CSE 101-02 2-23-23

- mid 2 : next Thursday

Dictionary ADT

states: a state is a set of ordered pairs

(Key, Value)

↑

must belong to some ordered set.

Keys are unique.

operations:

- lookup(key)
  returns value assoc. with key, or nil if key does not exist

- insert(key, value)
  adds a new pair to dict.
  Pre: lookup(key) = nil

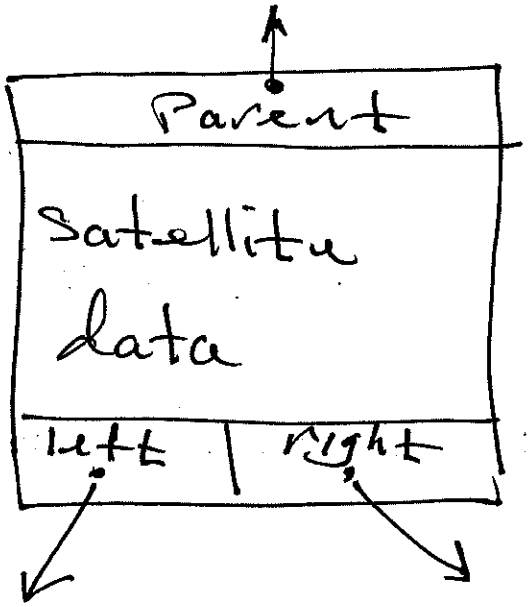
