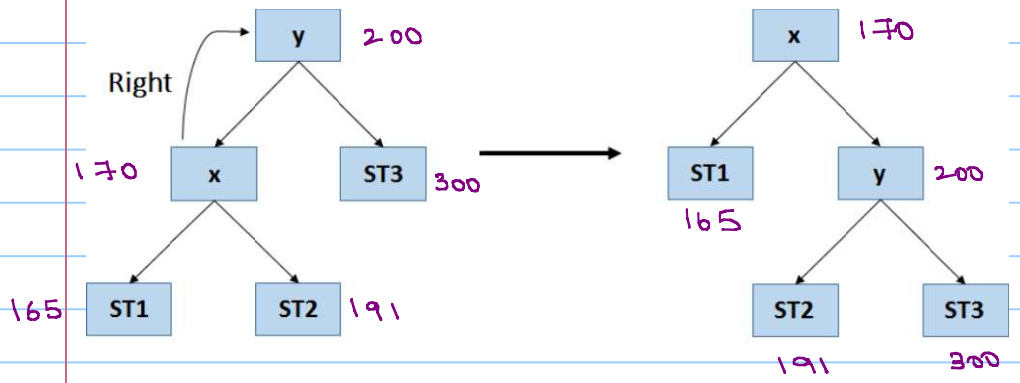
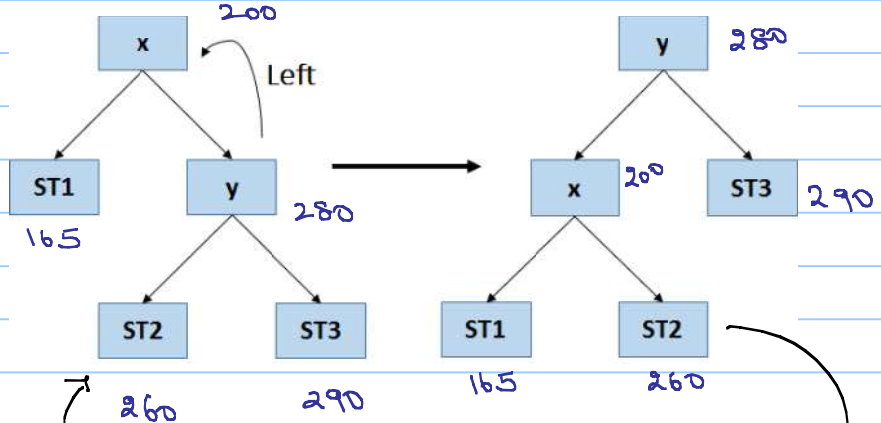
CSE 3010 – Lecture 41 – Self balancing Tree – Contd.

Note Title 3/4/2020





When a rotation is performed:

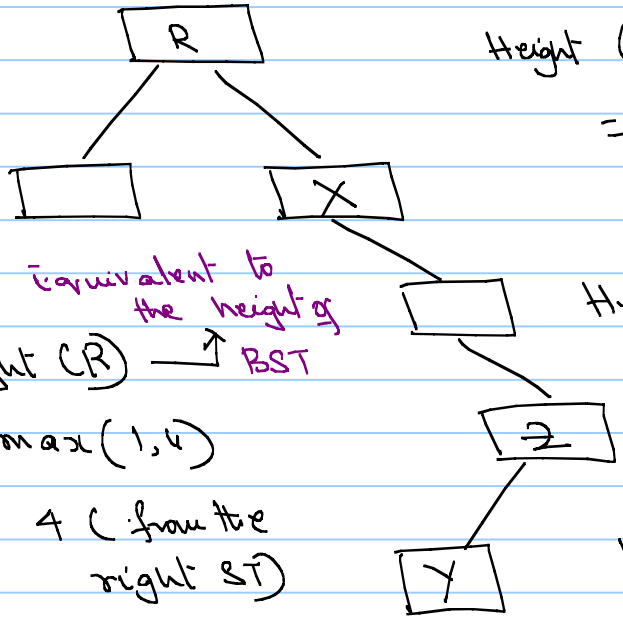
- ↳ root of the tree may change
- ↳ child nodes may get new parents

