COP3502 4/4/22

1. Video - Trees
2. △ Syllabus
3. Quiz 4 - Friday
4. Heaps - P6 data structure

1st hand very to learn these ideas
- closer to something real world
- rules "messy"
- ~~Then~~ sizeable example where you have to be very careful with dynamically allocated memory.

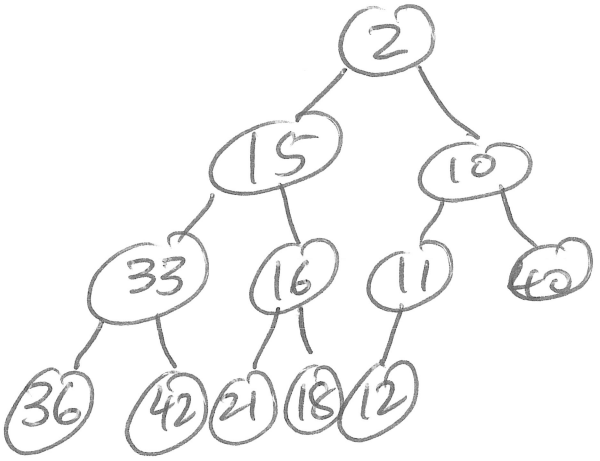
Binary Heap

O(log n)
- Insert items
- Delete Min

} 2 operations support.

runtimes similar to a Balanced BST.
Maybe a little simpler implementation

Binary Heap Drawing



for any node, everything in its subtree is greater than it!
↳ heap node property.

added nodes top to bottom left to right → complete binary tree

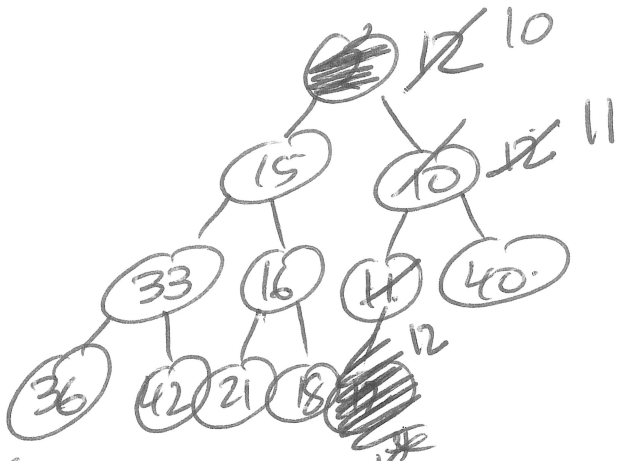
In order to be a valid binary heap, we need to be a
(1) complete binary tree
(2) obey heap node property.

To store in code

	2	15	10	33	16	11	40	36	42	21	18	12
0	1	2	3	4	5	6	7	8	9	10	11	12

array - for node at index i,
left child index $2i$
right child index $2i+1$
its parent is at index $i/2$ (int div)

① Delete Min
② Insert



Delete Min

① CEO takes off! (remove 1st slot)
② Mail boy realizes CEO gone. (copy last item into 1st slot)
③ VPs (child nodes) realize they should be in that spot. The lower number swaps w/ mailboy. Continue until mailboy finds a valid spot in the hierarchy

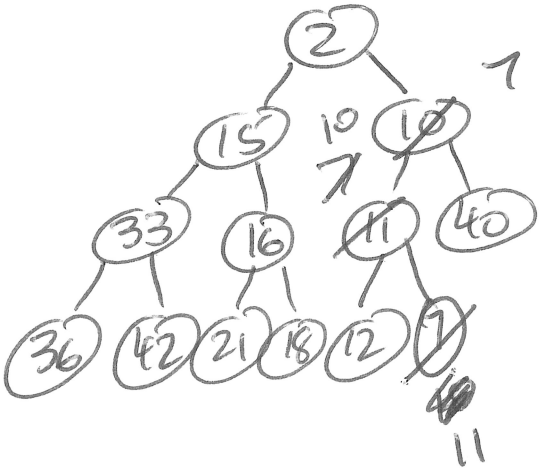
~~Percolate~~ Down

PERCOLATE

Run time: $n = 2^{h+1} - 1$ nodes
or at min $n = 2^h$ nodes

$h \leq \log_2 n$

Insert



Insert 7

① Add item in next valid location
② Percolate UP
- node challenges parent over
- swap if out of order

heapexample picture