Lecture 21 - April 3

Priority Queues, Heap, Heap Sort

PQ: List Implementations
Heap: Structure, Relational Properties
Heap: Insertion, Deletion
Heap Sort

Announcements/Reminders

- Assignment 4 (on linked Trees) released
- Makeup Lecture (for ProgTest2) to be posted
- Bonus opportunity: Final Course Evaluation
- Office hours 3pm Thu this week
- Office hours, review session, ex. questions to be releasd
- Lecture notes template, Office Hours, TA Contact

List-Based Implementations of Priority Queue (PQ)

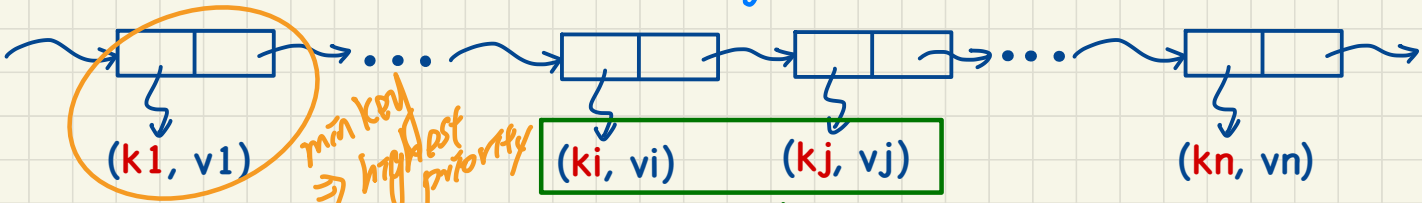
PQ Method	List Method	
	SORTED LIST	UNSORTED LIST
size, isEmpty	list.size O(1), list.isEmpty O(1)	
min	list.first O(1)	search min O(N)
insert	insert to "right" spot O(N) ≈ insertion sort	insert to front O(1)
removeMin	list.removeFirst O(1)	search min and remove O(N)

tracking space for temp

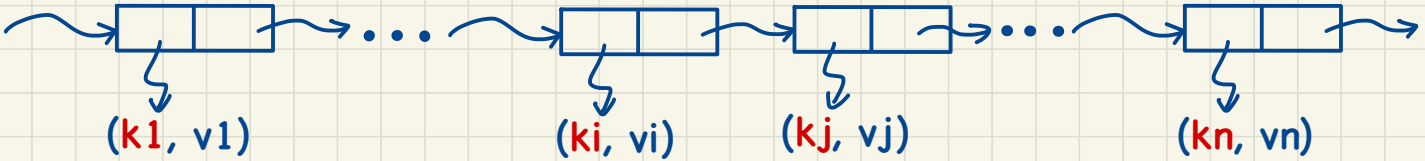
expansion of PQ. (2, infrequent

Approach 1: Sorted List

→ more suitable: (1) frequent retrieval/removal of top-priority entries.



