CSE 101 3-7-26

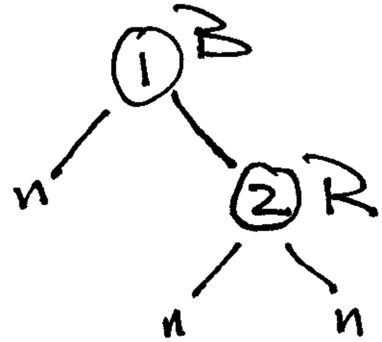
Supplemental Lecture

Red Black Trees

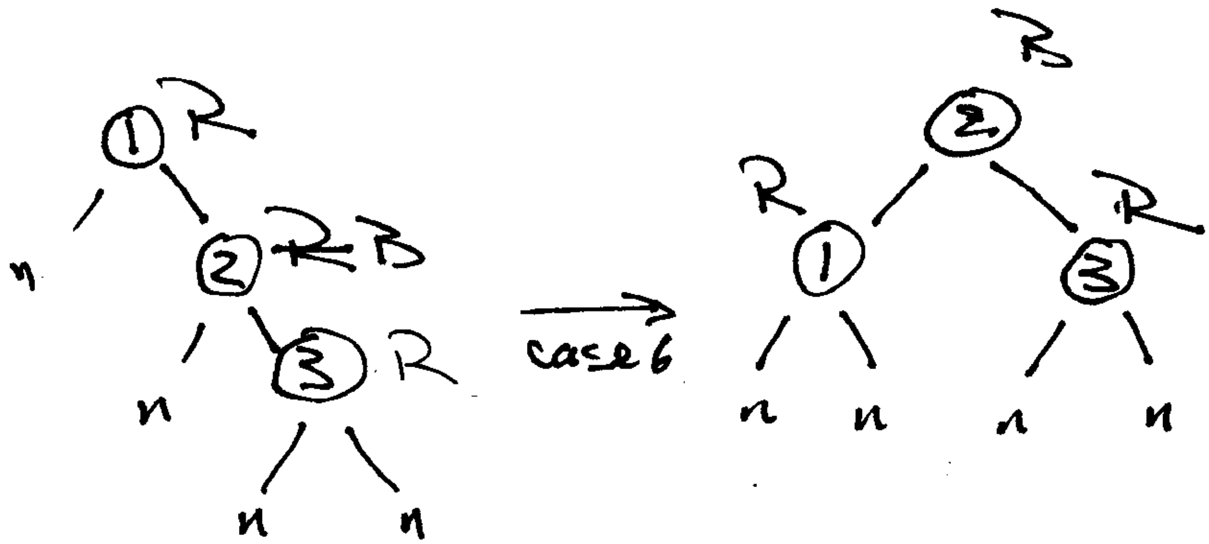
Ex. Insert 1, 2, 3, 4, 5 into an empty RBT

Insert 1:

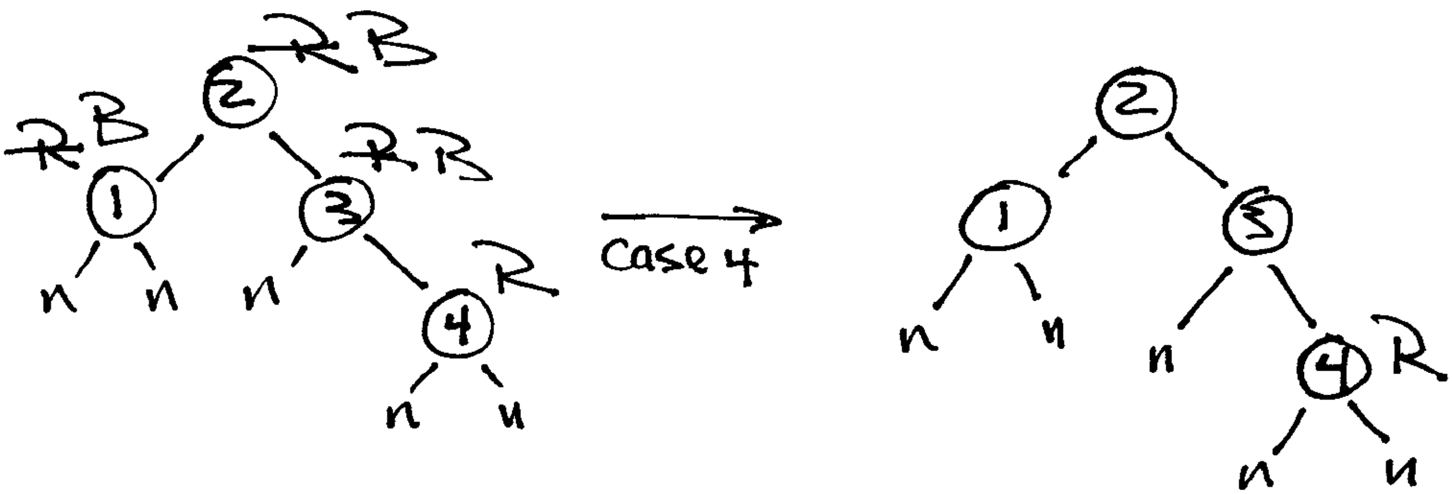
Insert 2:



