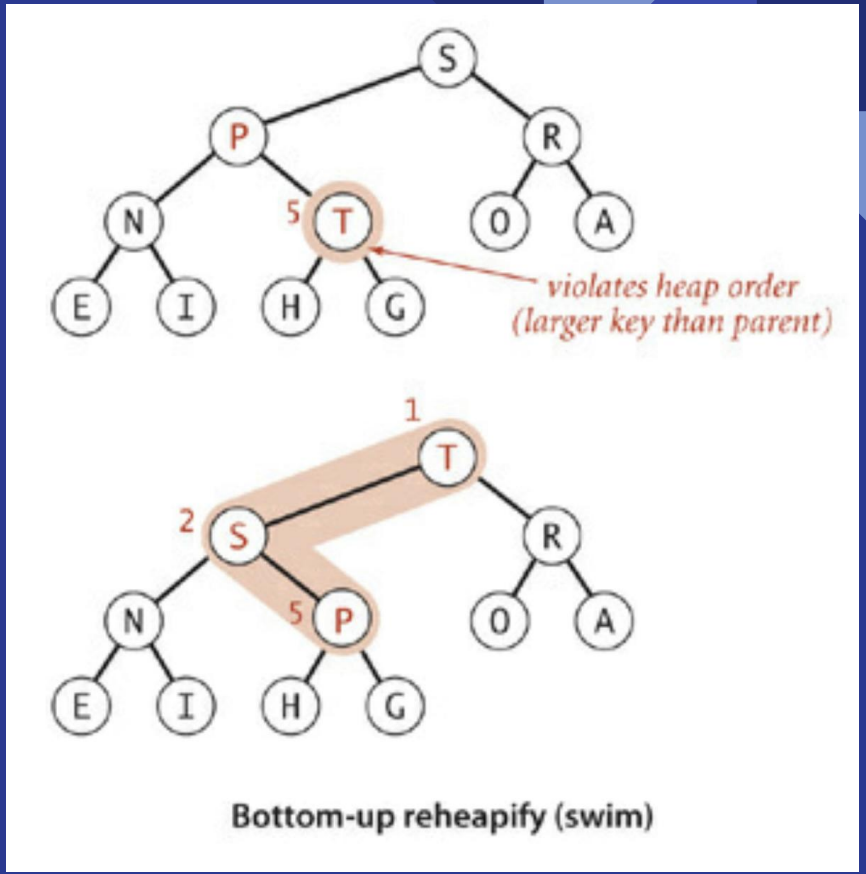




A heap-ordered complete binary tree

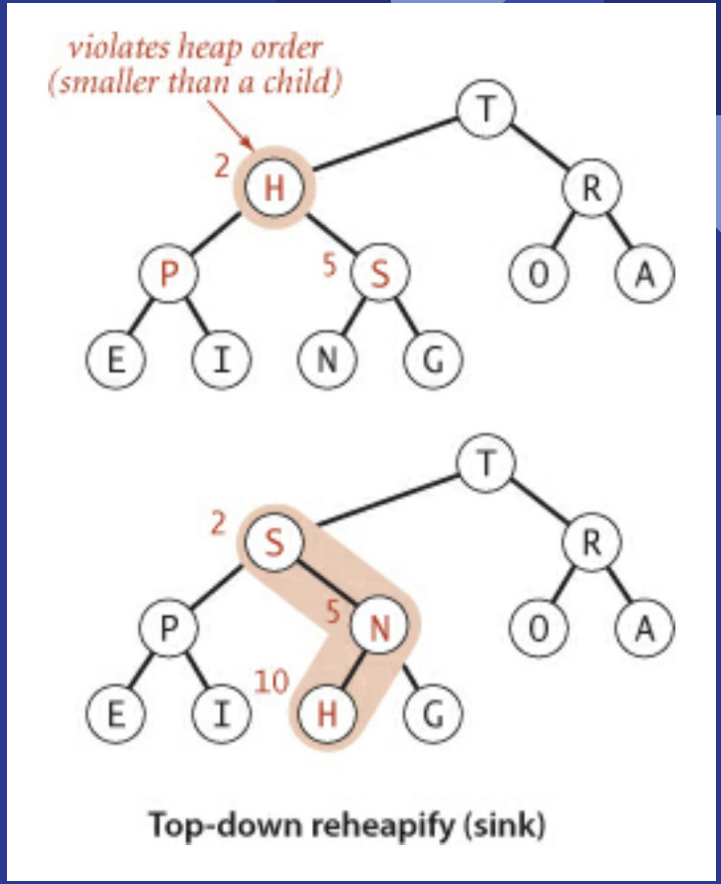
Picture from [1]

Swimming up...

