CSE 101-01 2-28-23

- midterm 2 Thursday 3-2-23

Runtimes for Tree ops.

- Traversals

  InOrderTreeWalk(x)
  Pre ↑ root
  Post

  cost = $\Theta(n)$, n = #nodes in T

- Queries ↓ root

  TreeSearch(x)
  " Min(x)
  " Max(x)
  " Successor(x)
  " Predecessor(x)

Def:

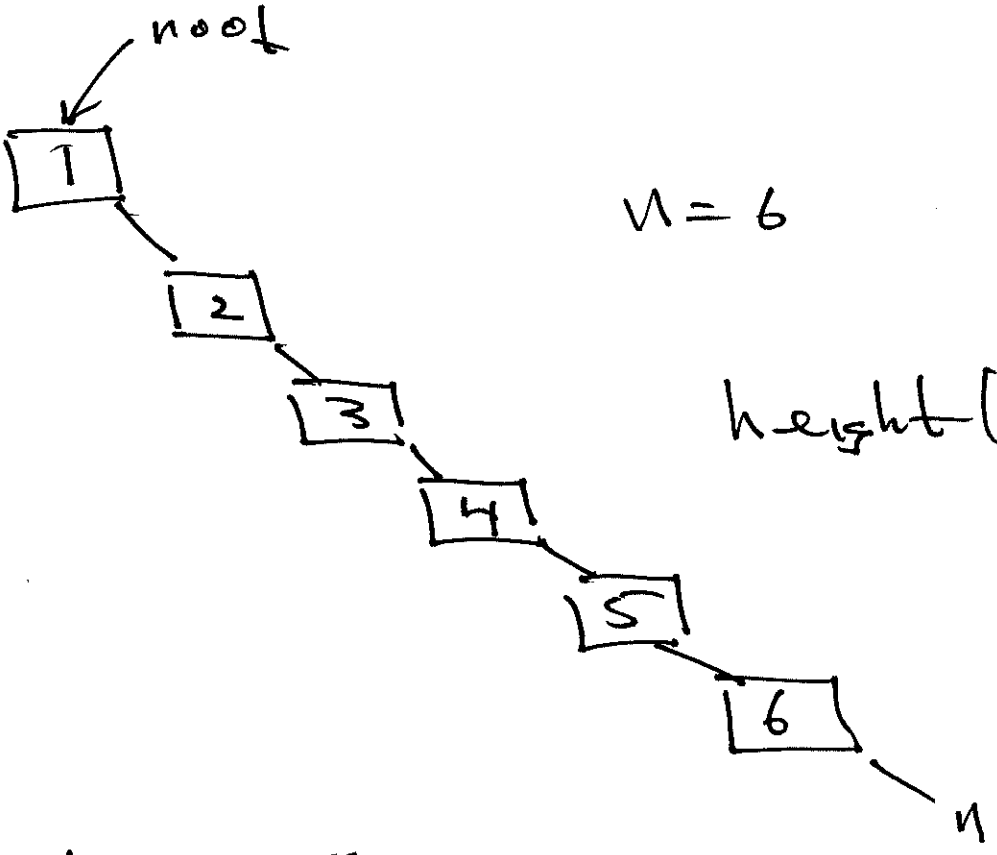
height(T) = dist. from root to deepest leaf.

height(x) = height(subtree rooted at x)

Note: height(x) $\leq$ height(T) for all nodes x.

Worst case cost of all queries is Θ(height(T))

Ex. worst BST for queries

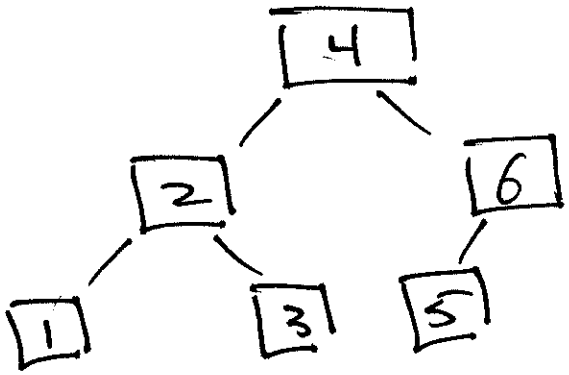


$n = 6$

height(T) = 5

cost = ~~6~~ 5

Ex. Best Possible BST



height(T) = 2

cost = 2

In general

$$\log n \leq \text{height}(T) \leq n$$

balanced tree

Sec 12.3 : Insertion & Deletion

To insert new node z

- set $z.key = k$
- set $z.parent = z.left = z.right = nil$
- find a node to adopt z as (right or left) child

How?

