BINARY SEARCH TREES, BST SORT

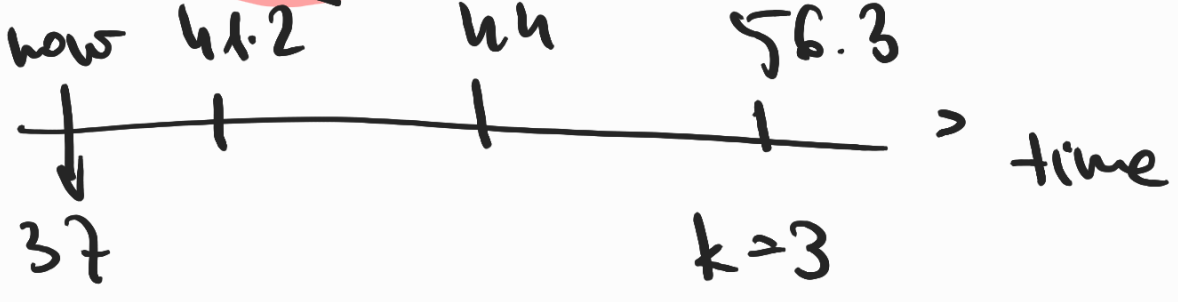
Runway reservation system:

Airport with single runway.
Reservations for future landings.
Reserve request specifies landing time t
Add t to the set R if no other landings are scheduled with in k minutes
Remove from set R after plane lands.

$|R| = n$ → size of a set

O(log n) time

Example



~~33~~ OK

44 not allowed (to close to 41.2)

20 not allowed (post)

Unsorted list/array:

Insert in O(1) w/o check

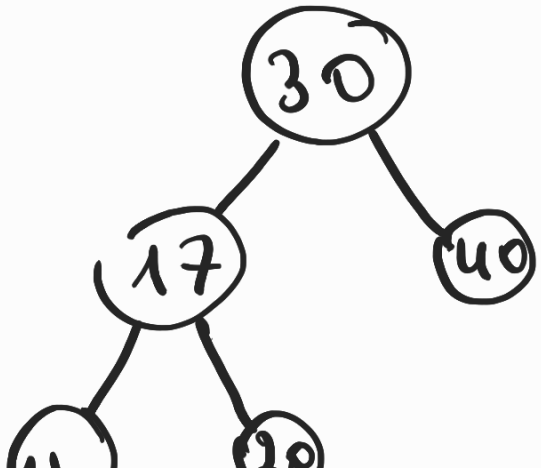
check + takes O(n) times

Sorted array:

Find smallest i such that A[i] ≥ t in O(lg n) time.

Compare A[i] and A[i-1] against t in O(1) time.

BINARY SEARCH TREES



node x: key(x)

pointers: parent(x)

left(x)

right(x)

For all nodes x, if y is in the left of x $key(y) \leq key(x)$.

