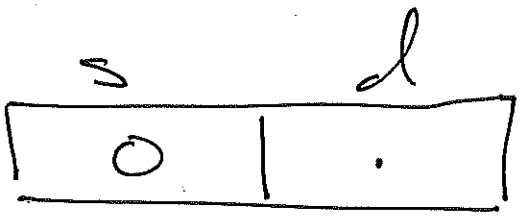
CSE 101-01 5-18-23

PaG : ext. 2 days → Monday

BigInteger : zero state



( ) empty list

from long constructor : x is a long

x > 0 : signum = 1, digits = (·, ·, ·, ...)

x < 0 : signum = -1, digits = (·, ·, ...)

from string constructor :

s = "1|23|45|67|89|87|65|43"

suppose base = 100, Power = 2

Use : in string.h

substr( )

stol( )

also : find_first_of( )

find_first_not_of( )

Ex. normalize & subtraction

let $p = 2$, $b = 100$.

$$(13\ 23\ 81)$$

$$(52\ 20\ 92)$$

$$\overset{-1}{(-39}\ \overset{-1}{3}\ -11) \quad \text{Vector Sub}$$

$$100 \qquad 100$$

$$(-1\ 61\ 2\ 89) \quad \text{normalize}$$

$$\boxed{-1}\ \overset{-1}{(1}\ \overset{-1}{-61}\ \overset{-1}{-2}\ -89)$$

$$-62\ \ -3$$

$$100\ \ 100\ \ 100$$

$$(0\ 38\ 97\ 11)$$

$$(38\ 97\ 11) \quad \text{normalize}$$

return — state of normalized list

Dictionary ADT

chapters (CLRS):

- ch 12 : BST
- ch 13 : RBT
- ch 11 : Hash Tables

state: a finite set of ordered pairs

(key, value)

i.e. a state is a set

$\{(K_1, V_1), (K_2, V_2), \ldots, (K_n, V_n)\}$

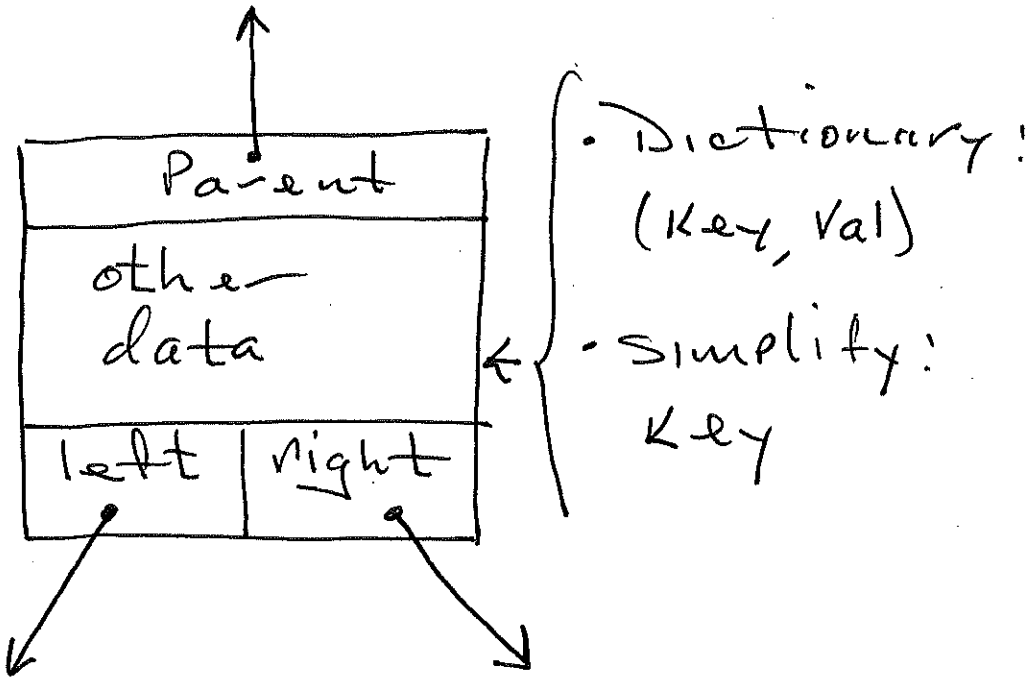
Require: keys are distinct.

operations

- lookup(key): return value assoc. with key or nil if key does not exist
- insert(key, value): adds a new pair to dictionary
  Pre: lookup(key) = nil
- delete(key): removes pair from dictionary
  Pre: lookup(key) ≠ nil

Binary Search Trees (BST)