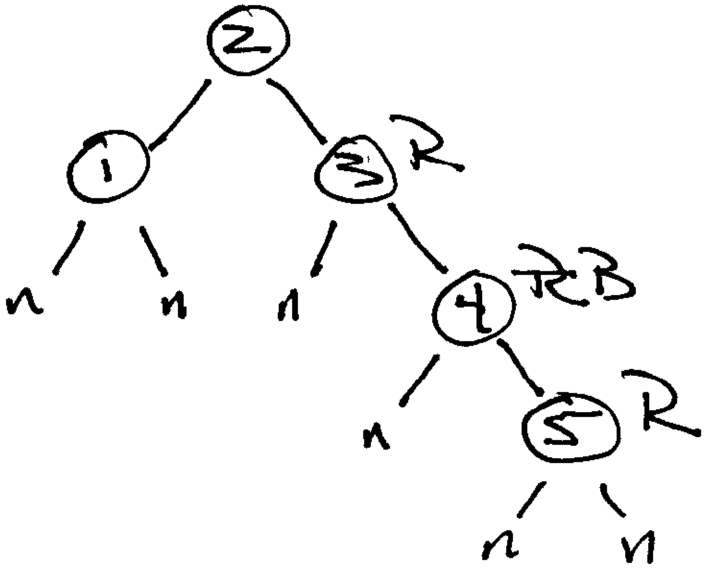
Insert 3



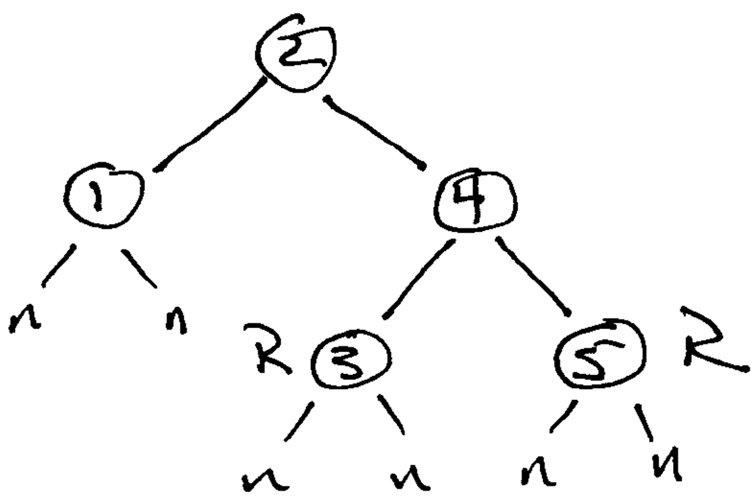
Insert 4



Insert 5



Case 6 →



6.5 Priority Queues

A Priority Queue (PQ) is an ADT that maintains a finite set S of elements (called records) with an attribute called key.

