Assignment

P-1
Roll - 1705045
Name - Iftekharr Hakim

① For insert operation, we can prove that at most one single/double rotation is needed. Trivially, when no violation happens after inserting, no rotation is needed.

Case of single rotation:-

The violated subtree part can be pictured as below. Assumed y to be right child without loss of generality.

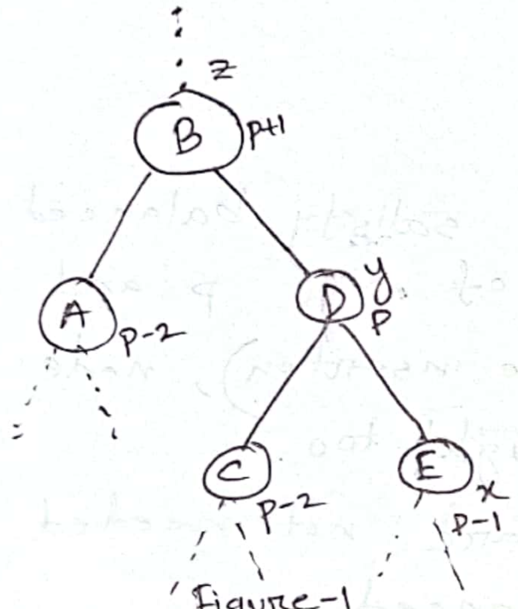


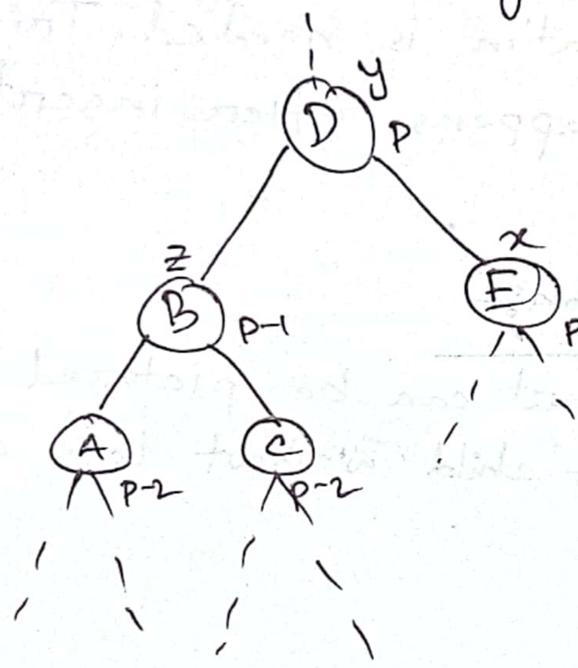
Figure-1

Here, if the height of node z is p+1, height of y is p, height of x is p-1. Height of A has to be p-2. Also, before this insertion, height of node z (or B) has to p. and height of y (or D) has to be p-1.

Note that subtree rooted at A or C were untouched during insertion and height of y were p-1 before insertion, so hight of node C is p-2 in figure-1.

P-2
Roll-1705045
Name-Iftekhar Hakim

Now, if we do a single rotation,

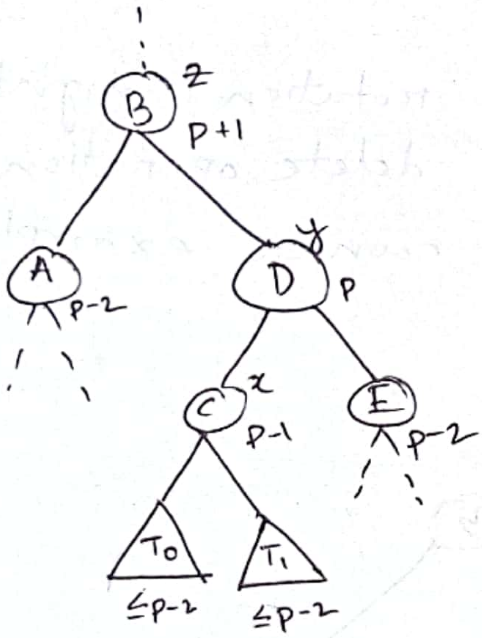


Here, node A, B, C, E satisfy balanced height.
Again, as new height of D is p and height of z was p (before insertion), node D satisfy balanced height too.
So, extra rotations are not needed anymore.
It is completely balanced.