If y is the right .... $key(y) \geq key(x)$

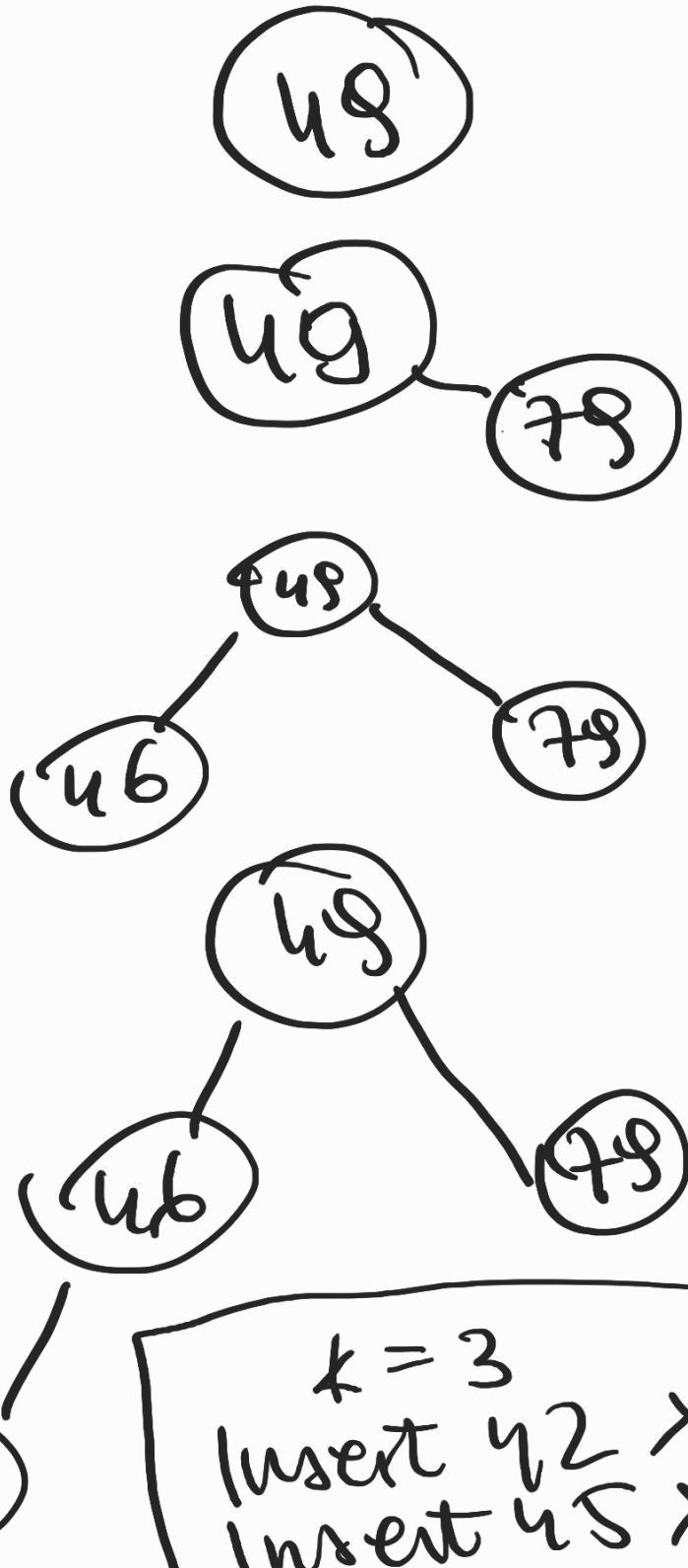
INSERT

Insert 49

Insert 79

Insert 46

Insert 41

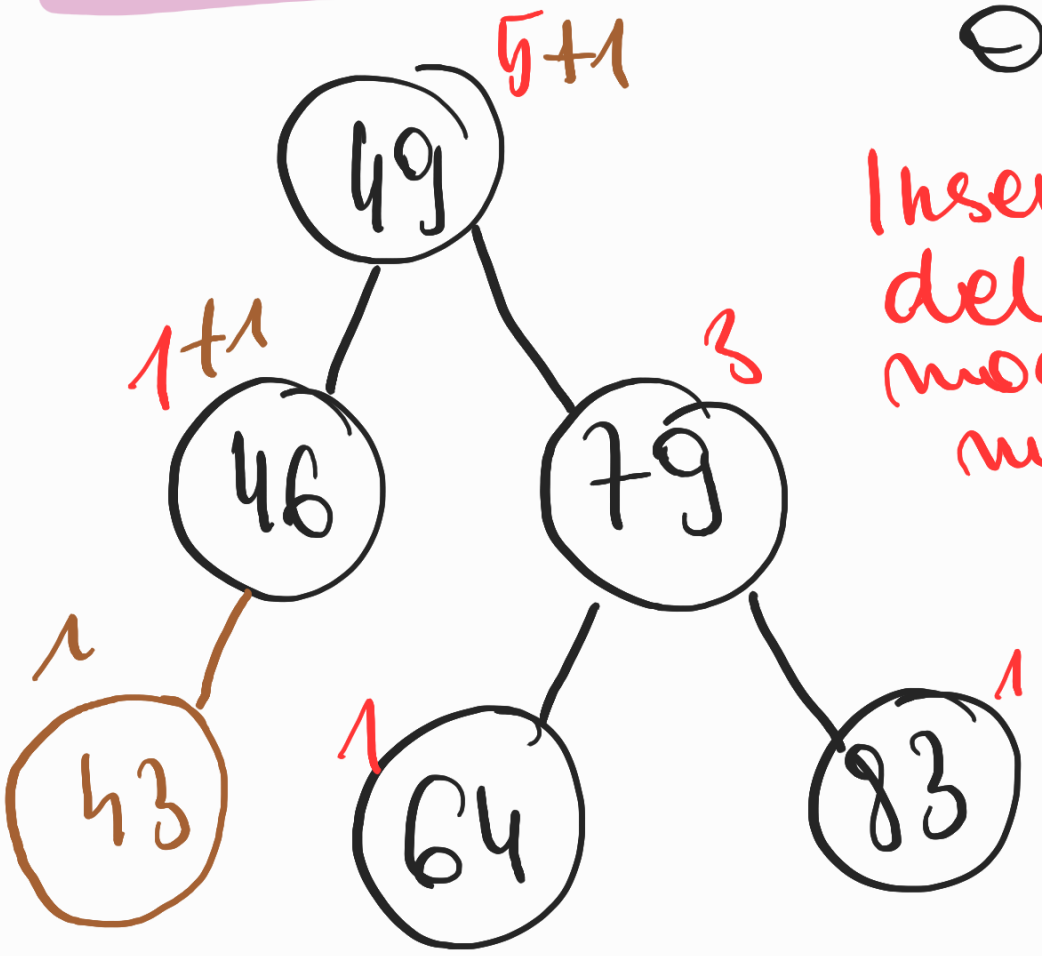


h

height of tree
insertion $O(h)$ time

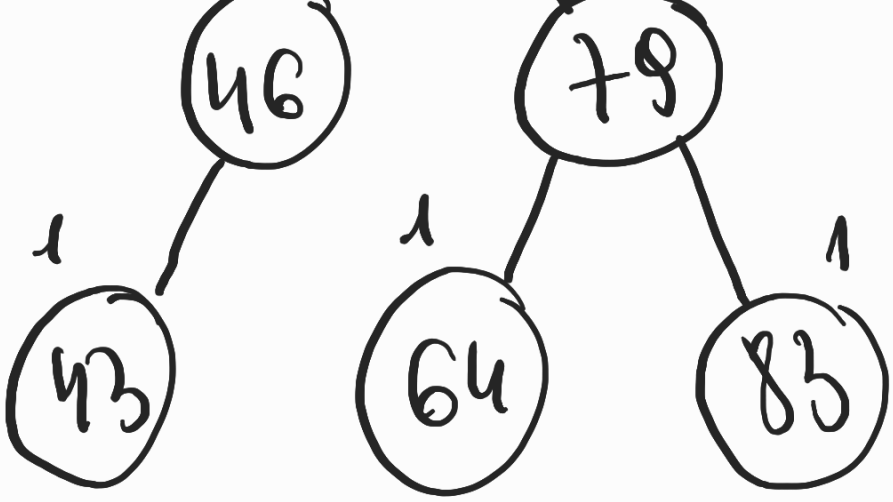
FIND-MIN( ) keep going left
$O(h)$

FIND-MAX() keep going right
$O(h)$



Insert or delete modify "size" numbers

What bounds before t?
$O(h)$



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

$t = 79$

49 add 1

add 2 (subtree 46)

79 add 1

add 1 (subtree 64)

5

Rank(t): how many planes are scheduled to land at

trees $\leq t$