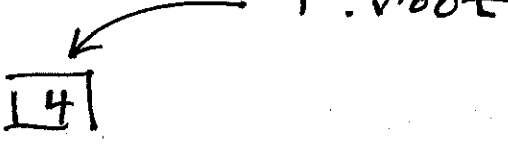
search for key k in T, find nil child where k belongs, place z there.

see pseudo-code for TreeInsert(T, z)

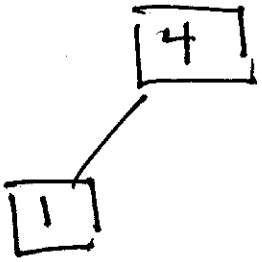
Ex. Keys: 4, 1, 7, 2, 8, 3, 5, 6

T.root = nil, empty state

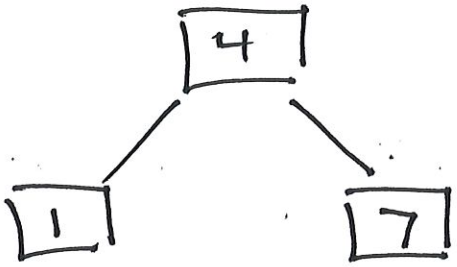
insert(4)



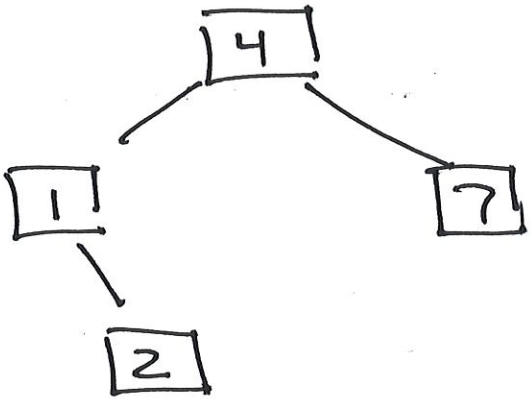
insert(1)



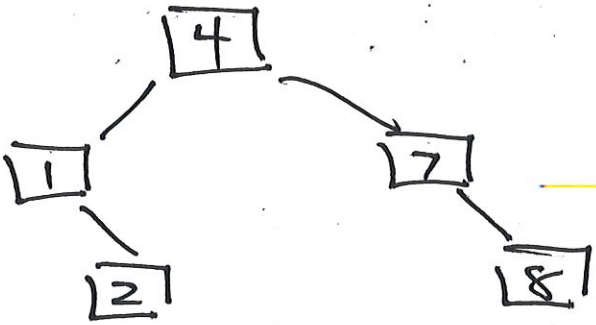
insert(7)



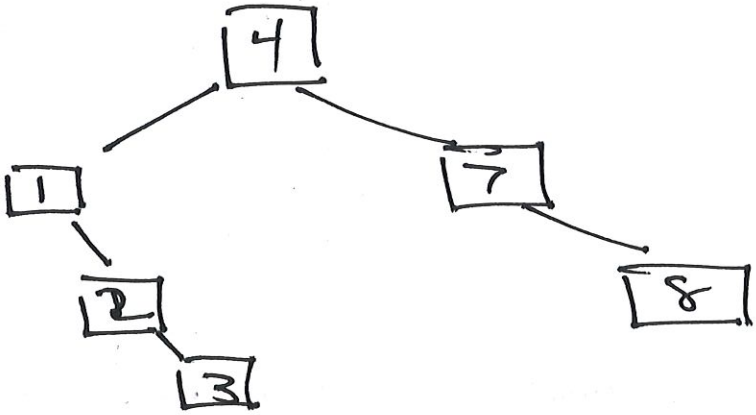
insert(2)



