Ordered Dictionaries

→ In addition to dictionary, we want to support

min() : brings minimum key

max() : " max key.

Predecessor(k) : brings previous key

Successor(k) : brings next key.

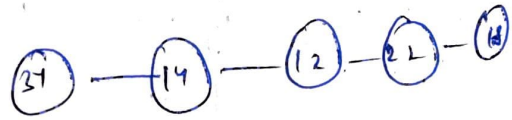
→ we need some kind of order on keys.

A List based implementation

Unordered list

(i) searching takes O(n) time

(ii) inserting → O(1)



Ordered list

searching → O(n)

inserting → O(n)

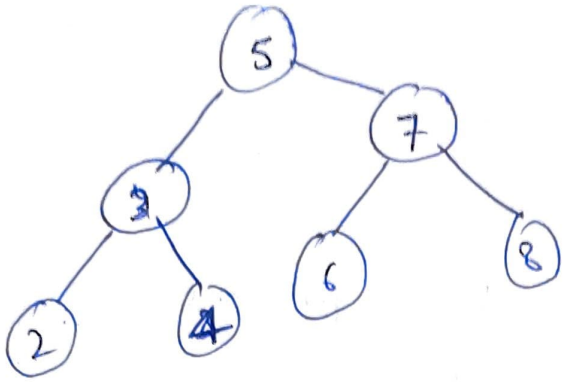
for array, searching → O(log n).

Binary Search Tree

A tree in which keys are stored

For every node, left subtree has keys that are less than node and right subtree has keys greater than node.

ex. 2, 3, 4, 5, 6, 7, 8



BST search, search like binary search.

→ Recursive version

```
search (T, k)
    x = root(T)
    if x = NIL, then return NIL
    if x = key
```

→ min().

keep going left and stop where there is no left tree

max()

keep going right and stop where there is no right subtree

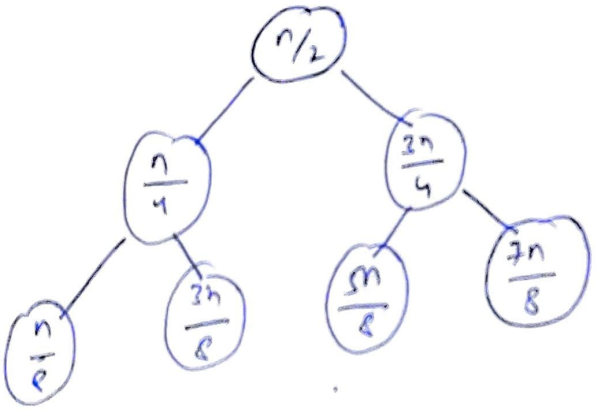
successor

case 1 right subtree exists → find min(right subtree)

case 2 right subtree done → parent (key).

keep going to ancestor until. the key is in left subtree of the ancestor.

what kind of BST do we want to make given $n = 2^k - 1$ elements.



This will make a complete binary tree.

This tree is good because all our operations are $O(h)$.