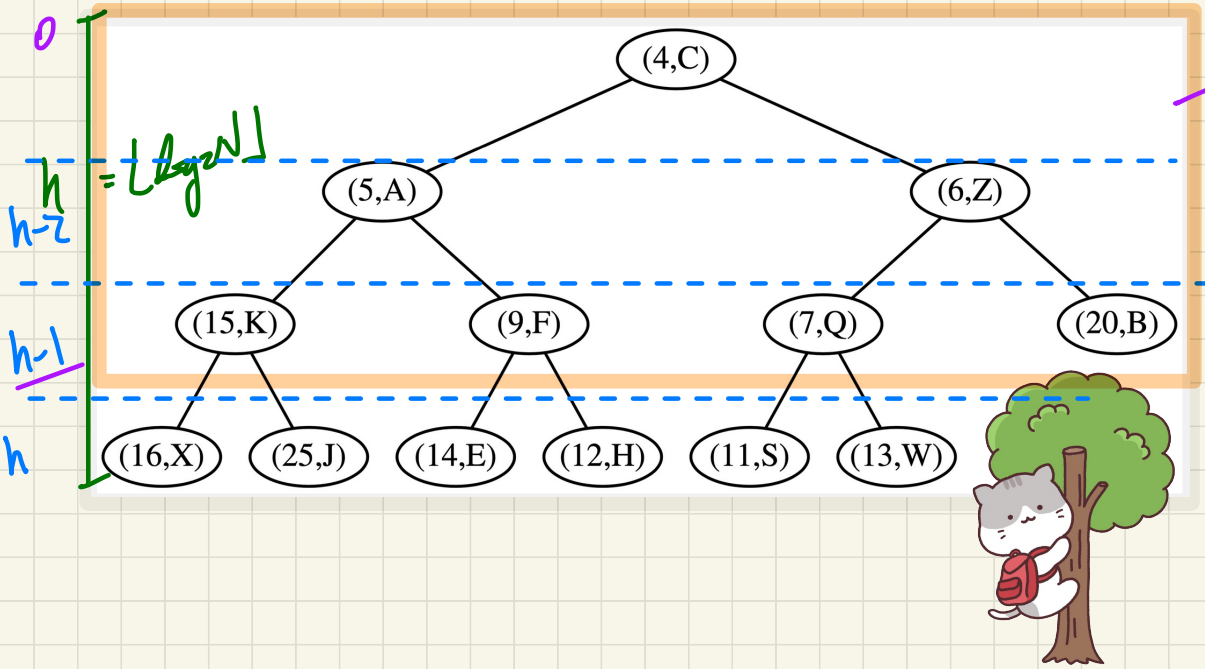
Approach 2: Unsorted List



Heaps: Structural Properties of Nodes

→ height: $\lfloor \log N \rfloor$.

Property: The tree is a complete Binary Tree



→ # nodes from levels 0 to h-1:

$2^h - 1$

Min # nodes:

$(2^h - 1) + 1$ 2^h

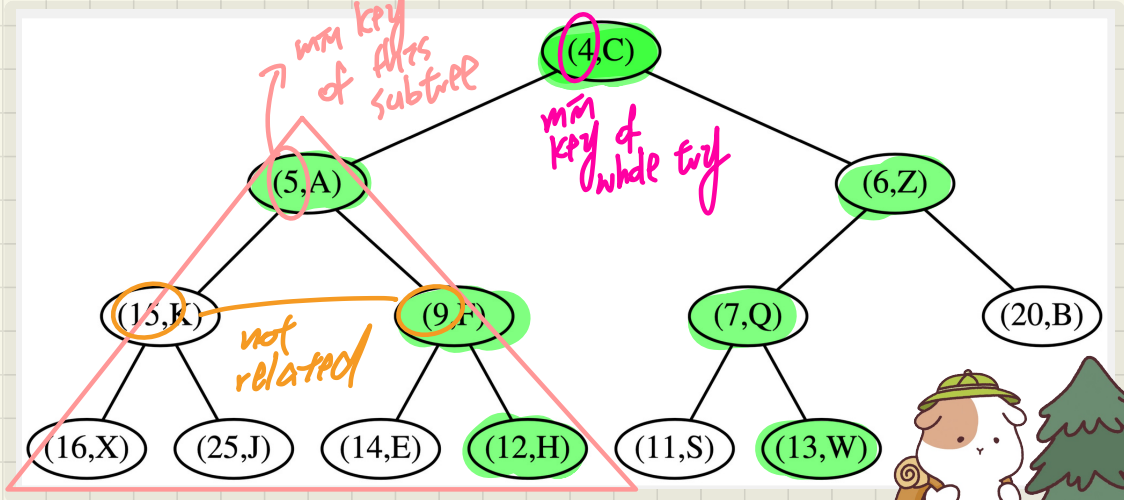
Max # nodes:

$(2^h - 1) + 2^h$

$2^{h+1} - 1$

Heaps: Relational Properties of Keys

Property: Each non-root node n is s.t. key(n) ≥ key(parent(n))



P1. Any leaf-to-root path has a sorted seq. of keys (non-ascending order).