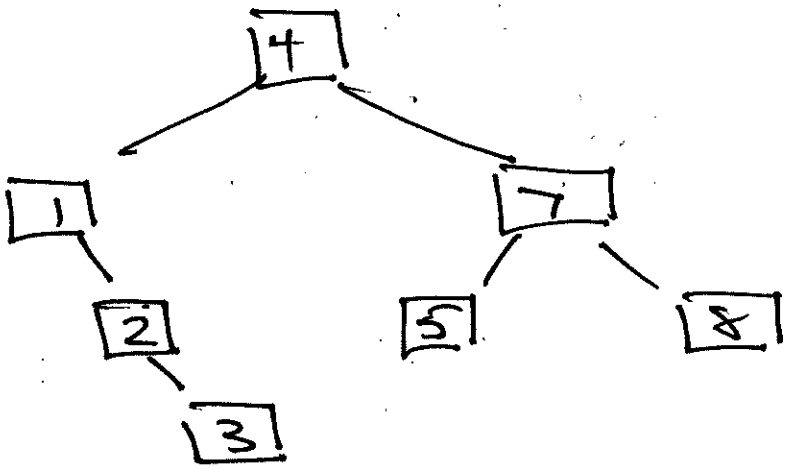
insert(8)



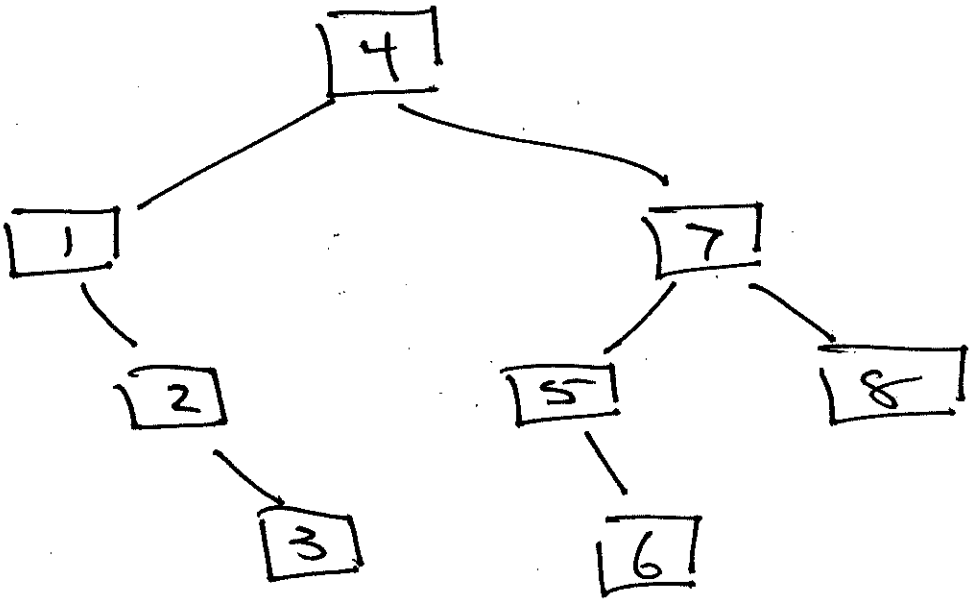
insert(3)



insert(5)



insert(6)