Node Object



BST Properties

let x, y be nodes

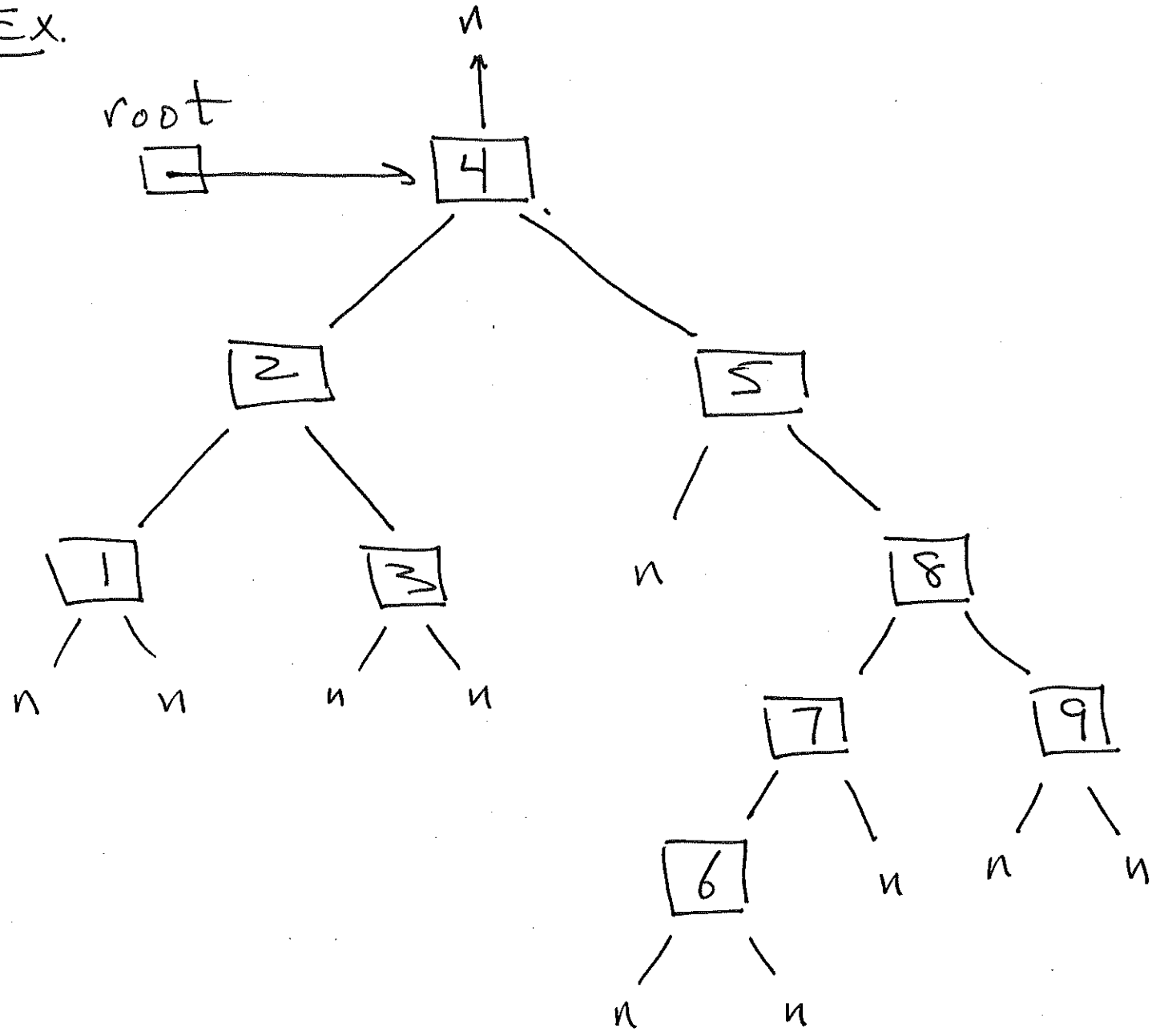
(1) if y is in left subtree of x, then

$$Key[y] \leq Key[x]$$

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

$$Key[x] \leq Key[y]$$

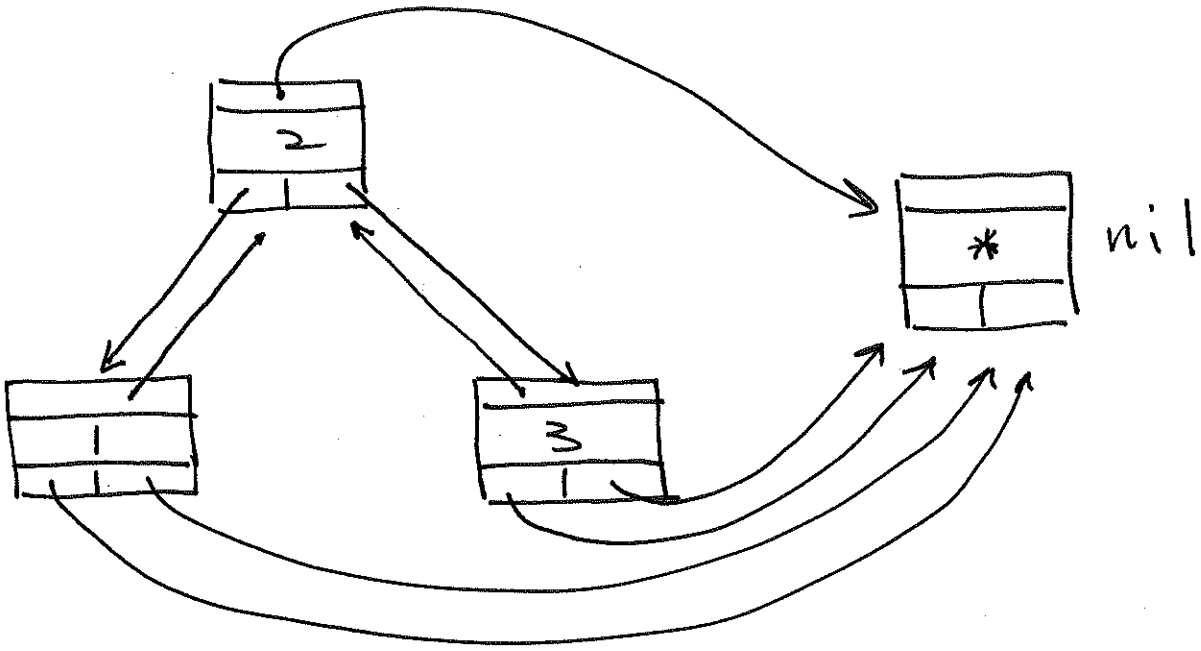
Ex.