P2. the min key exists in the root.

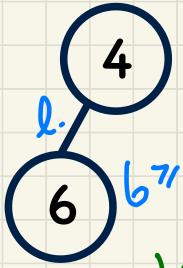
P3. key values between LST and RST are not related

Example Heaps

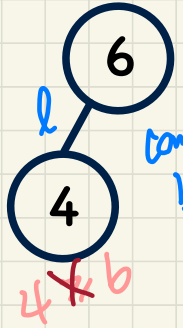
Example 1

smallest possible one-node heap.

Example 2

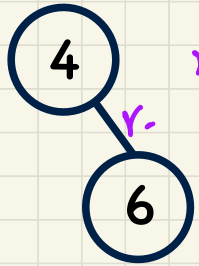


Example 3



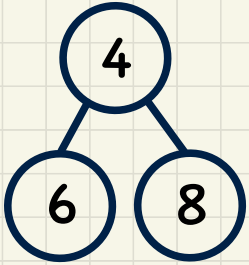
complete but violates H.O.P.

Example 4

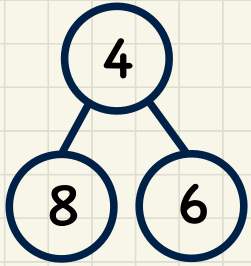


not complete but sat. H.O.P.

Example 5



Example 6



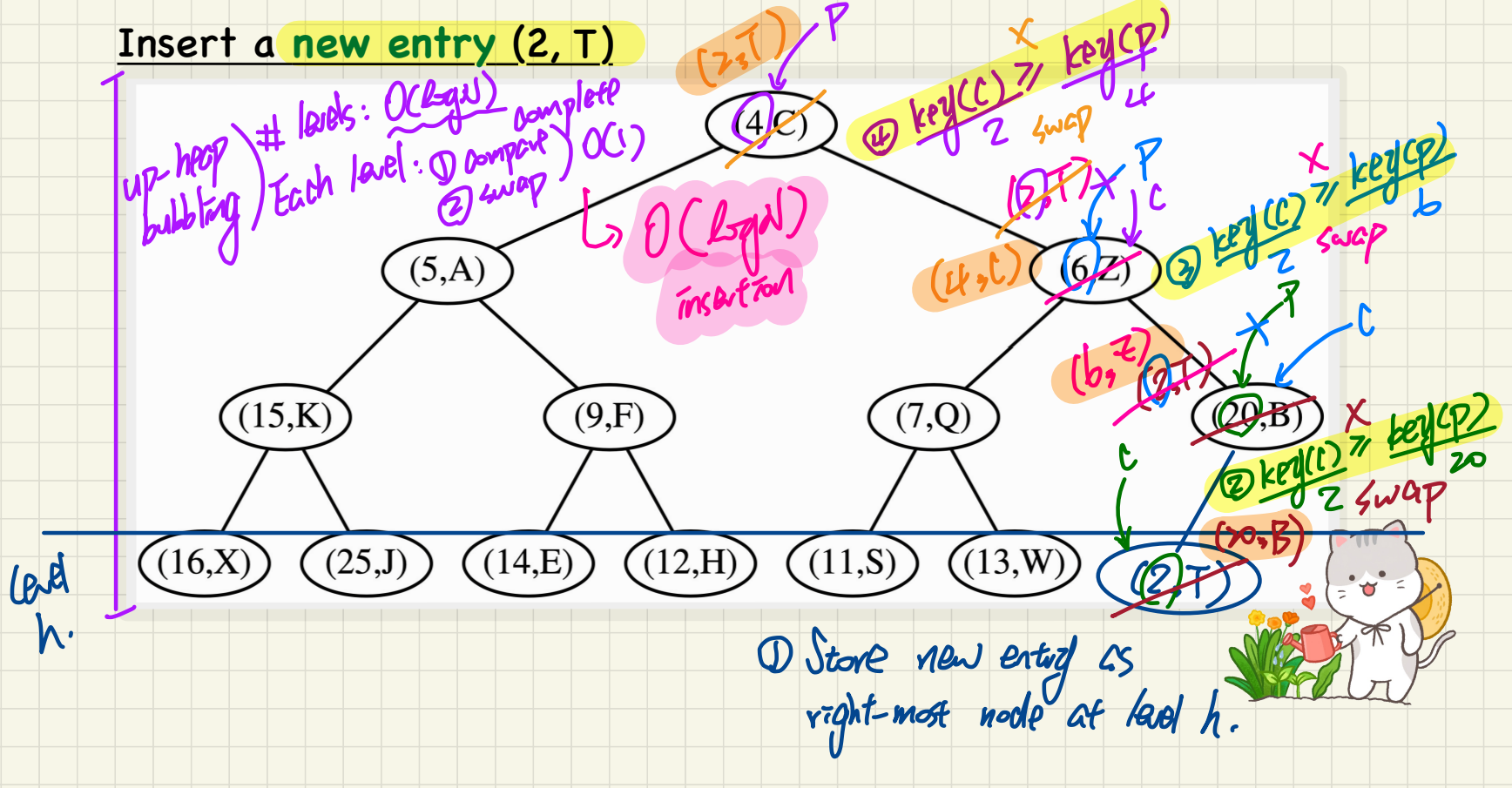
heaps!