Worst case cost: $\Theta(\text{height}(T))$

Deletion delete z from T

3 cases

case 1 : z has no children

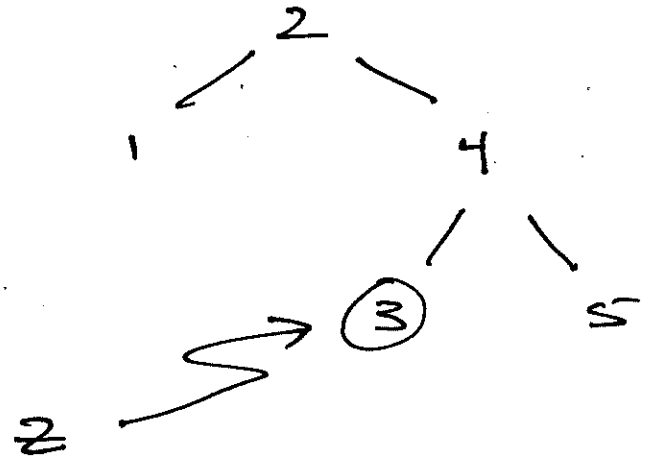
case 2 : z has 1 child

2.1 : z has right, no left

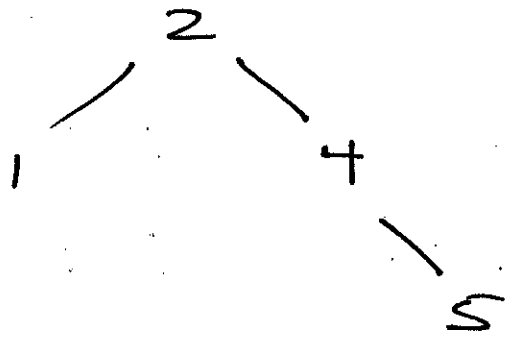
2.2 : z " left, " right

case 3 : z has 2 children.

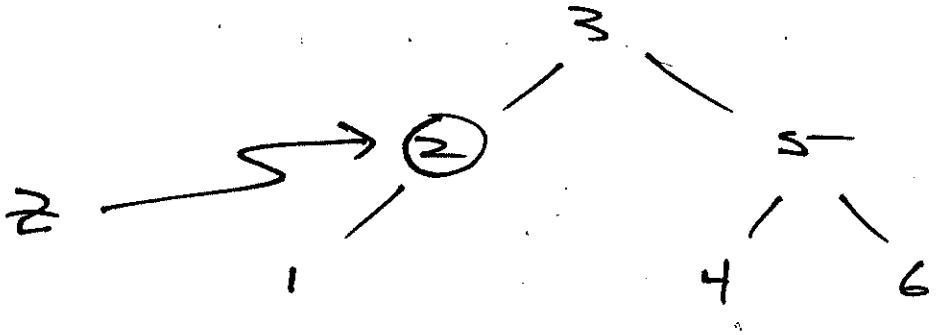
Ex. Case 1 : z is a leaf



delete(3) :

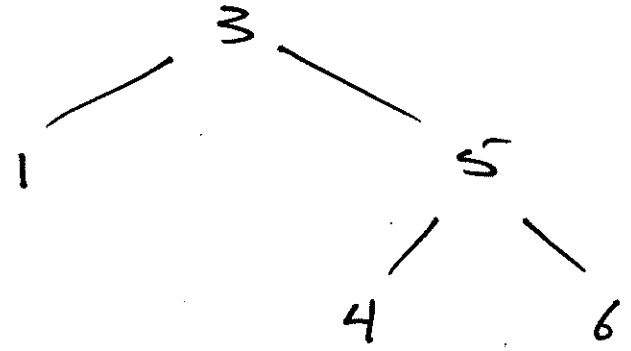


Ex Case 2, subcase 2.2
(left but no right)



splice z out of the 'local' linked list.

delete(2)

