Binary Search Trees, BST Sort

□ Runway reservation system

Airport with a single runway

Reservations for future landings is interested

Reserve request specific landing time t

Add t to set R if no landings are scheduled within k min

Remove from set R after plane landing

We'd like to do all above in $O(\lg n)$ time, $|R| = n$

◦ Example

$k = 3$ min

53 → ok

44 → not allowed, to close to 41.2

20 → not allowed, passed

△ Unsorted list / array: can insert in $O(1)$ w/o check, but sadly check takes $O(n)$ time

△ Sorted array:

4 20 32 37 45

By doing binary search, can find smallest i such that $R[i] \geq t$

in $O(\lg n)$ time.

Compare $R[i]$ and $R[i-1]$ against t in $O(1)$ time

Sadly, actual insertion (shifting) will take $O(n)$ time

& Heaps

To find element that is $\leq k$ or $\geq k$ from t, take $O(n)$ time

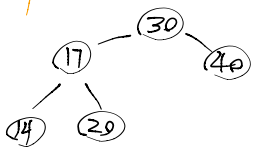
* if we can do fast insertion into a sorted array, we would be able to make it $O(\lg n)$ time with sorted array!

main problem: Sorted array doesn't allow fast insertions

↳ BUT BST do!

□ Binary Search Trees

◦ example



node x key(x)

pointers: (unlike a heap)
- parent(x)
- left(x)
- right(x)

◦ Invariant

For all nodes x, if y is in the left subtree of x

↳ $key(y) \leq key(x)$

if y is in the right subtree of x

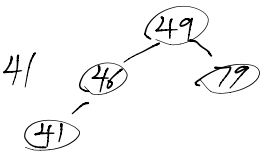
↳ $key(y) \geq key(x)$

- how BST solve the problem
  - Insert:

insert 49 : (49)

insert 79: (49) — (79)

insert 46 

insert 41 

h (height of tree, could be O(n))

now new requested time 42 (h=3):

Compare 42 vs 49 ✓

42 vs 46 ✓

42 vs 41 X (< 3)

* if h is height of the tree, then insertion w/ check to be O(h) time

* Beauty of BST: can do "check" operations while inserting, w/o adding extra cost.

- Operations that can be done in O(h) time
  - find-min(): go to the left until hit the leaf (heap takes Θ(1) time)

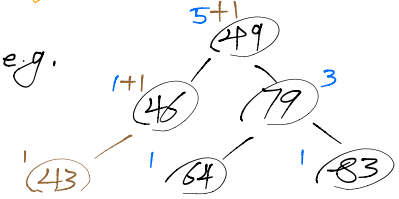
- next_large(x)

New requirement

Be able to compute Rank(t) = how many planes are scheduled to land at times $\leq t$

Augment the BST structure

e.g.



blue = subtree size

- add extra number associated with each node → subtree size
- insert or delete → modify subtree size

add (43)

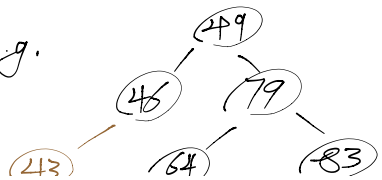
Answer the question (rank(t))

What lands before t? Want in O(h) time

1. Walk down the tree to find the desired time.
2. add in the nodes that are smaller
3. add in the subtree size to the left

e.g. t=79

e.g.



49 < 79 add 1 → move to right → add subtree sizes of 46
(add 2)

79 ≤ 79 add 1 → move to right → add 1 (subtree of 64

total = 5

Bad news: h could be Θ(n) !
see u next time!