- delete(key)
  removes (key, value) from dict.
  Pre: lookup(key) ≠ nil.

Implementations

- BST : chap 12 (Pa7)
- RBT : chap 13 (Pa8)
- HashTable : chap 11

Binary Search Tree (BST)

Node Object



Dictionary (key, value)

Simplify : Key

BST Properties

Let x, y be (pointers to) nodes.

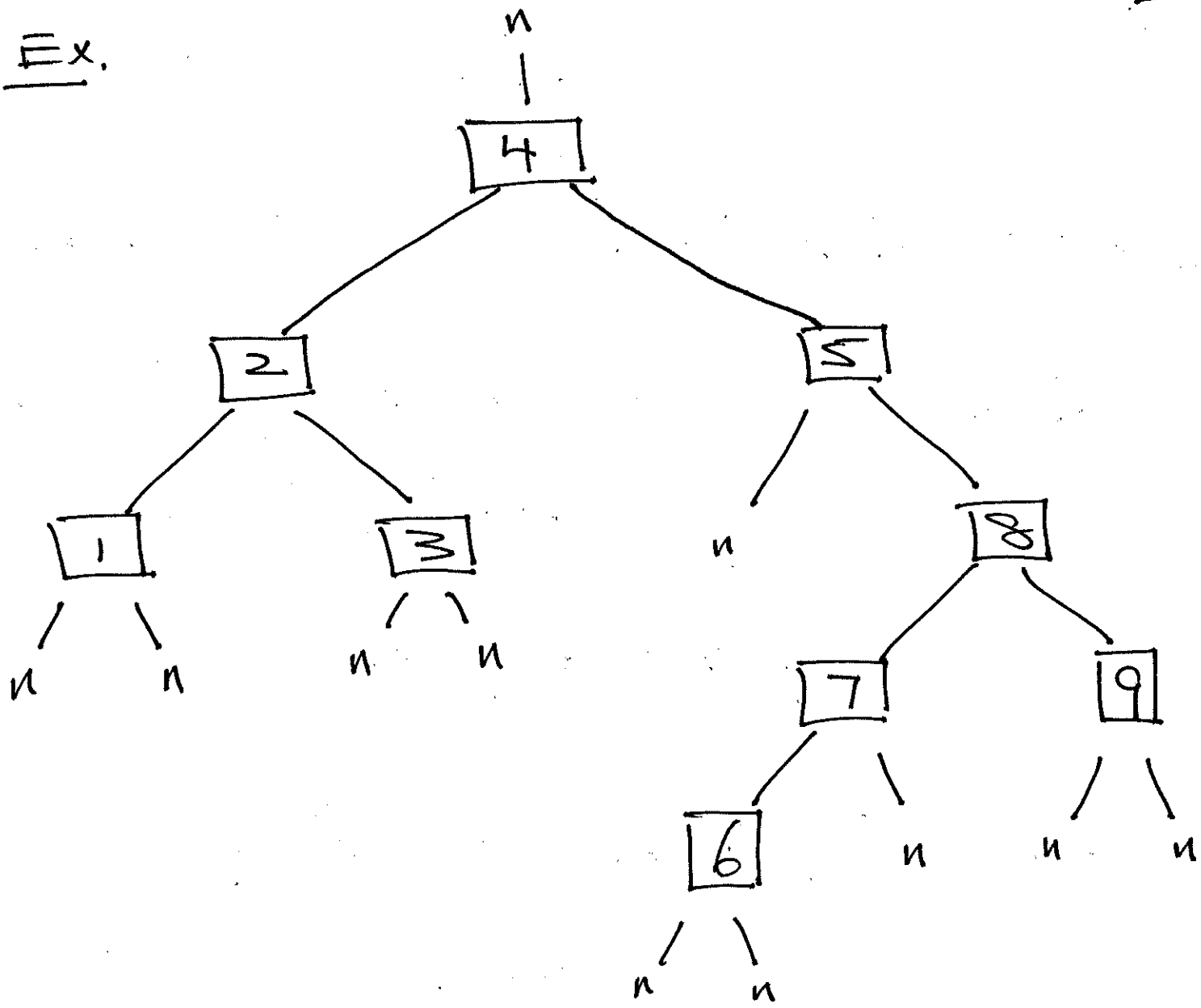
(1) if y is in the left subtree of x, then