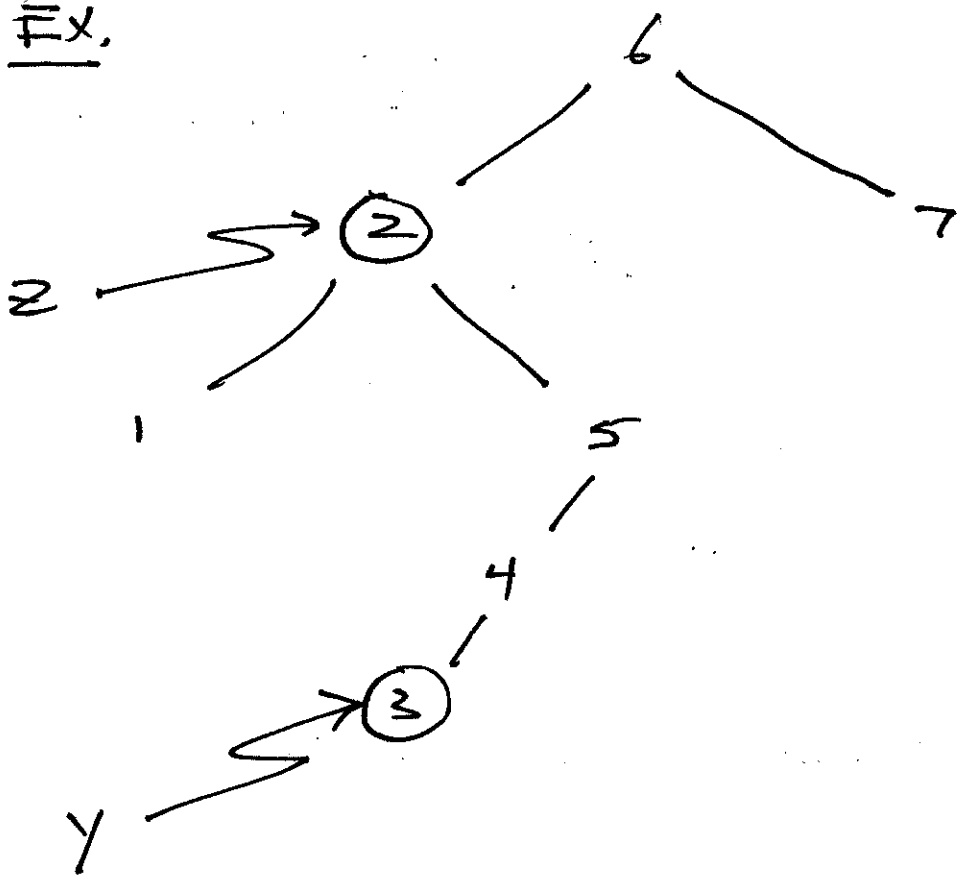


Exercise: do example of 2.1

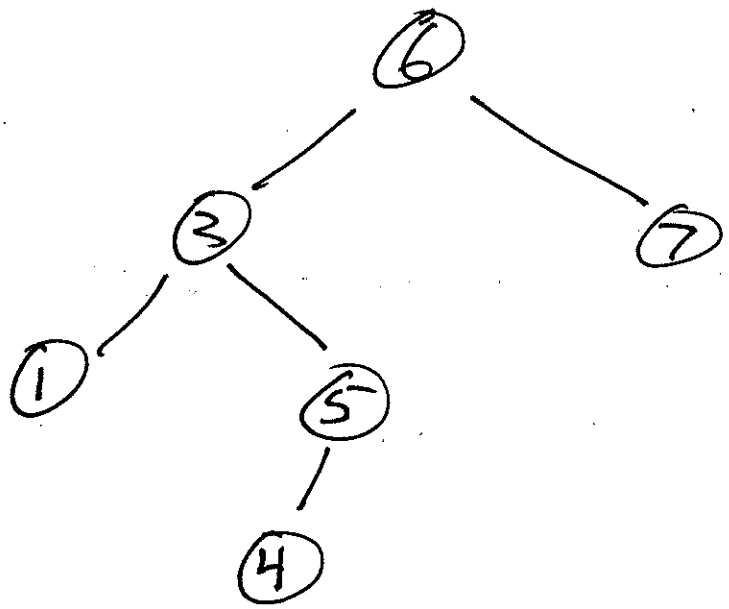
Case 3 z has 2 children

splice out z's successor y (which has no left child) then replace z by y.