$$x = (\underbrace{\cdot, \cdot, \cdot}_{\text{satellite data}}, \text{key}) \in S$$

state: $S = \{ \cdot, \cdot, x, \cdot, \cdot \}$

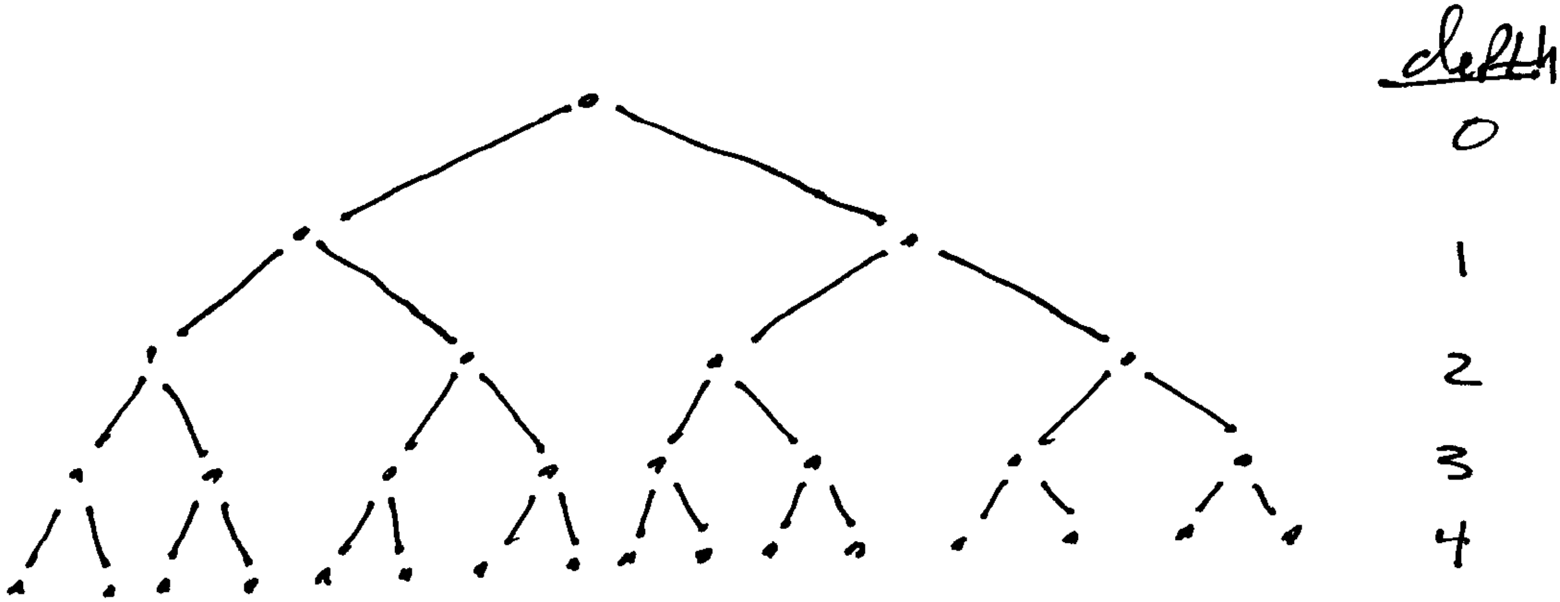
A <u>max PQ</u> supports the following ADT ops.

- Insert(S, x): insert x into S
- Max(S): returns record with largest key
- ExtractMax(S): return and delete record with largest key.
- IncreaseKey(S, x, k): change x.key to k if k > x.key, else do nothing. (Book considers else to be an error condition.)

Binary Heaps (chap. 6)

Defn

A complete Binary Tree (CBT) is a Binary Tree in which all leaves have same depth, and all internal nodes have 2 children.