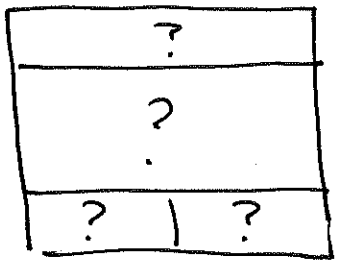
Depth

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

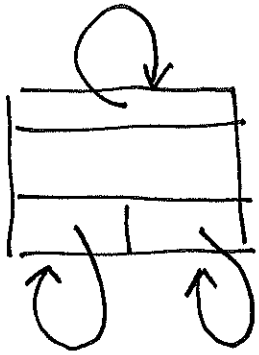
Recommend: dummy nil node



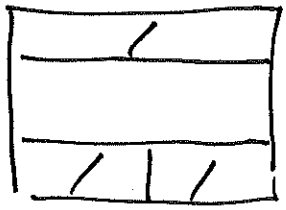
nil:



← what goes here



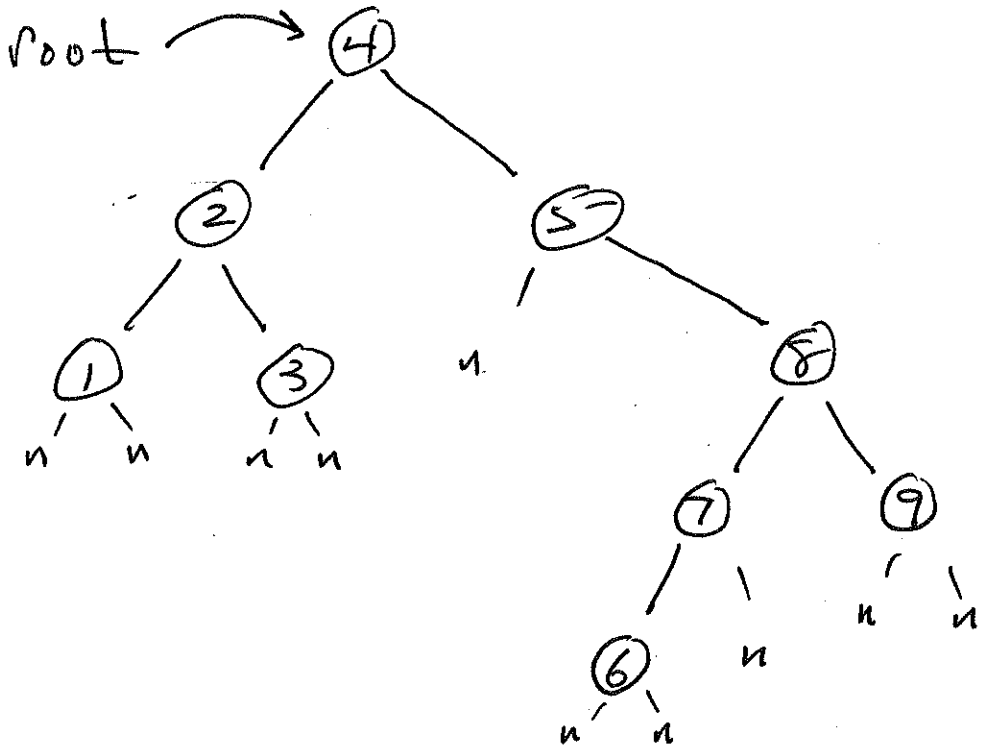
or



Tree traversals:

- InOrderTreeWalk(x)
  ↑ node in a BST
- PreOrderTreeWalk(x)
- PostOrderTreeWalk(x)

Ex.



IOTW: 1 2 3 4 5 6 7 8 9

Pre OTW: 4 2 1 3 5 8 7 6 9

Post OTW: 1 3 2 6 7 9 8 5 4