$$y.key \leq x.key$$

(2) if y is in the right subtree of x, then

$$x.key \leq y.key$$

Ex.

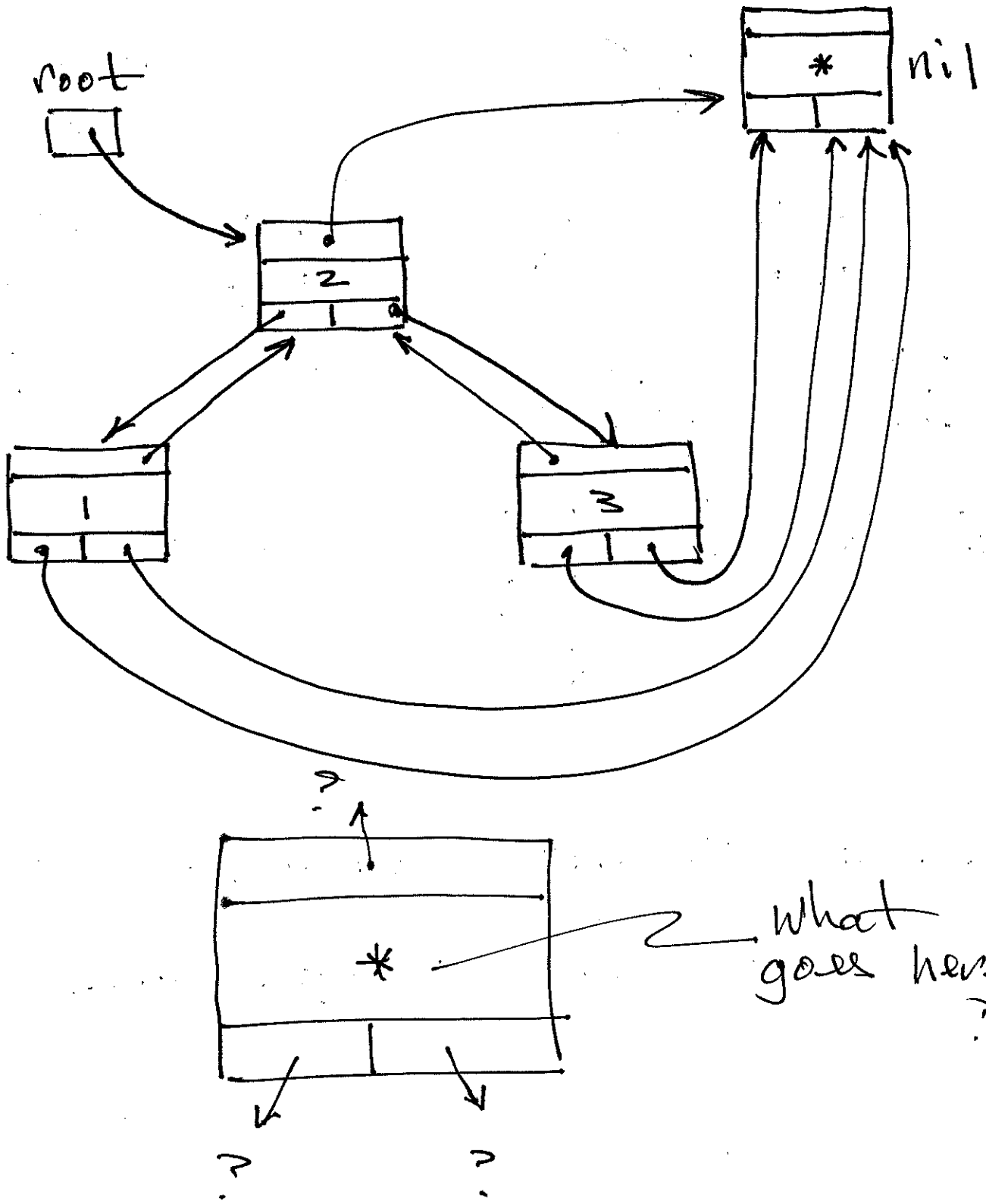


Defn

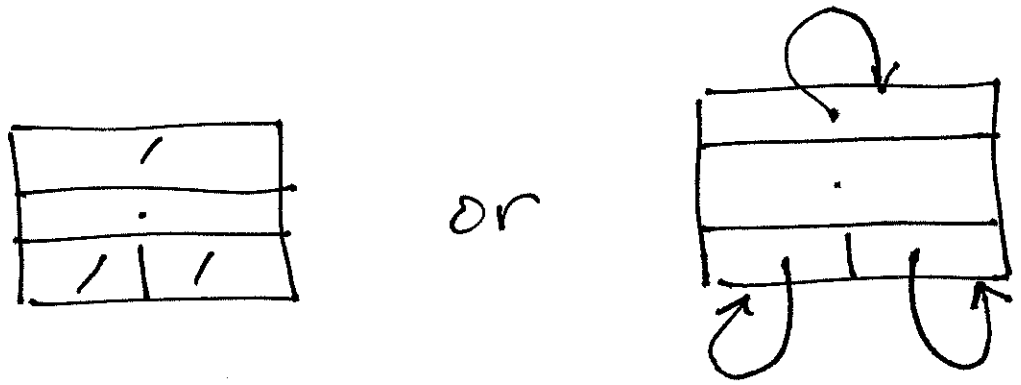
- a leaf is a node with no children
- an internal node is a non-leaf.

Implementing with Sentinel nil node

Ex



could be



- Tree traversals

node

- InOrderTreeWalk(x)
- PreOrderTreeWalk(x)
- PostOrderTreeWalk(x)

Output: Ex.