<u>note</u>: # nodes at depth d is 2^d

Thus the # of nodes n in a CBT of height h

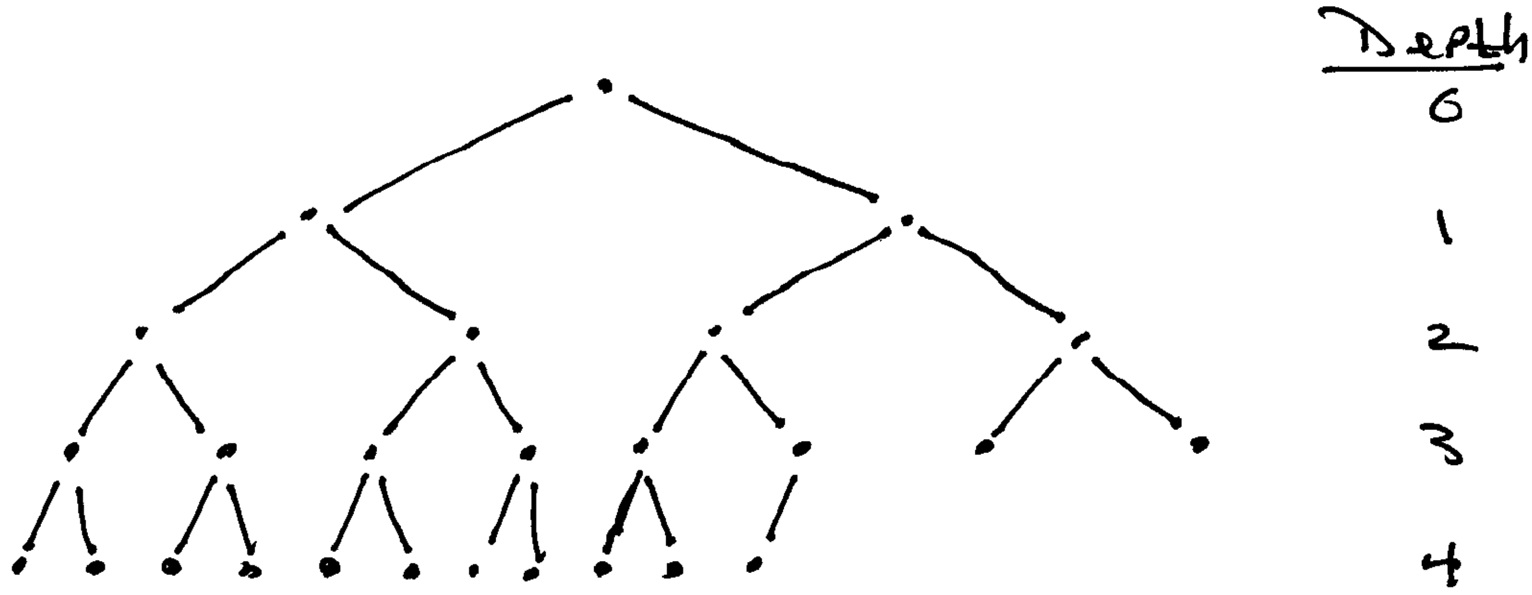
$$n = \sum_{d=0}^{h} 2^d = \frac{2^{h+1} - 1}{2 - 1} = 2^{h+1} - 1$$

Thus

$$\boxed{h = \lg(n+1) - 1}$$

<u>Defn</u>

An <u>Almost Complete Binary Tree</u> (ACBT) is a Binary Tree that is filled at all levels, except possibly the last, which may be partially filled from left to right.