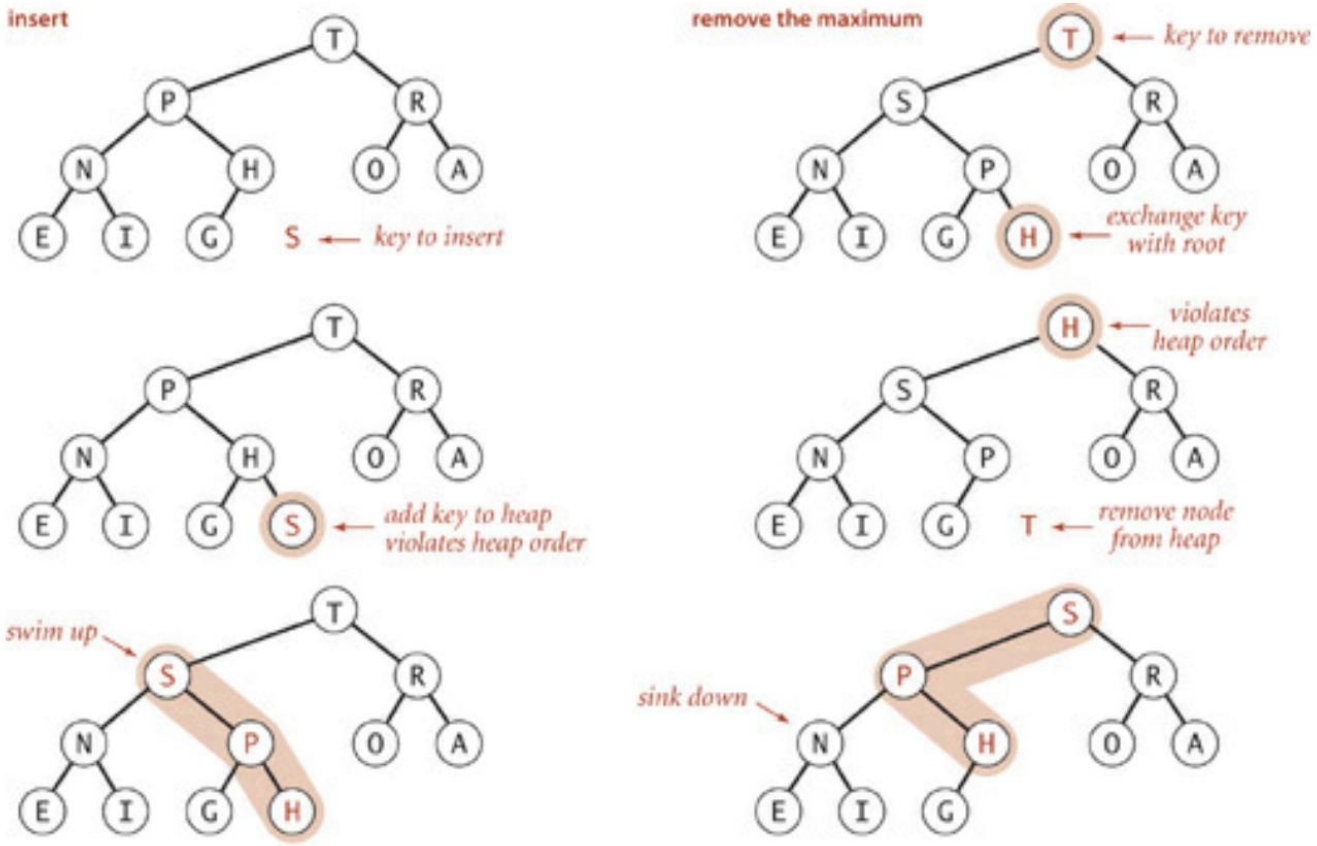


Picture from [1]

Sinking down...