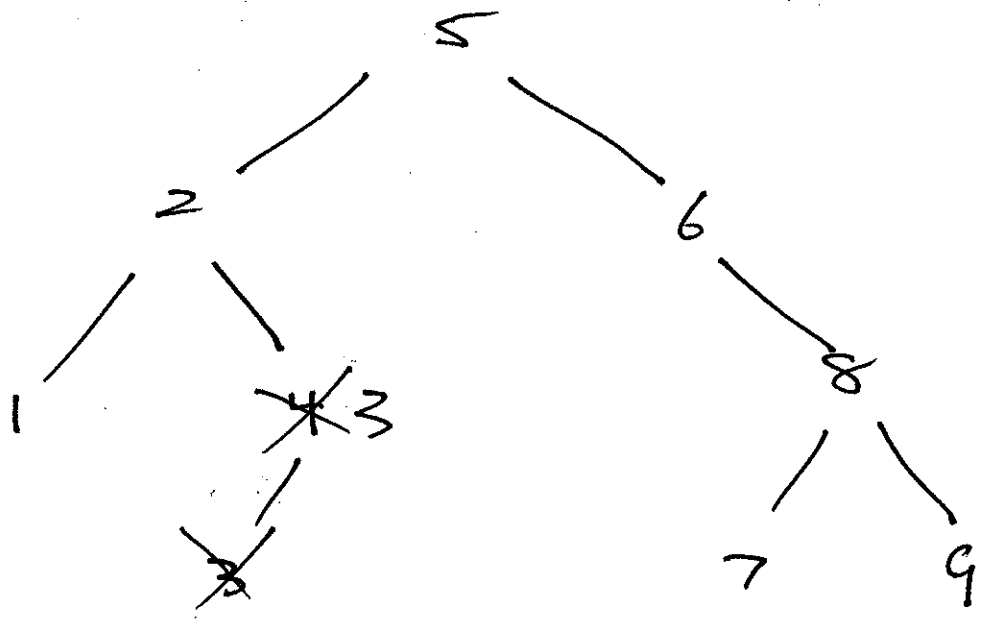
Ex.



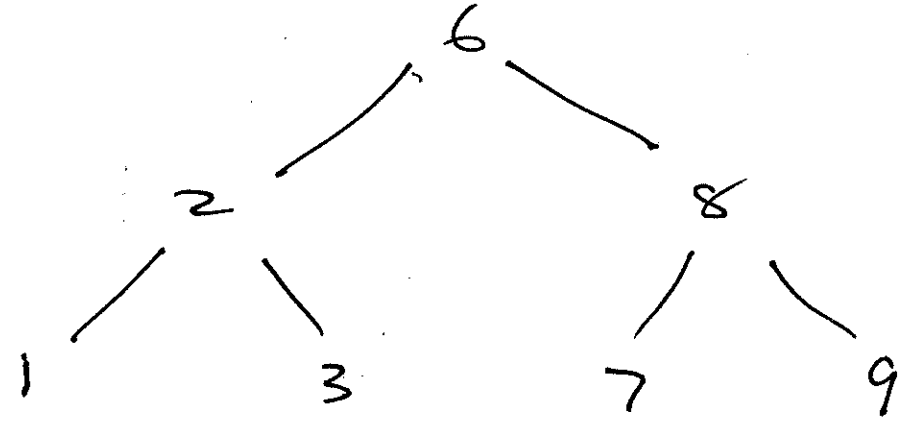
delete(2)