Let $n = \#$ nodes in an ACBT of height h. Then

$$2^h - 1 < n \leq 2^{h+1} - 1$$

$$\therefore \quad 2^h \leq n < 2^{h+1}$$

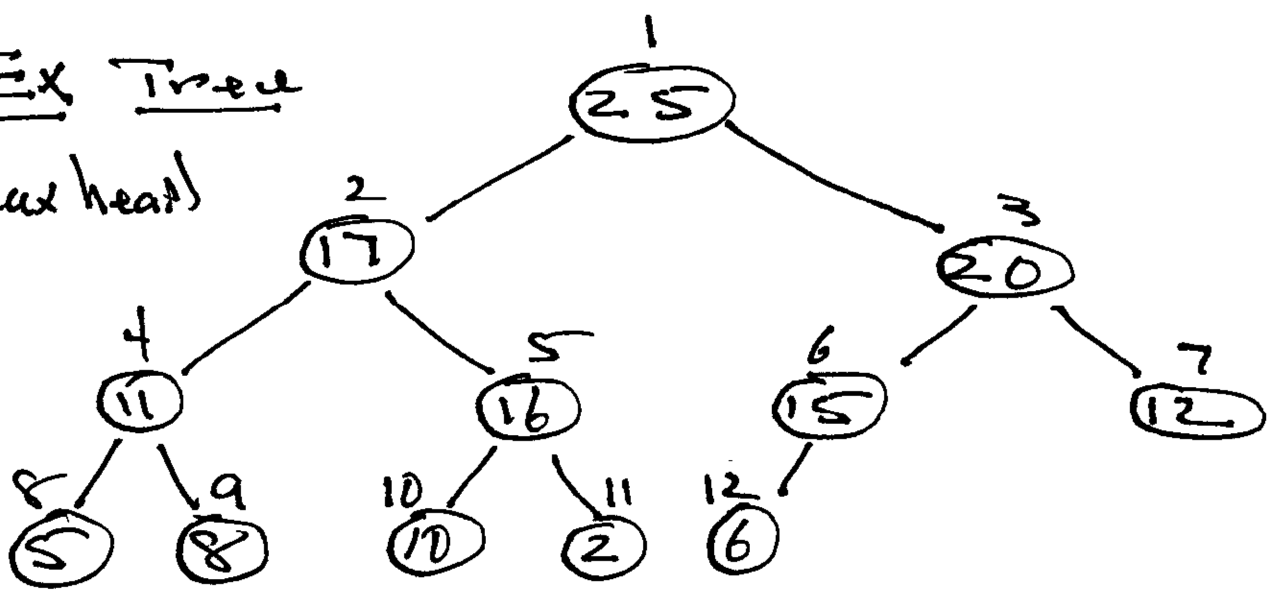
$$\therefore \quad h \leq \lg(n) < h+1$$

$$\therefore \quad \boxed{h = \lfloor \lg(n) \rfloor}$$

6.1 (Binary) Heaps

A <u>Heap</u> is an array object that represents an ACBT.

<u>Ex</u> <u>Tree</u>
(max heap)



<u>Array</u>:

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
A	25	17	20	11	16	15	12	5	8	10	2	6	•	•	•	•

<u>Array attributes</u>

- A.length = 16
- A.heapSize = 12

Helper functions

- $\text{Parent}(i) = \lfloor \frac{i}{2} \rfloor$ (for $i \geq 2$)
- $\text{left}(i) = 2i$
- $\text{right}(i) = 2i + 1$

Heap Properties

- max heap Property:

$$A[\text{parent}(i)] \geq A[i]$$

or

- min heap property:

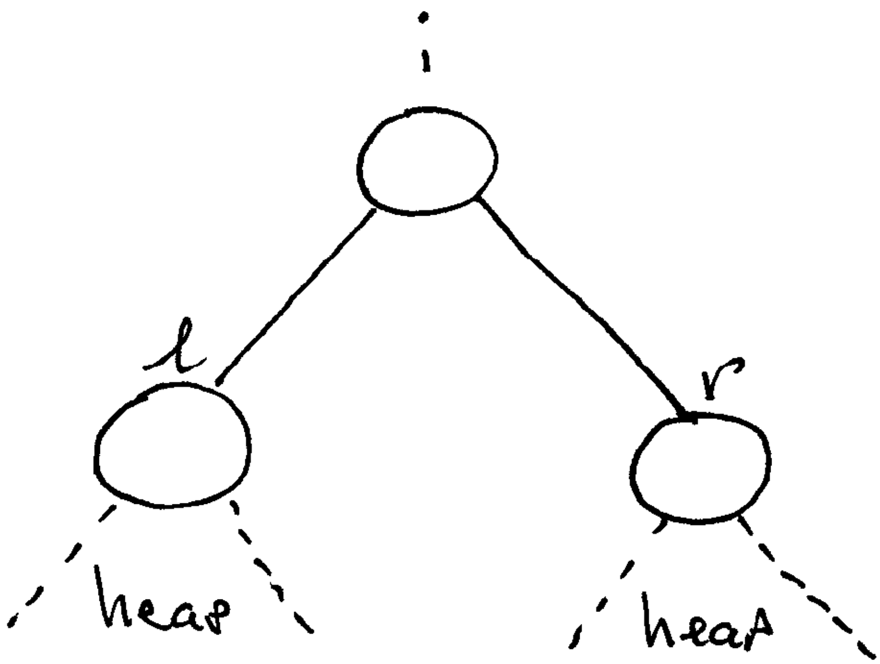
$$A[\text{Parent}(i)] \leq A[i]$$

Heap operations

- Heapify()
- BuildHeap()
- Heapsort()

