Review RB

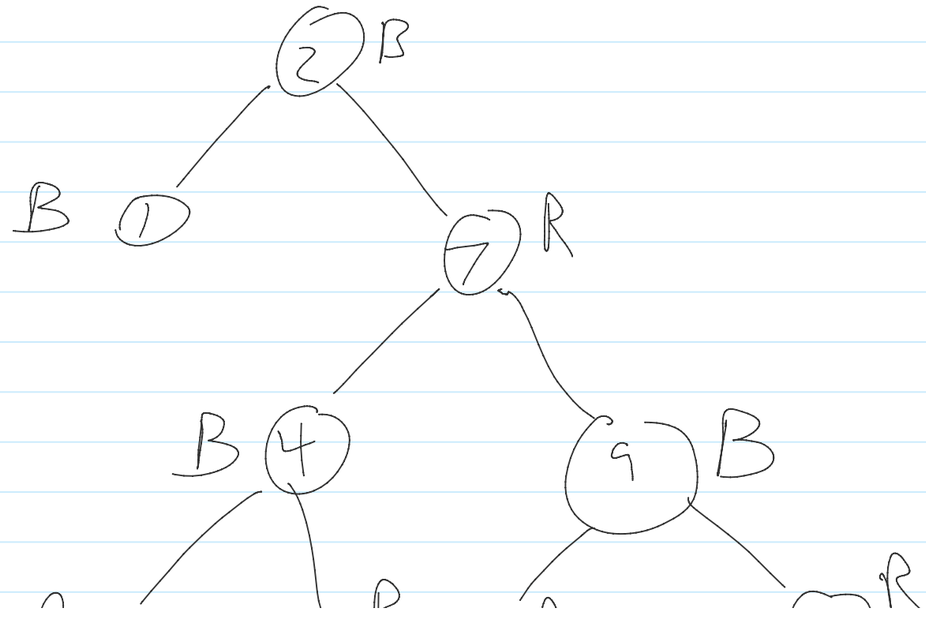
Sunday, February 11, 2018 9:16 PM

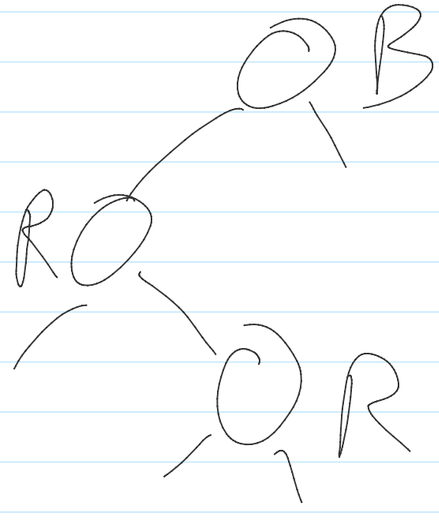
A Red Black tree is a binary search Tree

