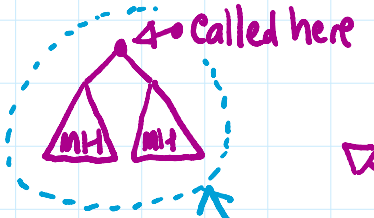
CMSC351-2023-10-09

Monday, October 9, 2023 8:35 AM

HEAPSORT – THE SEQUEL!



(1) Reminders:

(A) Max Heap: Every key is ≥ its child keys

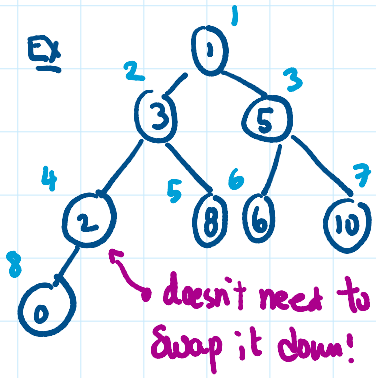
(B) Maxheapify: Only ever called on a node for which the child subtrees are max heaps. If key < either child key, swap w/ the largest child key then call recursively on that child node

Result: whole thing is a max heap

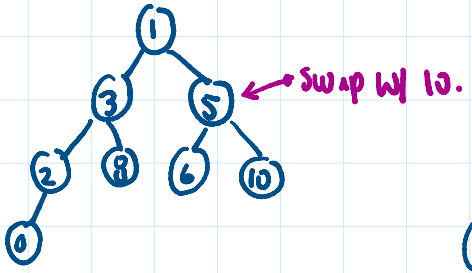
(C) Convert to max heap: Runs maxheapify on nodes w/ indices $\lfloor n/2 \rfloor, \ldots, 2, 1$. Converts a complete binary tree to a max heap.

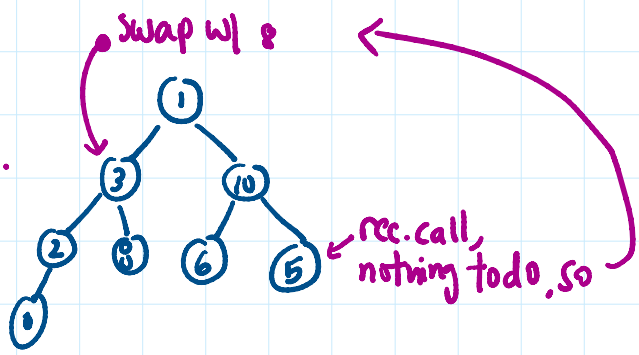
↳ last node w/ children!

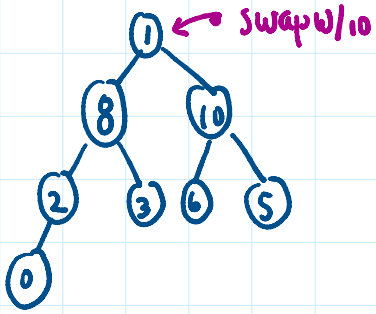
Ex

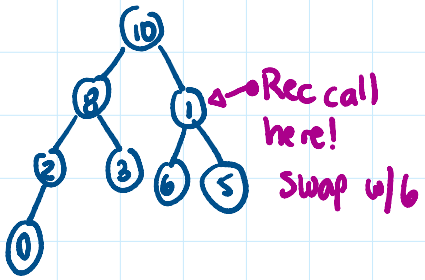


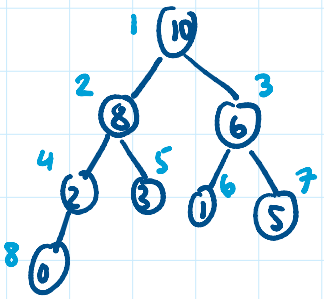
Start at index $\lfloor 8/2 \rfloor = 4$











MAX HEAP!