IOTW: 1 2 3 4 5 6 7 8 9
Pre: 4 2 1 3 5 8 7 6 9
Post: 1 3 2 6 7 9 8 5 4

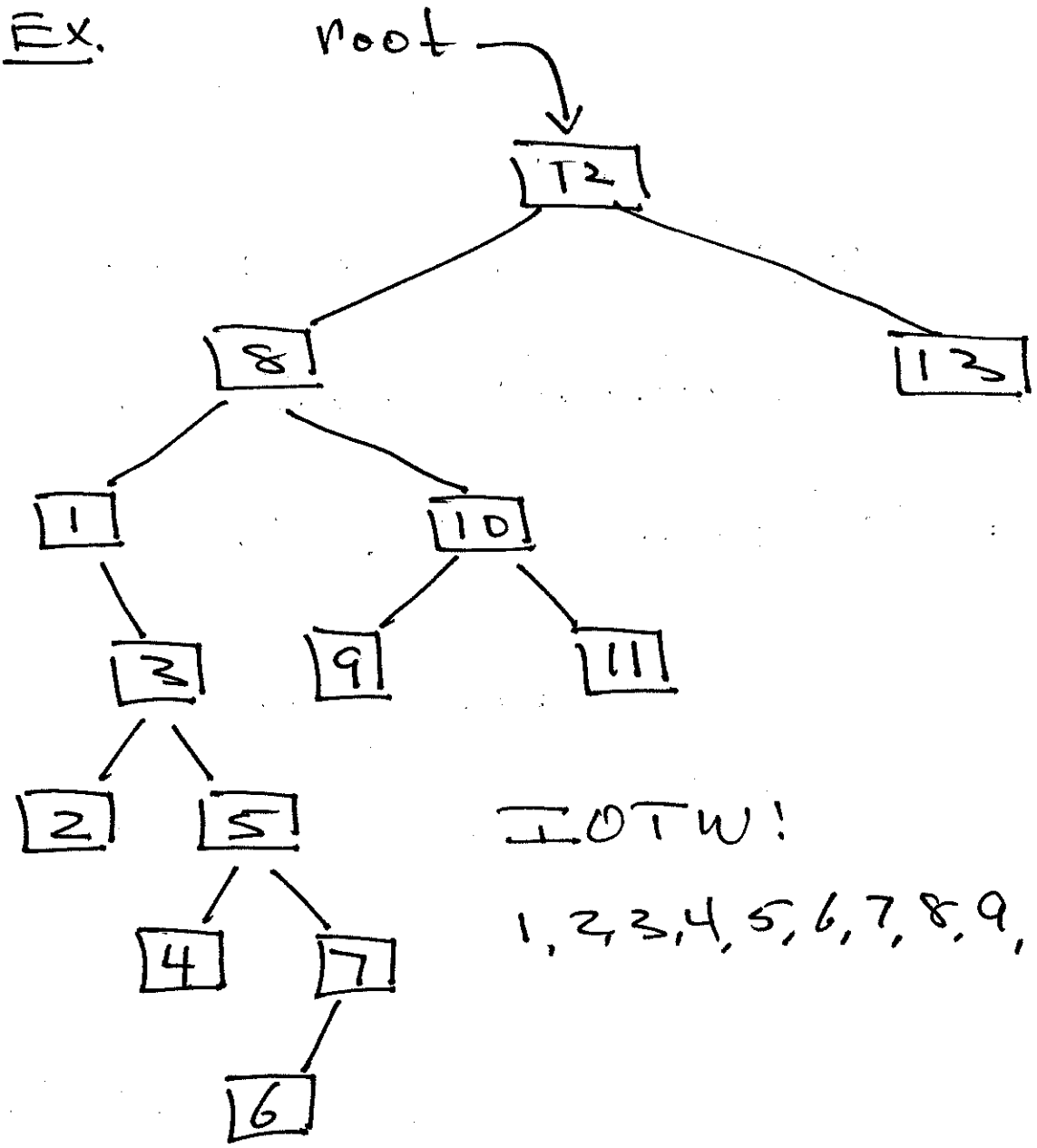
Queries

- TreeSearch(x, k)
- TreeMinimum(x)
- TreeMaximum(x)
- Successor(x)
- Predecessor(x)

Defn

The successor of a node x is the next node after x to be processed in an in-order tree walk

Ex.



IOTW!

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13

Successor(1) = 2 case 1

Successor(8) = 9 case 1

Successor(7) = 8 case 2

Two cases: Successor(x)

Case 1: x has a right child. Successor(x) is the leftmost descendant of that child.

case 2: x has no right child. Successor(x) is the lowest ancestor of x whose left child is also an ancestor of x. (note: x is its own ancestor.)