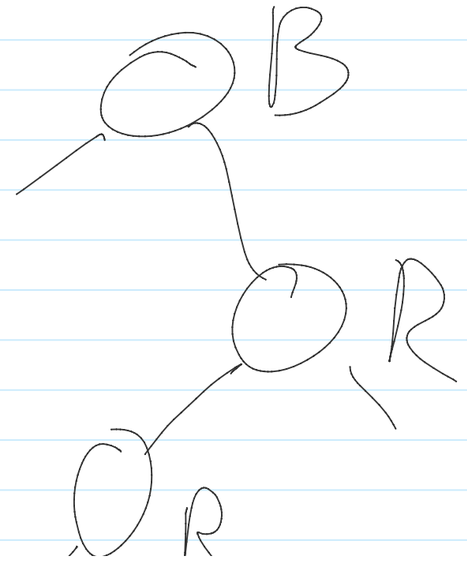
Rules

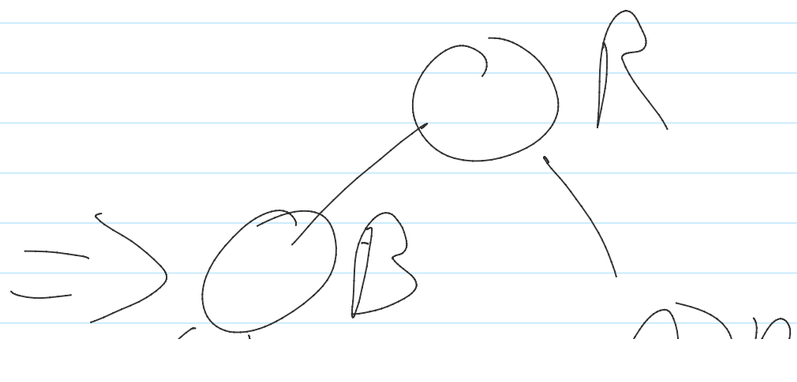
1. Each node is Red or Black
2. Each leaf node is black (null leaf nodes)
3. Each red node has no red child
4. Each path from Root to leaf has an equal # of blacknodes
5. Root of the tree is black (optional)
   (on test not optional)