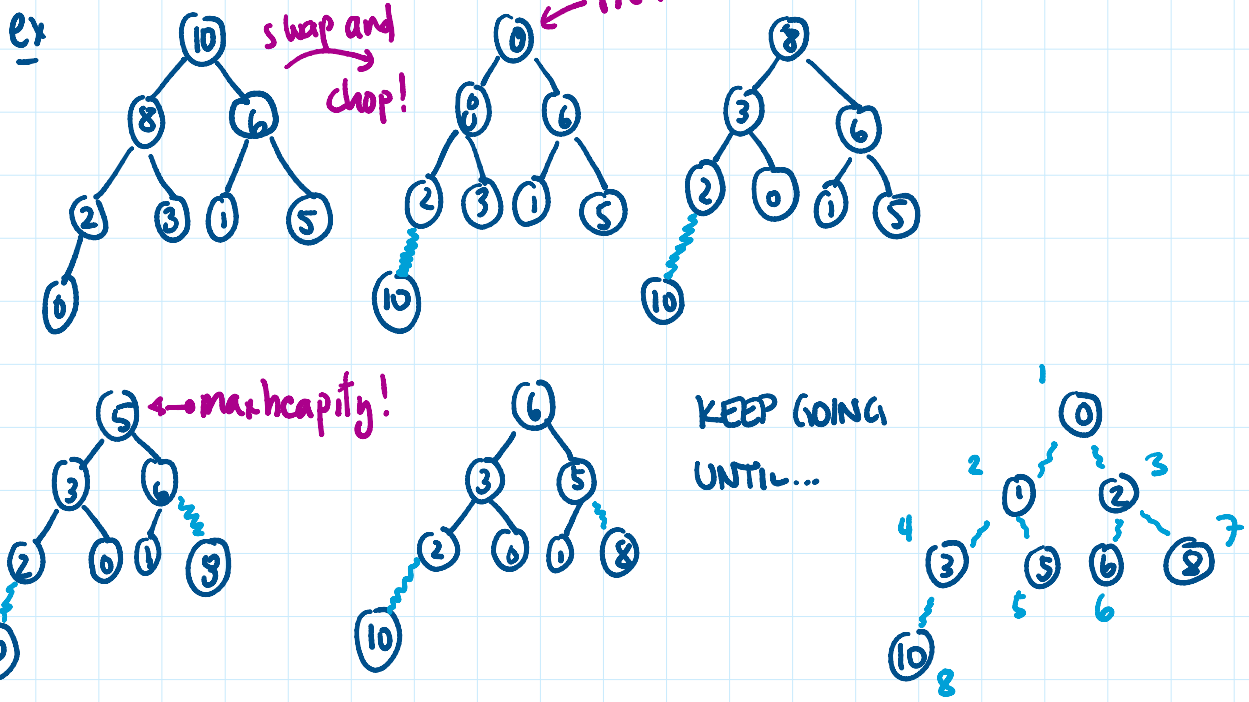
(2) Worst-case time is $O(n \lg n)$.

(2) Heap sort: Given a 1-indexed list, we first convert to max heap as above! Then:

- swap last key w/ root key and "chop off" the last key.

- swap last key w/ root key and "chop off" the last key.
  means: leave it in the list/heap but ignore it!
  Call maxheapify on root.
- Repeat until...

ex



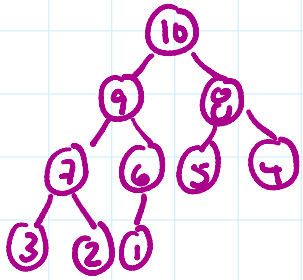
now they are sorted according to index!

③ Stuff

- Time Complexity. Worst-Case: Takes O(nlgn) to convert to max heap.
  Then, when we call maxheapify on the root
  in the wc it swaps down to the leaves.
  This is Θ(lg(# nodes in the tree, not nec n bc we're chopping nodes off!))
  However its O(lg n).
  Done n times. So another O(nlgn)
  Total: O(nlgn)
- Best-Case: Ideas.
  - If the complete bin. tree is a sorted list it seems nice but convert to maxheap screws it up! It conv. to a

but convert to maxheap screws it up! It conv. to a max heap at $O(n \lg n)$ then does the sorting part at $O(n \lg n)$.

- If it's in reverse sorted order?
  When we CTMH the calls to MH do nothing - each call to MH is $\Theta(1)$ so CTMH is $\Theta(n)$.
  But the chop/swap/maxheapify will still have to move each key down from the root that's all still $O(n \lg n)$. So $O(n \lg n)$ ∴



- Aha, if all keys are equal then CTMH is $\Theta(n)$ and the chop/swap/maxheapify will be $\Theta(n)$ too blc calls to MH will be $\Theta(1)$ each. Thus $\Theta(n)$ is the best case!

- In-Place? Yes!
- Aux Space? $\Theta(1)$ for loop and swap vars.
- Stable? No. Can you think of an example?

③ Pseudocode: See Later!