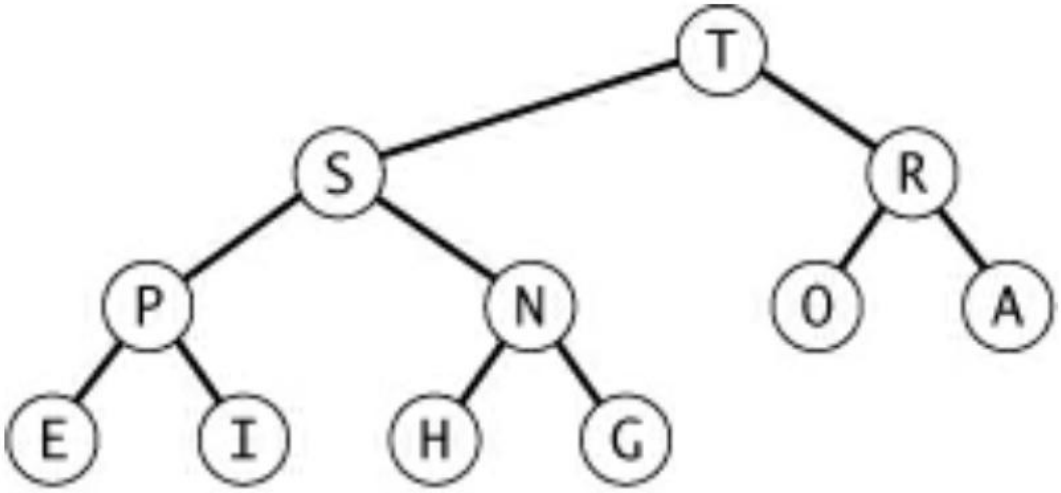
Picture from [1]



Heap operations

Picture from [1]

Why is it O(logN) for the operations?



A heap-ordered complete binary tree

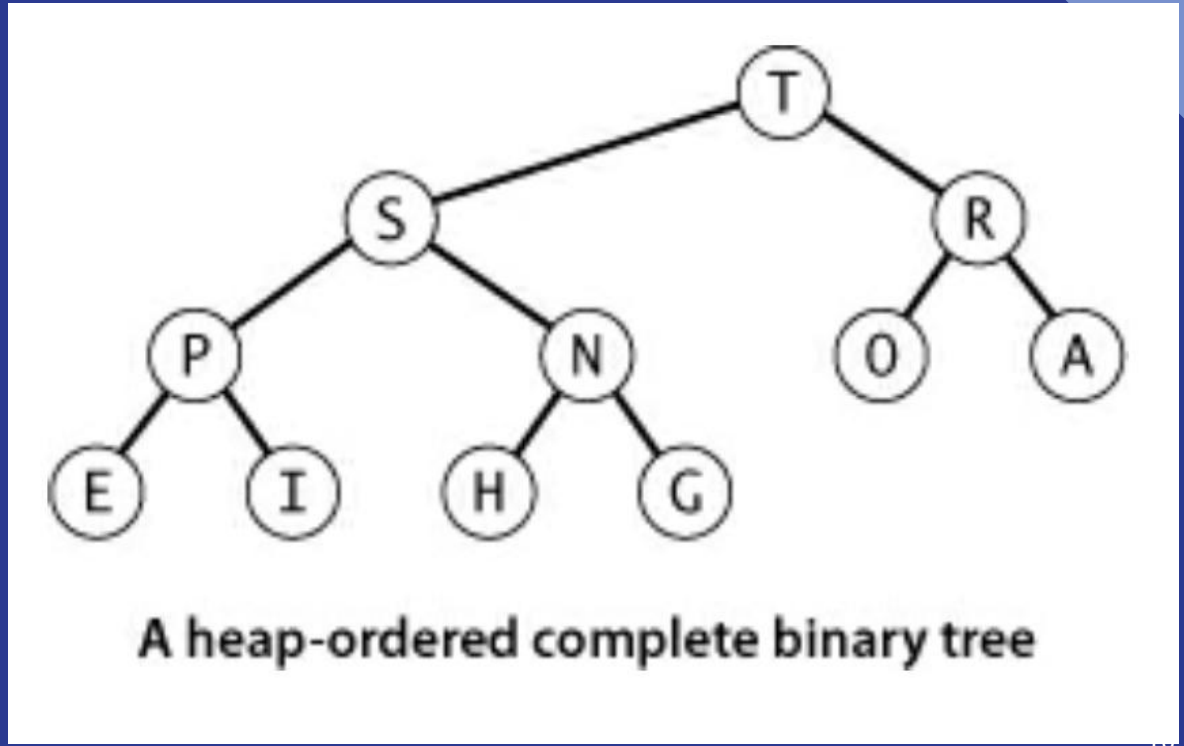
O(height of the tree)