ST2(260) → parent = y(280)

ST2(260) → parent = x(200) (new)

(if parent maintained, adjustment of parent must be done)

Height of a tree



equivalent to the height of BST

Height (R) = max (1, 4)

= 4 (from the right ST)

Height (X)

= max (0, 3)

= 3

Height (Z)

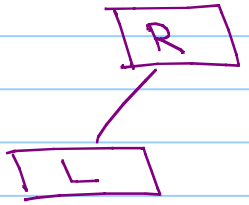
= max (1, 0)

= 1

Height (Y)

= max (0, 0) = 0

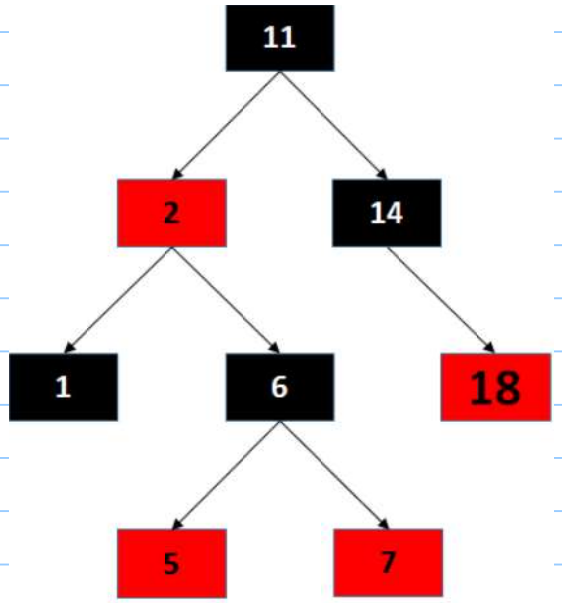
Algorithm will return 1



1 + max (left subtree) = L

1 + (max (left subtree) = NULL

= 2

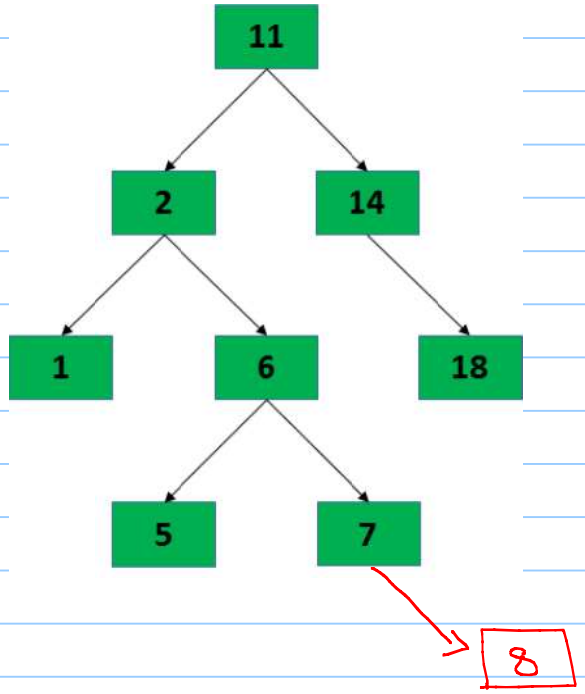


Red-black tree

↳ the node has an additional property of color of node

```
typedef struct rbnode {
    ITEM key;
    struct rbnode * left;
          "       * right;
          "       * parent;
          int color; or char color[10];
} RBNODE;
```

This is important property of a red-black tree



Balance factor

bf (root) = bf (11) = height (14) - height (2)

= 1 - 2

= -1 ✓

bf (6) = height (5) - height (7)

0 - 0

= 0 ✓

if balance factor is > 1 → left rotation

if balance factor is < -1 → right rotation