Case of double rotation:-

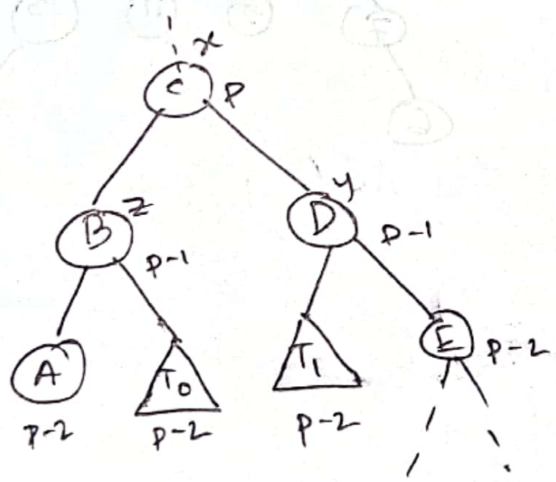
Assuming y to be right child without loss of generality, the violated part can be—

Roll-1705045
Iftekhar Hakim



At least one of T0 and T1 has height p-2. Other heights are written with same reason as before.

Now, if we do rotation(double),



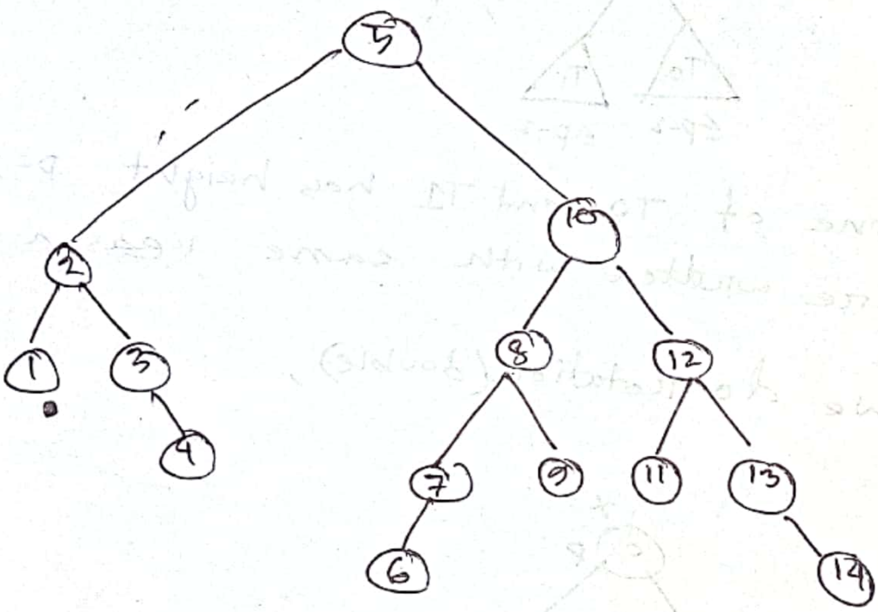
Here, nodes A,B,D,E ~~se~~ are seen to have balanced height. Again, new height of C is p and height (before insertion) of z was p, so node C has balanced ~~weight~~ height too. So, it balanced now.

That's why, more than one rotation is not needed.