||

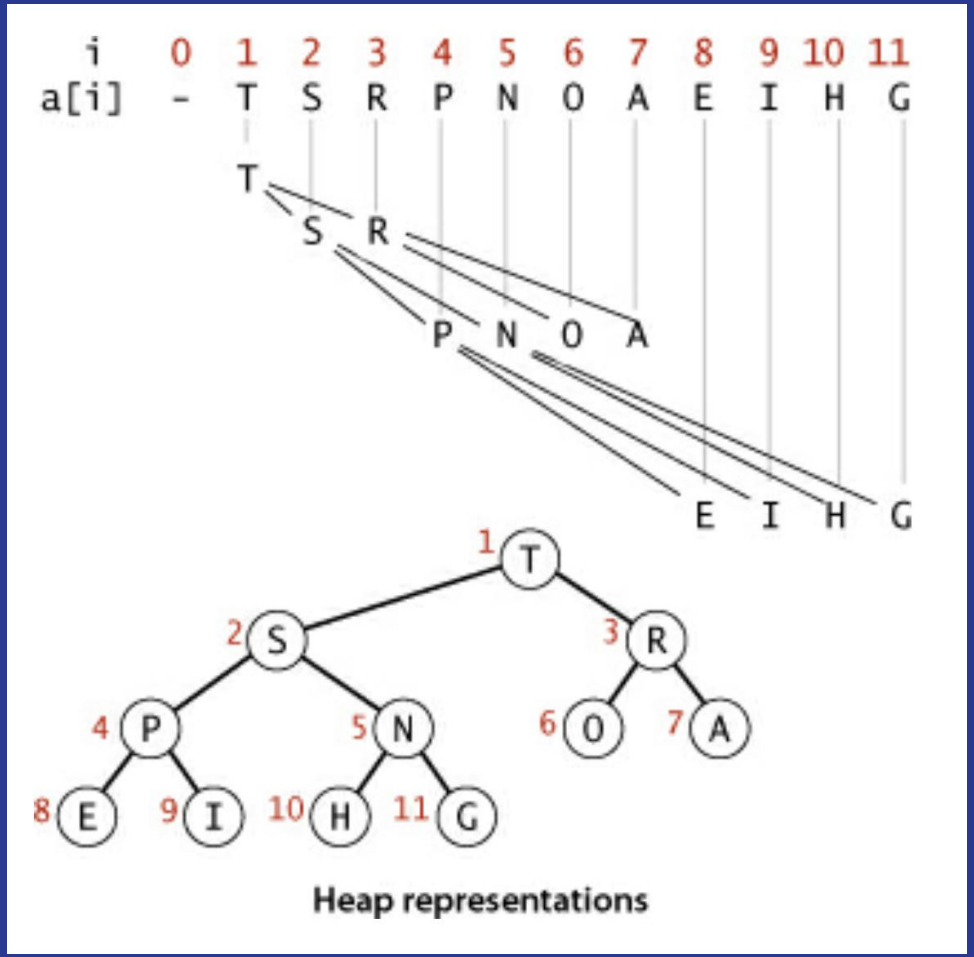
O(logN) b/c it is balanced

Pictures from [1]

How would you implement a binary heap?



Picture from [1]