full BT

⇒ complete BT.

Heap Operations: Insertion

Insert a new entry (2, T)

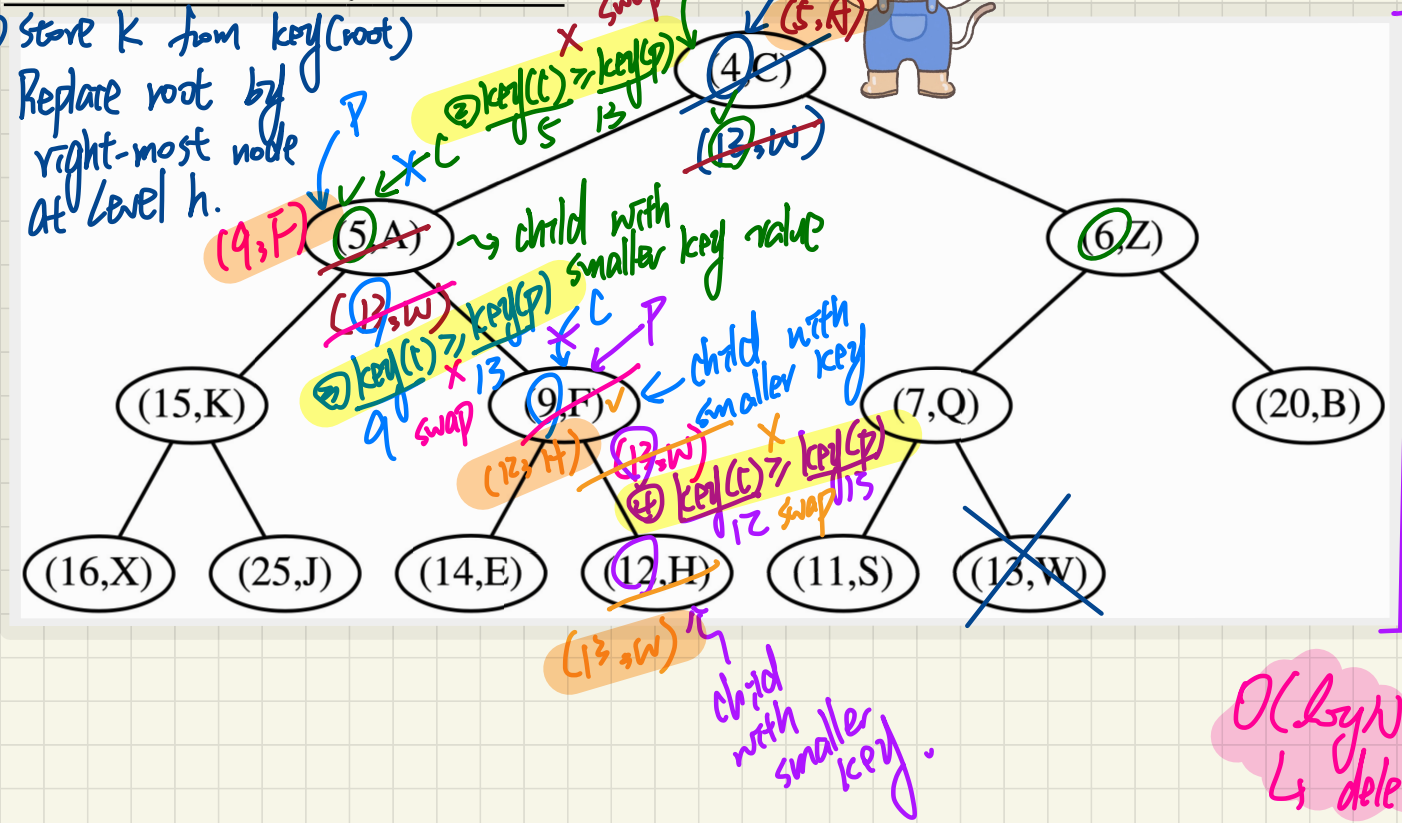


Heap Operations: Deletion

entry with min key (root).

Delete the root/minimum

① store K from key(root)
Replace root by right-most node at level h.

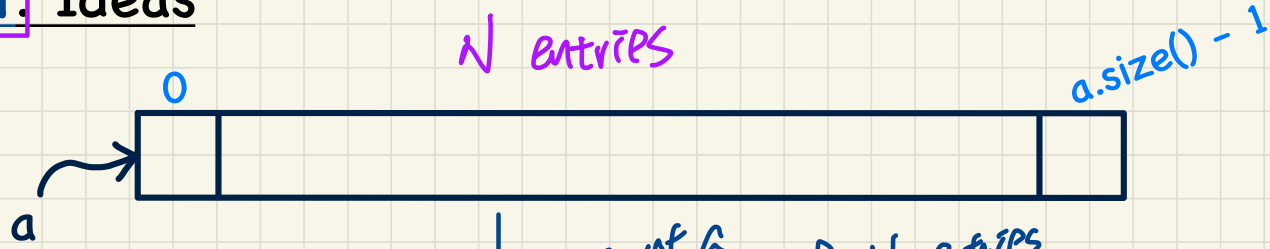


down-heap bubbling.
levels: O(log N)
Each level: O(1)
① choose c
② compare
③ swap
O(1)

O(log N) ↳ deletion

$O(N \cdot \log N)$.

Heap Sort: Ideas



Construct a heap out of N entries

(A) Top-Down

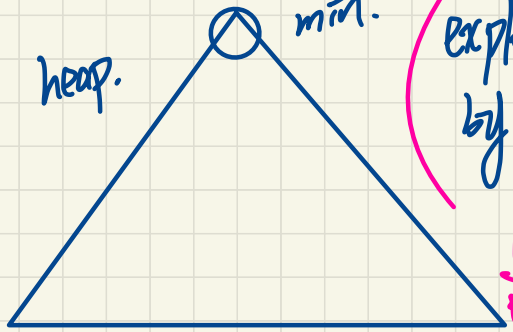
$O(N \cdot \log N)$.

(B) Bottom-Up

$O(N)$

$\approx$ selection sort

$O(N^2)$



(exploit the HOP (relational) by continuing to delete/extract min/root until tree is empty.

$O(N \cdot \log N)$ → heap sort.

N deletions, $\log N$ each deletion

Exam

~ 1~2 review sessions ~ Jeff, Sunda, Jackie

~ PDF guide

~ slides / iPad Notes

~ question booklet (answer booklets)

~ example code
↳ RSTNode.

↳ no calculator

~ assignments.

↳ little to none multiple choice questions

↳ definitions

↳ short answer (explanations, justification)

↳ coding / tracing

↳ proofs → e.g. asymptotic u.b.