Ex. insert: 5 2 6 1 4 3 8 7 9

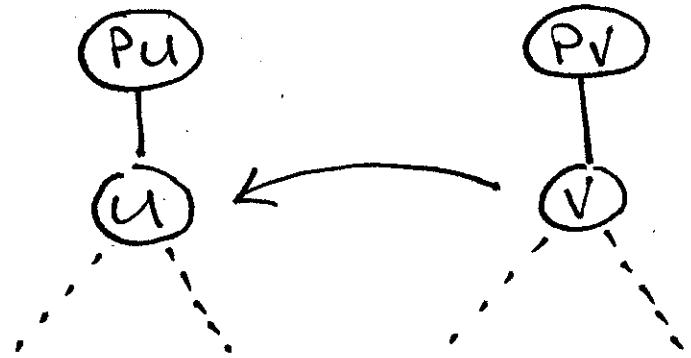


delete: 4, 5

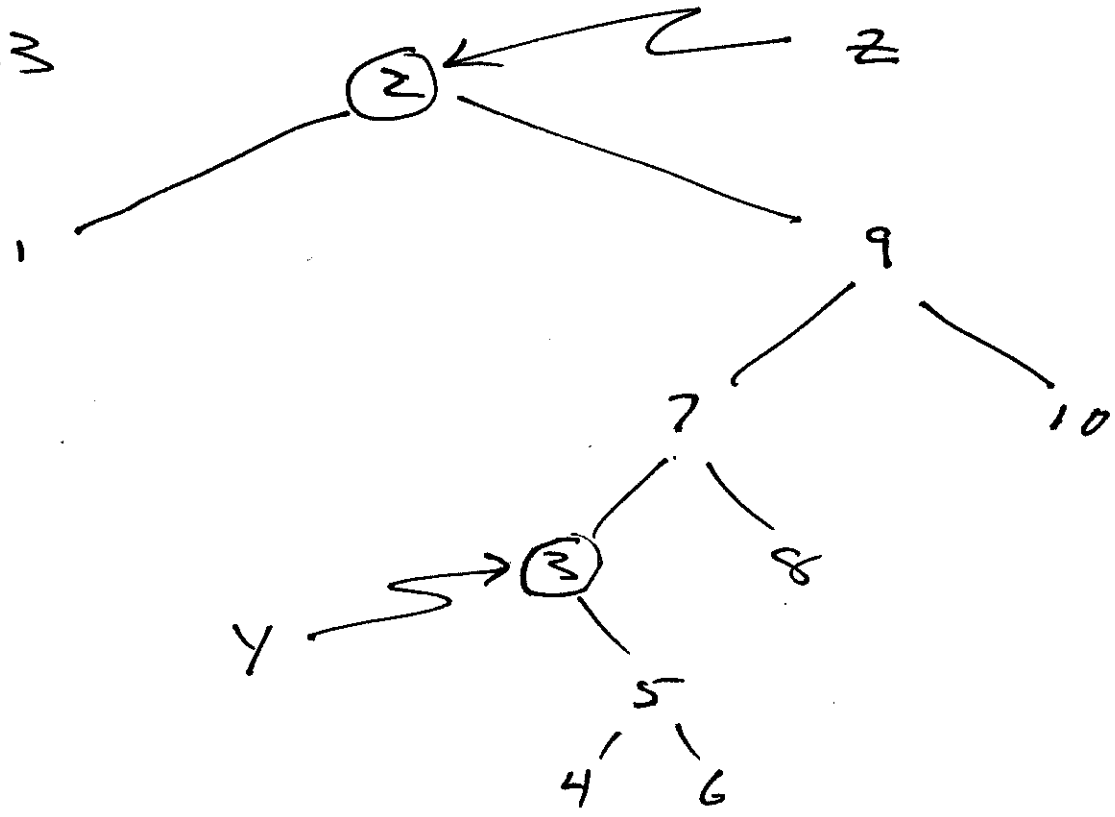


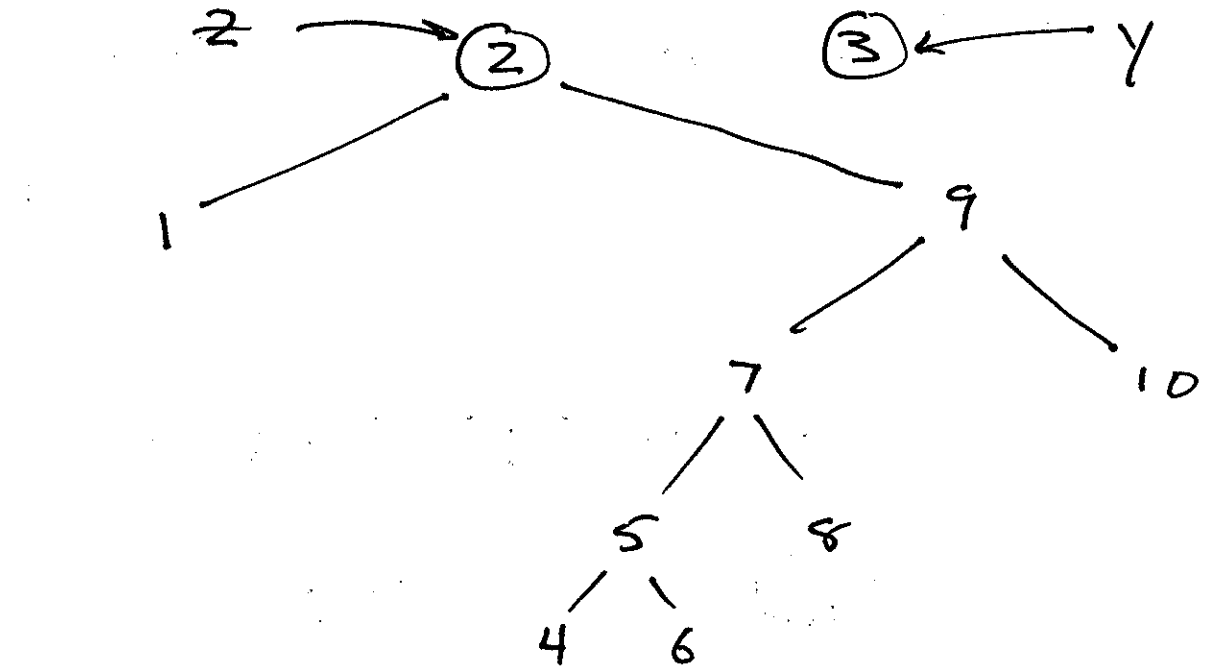
Pseudo-code for delete() in BST.