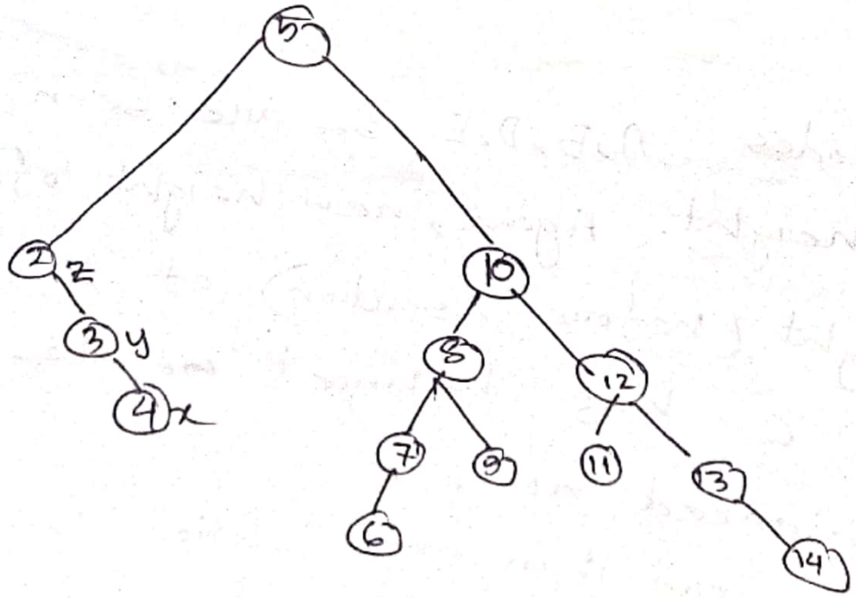
P-4
Roll- 1705045
Iftekhar Hakim

(2) More than one rotation might be necessary after delete operation. We can prove it with counter example.

Initial tree:-

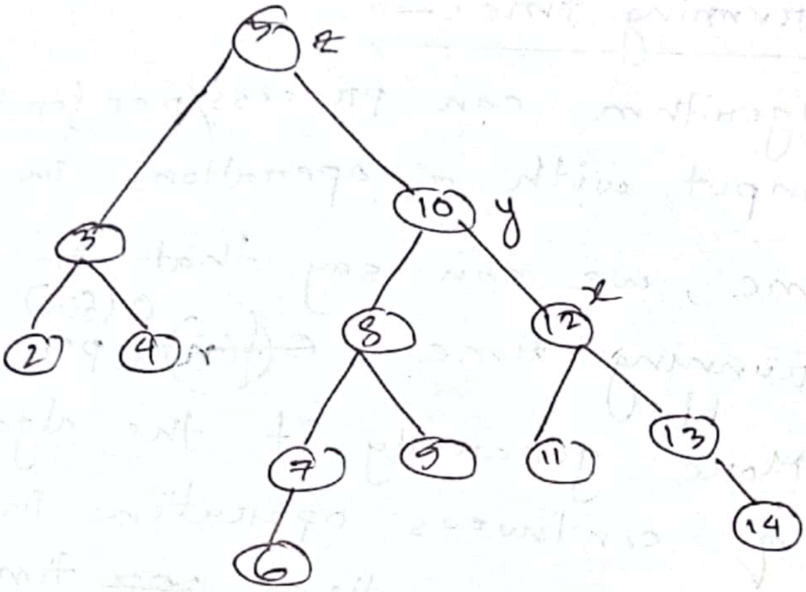


Deleting 1,

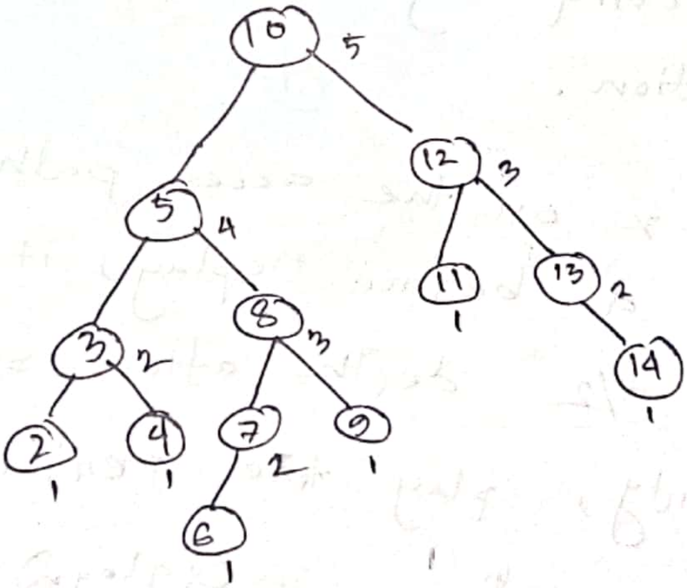


P-5
Roll-1705045
Iftekhar Hakim

Doing single rotate,



Still not balanced,
Doing single rotate,



It is now balanced.
By this example, we can say that more than one rotation can be necessary for delete operation