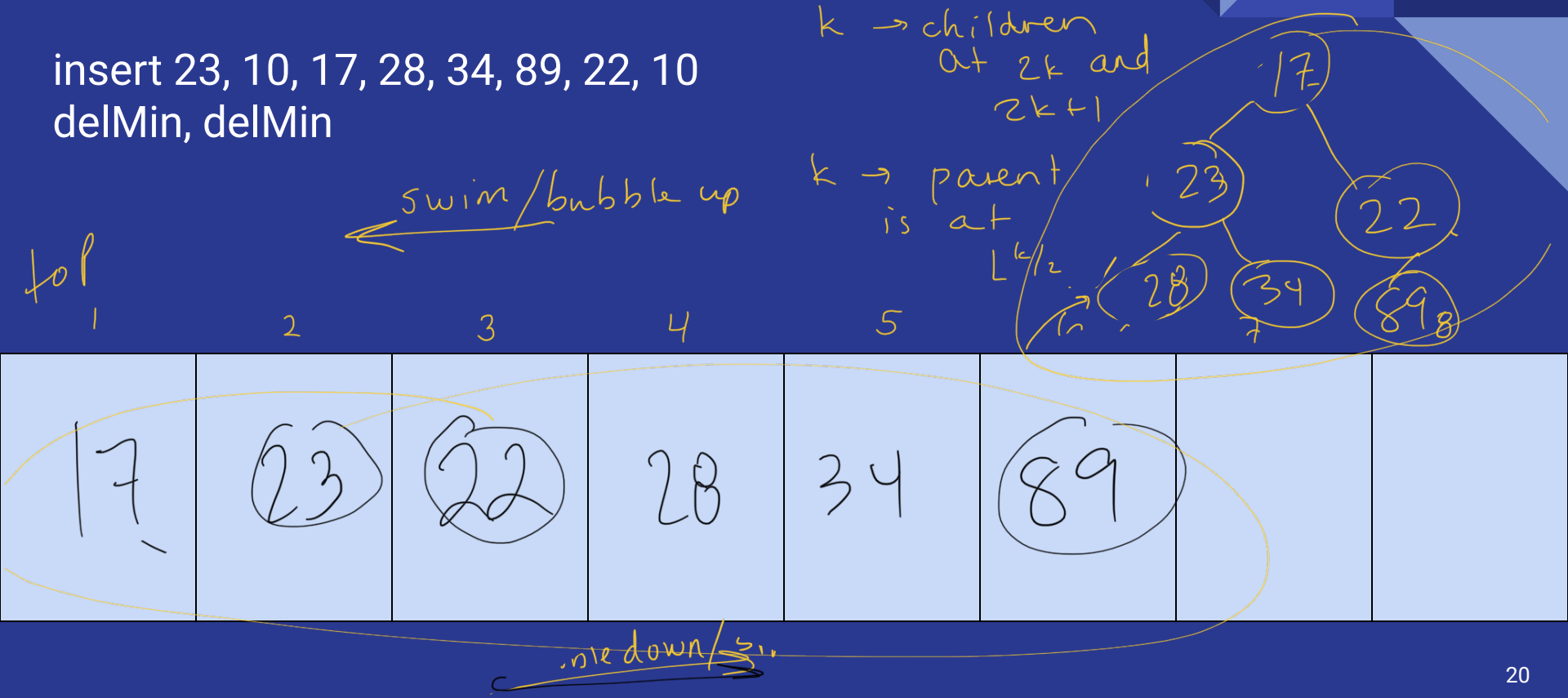


Heap representations

Picture from [1]

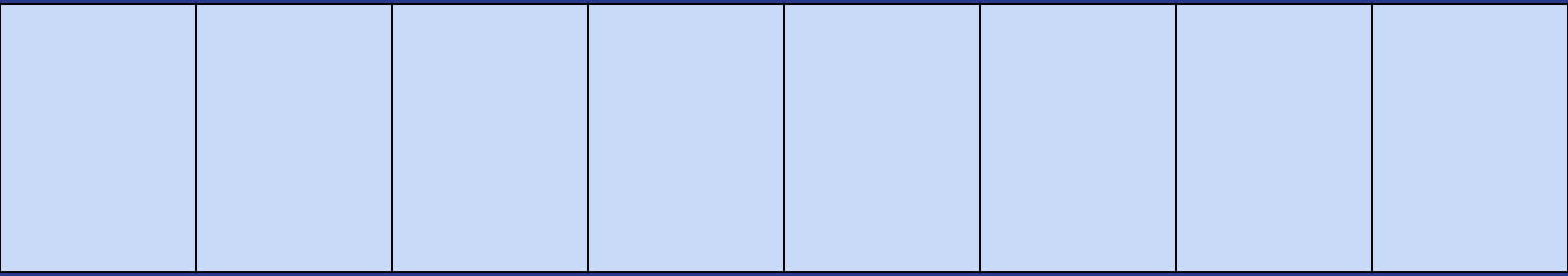
Array-based Binary Heap: Min PQ

insert 23, 10, 17, 28, 34, 89, 22, 10
delMin, delMin



Array-based Binary Heap: Max PQ

insert 23, 10, 17, 28, 34, 89, 22, 10
delMax, delMax



Summary and Clarifications

- A binary heap is not the same thing as a priority queue. A priority queue is an ADT that stores and processes data according to some kind of priority ranking. A binary heap (which can be max-ordered or min-ordered) is a common way of implementing a PQ because it is a structure that does *insert* and *delMax* (or *delMin*) efficiently.
- A binary heap, in turn, can be implemented with nodes and pointers or with an array.
- You can store anything in a PQ as long as you can assign some kind of ranking to it. Often, the “priority” is treated as the key in a key-value pair. e.g. (IQ, Employee), but the “key” and the “value” could also be the same.
- A max-ordered binary heap is built to do two things efficiently: insert & delMax.
- A min-ordered binary heap is built to do two things efficiently: insert & delMin.
- The binary heap structure is not related to the memory heap.

References

[1] *Algorithms, Fourth Edition;* Robert Sedgewick and Kevin Wayne (and associated slides)
[2] Book slides from Goodrich and Tamassia