PQ ops: HeapInsert()
HeapExtractMax()
HeapMax()
HeapDeleteMax()
HeapIncreaseKey()

6.2 Heapify

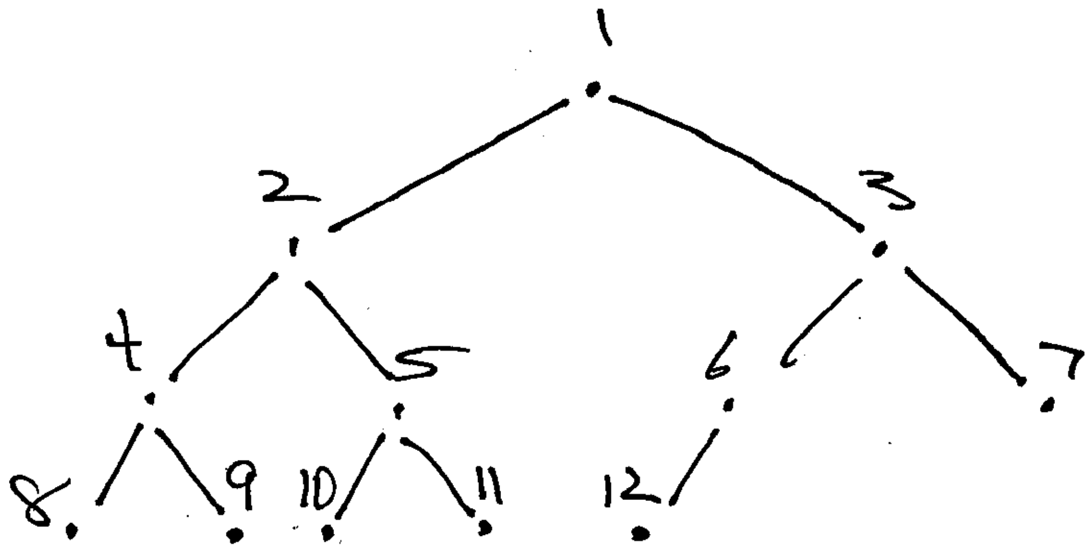


see pseudo-code

note: runtime = $\Theta(\log n)$

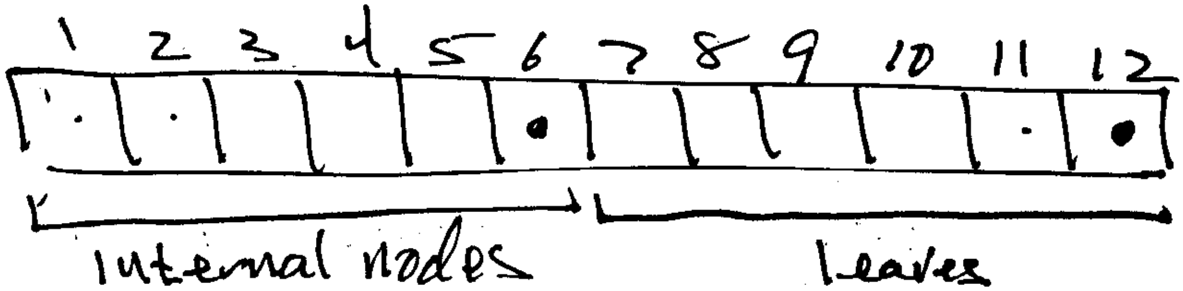
6.3 BuildHeap

Ex Tree



array

A



see pseudo-code:

Runtime = $\Theta(n)$