P-6
Roll-1705045
Iftekhar Hakim

(3)

Amortized running time:-

If an algorithm can process/perform any valid input with q operations in $O(qf(n))$ time, we can say that it has amortized running time ~~$\Theta(f(n))$~~ $O(f(n))$ per operation. More generally, if the algorithm performs q continuous operation in a time complexity, amortized ~~opr~~ time complexity is said to be the average time per operation. But it does not bound the time complexity for a single and lone operation.

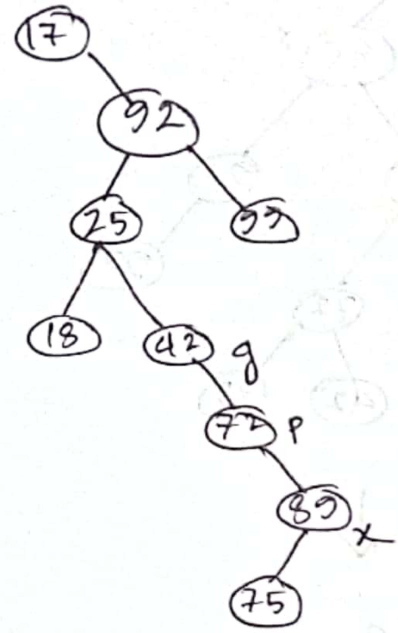
For splay tree,

If a node x on the access path is at depth d before splay, it is at about $d/2$ depth after splay. With this property, splay tree perform q continuous operation in $O(q\log n)$ time. That's why it has $O(\log n)$ amortized time complexity per operation.

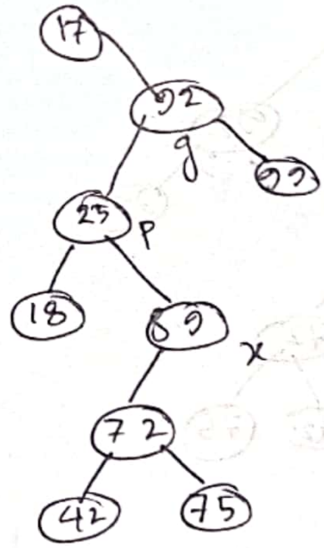
P-7
Roll-1705045
Iftekhar Hakim

4.

~~1st Rotation:~~

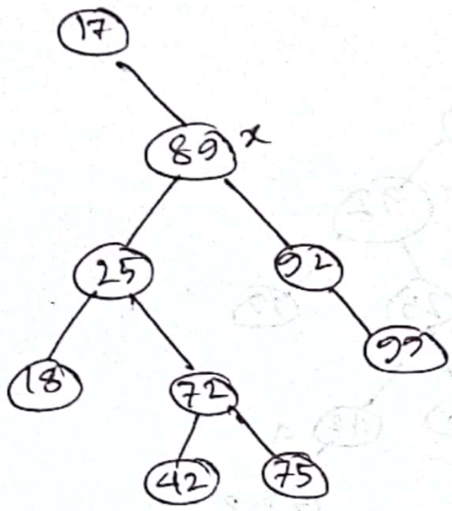


After rotation-1 :-

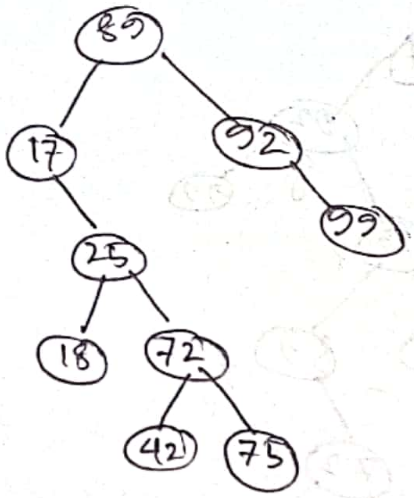


P-8
Roll-1705045
Iftekhar Hakim

After rotation-2,



After rotation 3,



Now, 89 is at the root.

So, 3 rotations are needed.