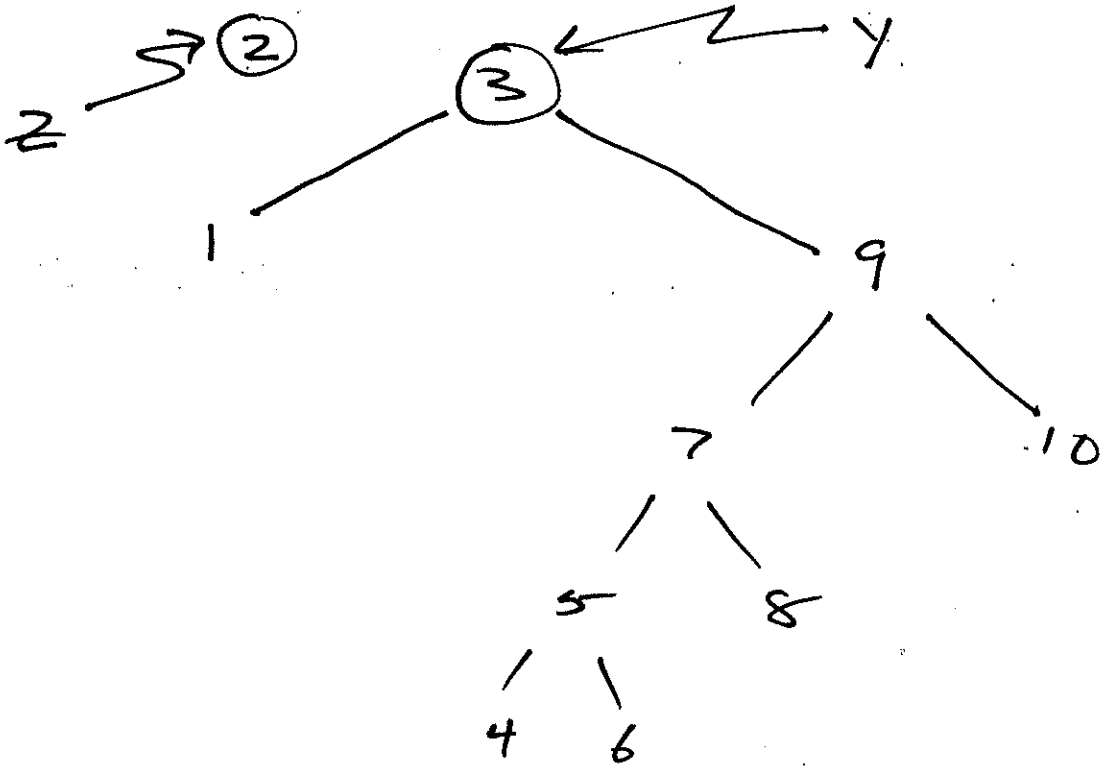
Transplant(T, u, v)



Ex. case3




z
2
3
y
1
9
7
10
5
8
4
6


z
2
z
y
3
1
9
7
10
5
8
4
6