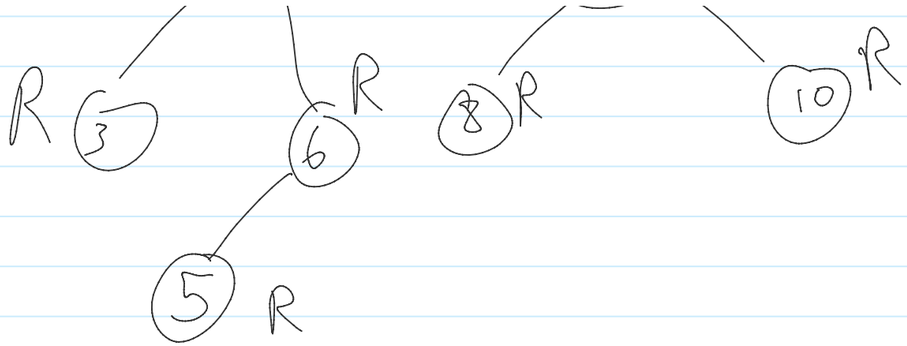
Balance this R-B tree

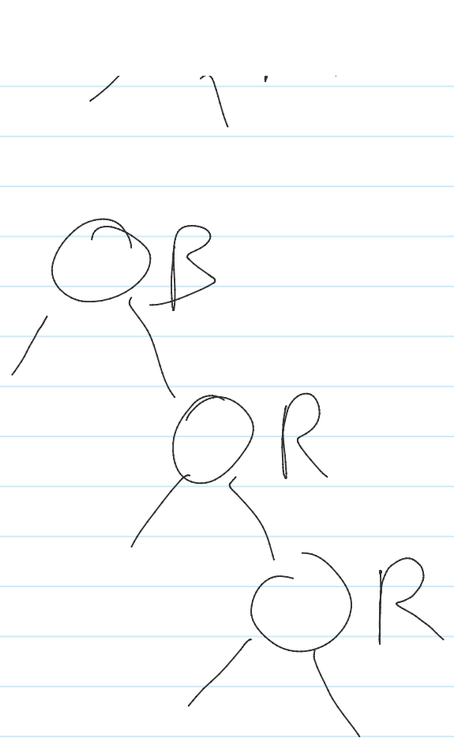


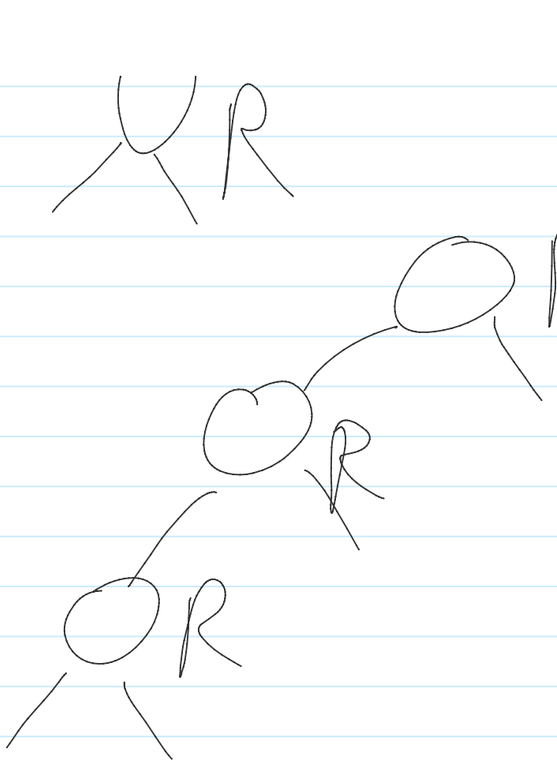


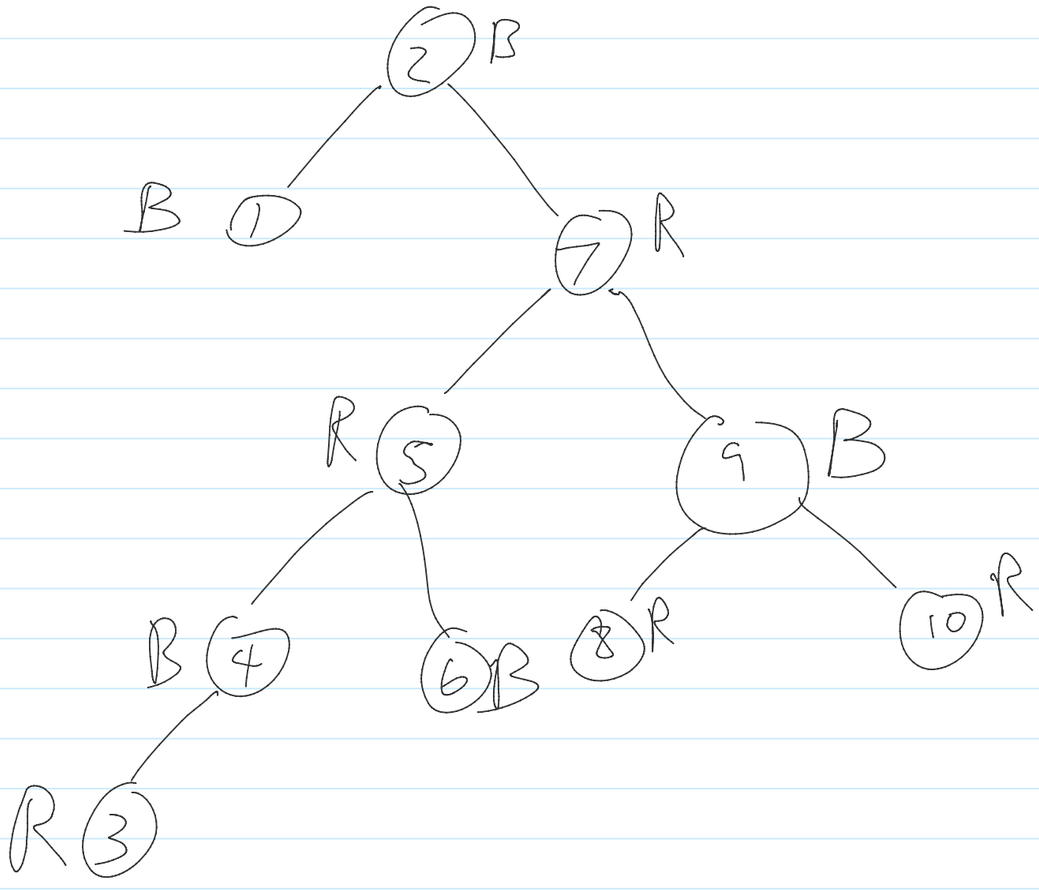



R
B

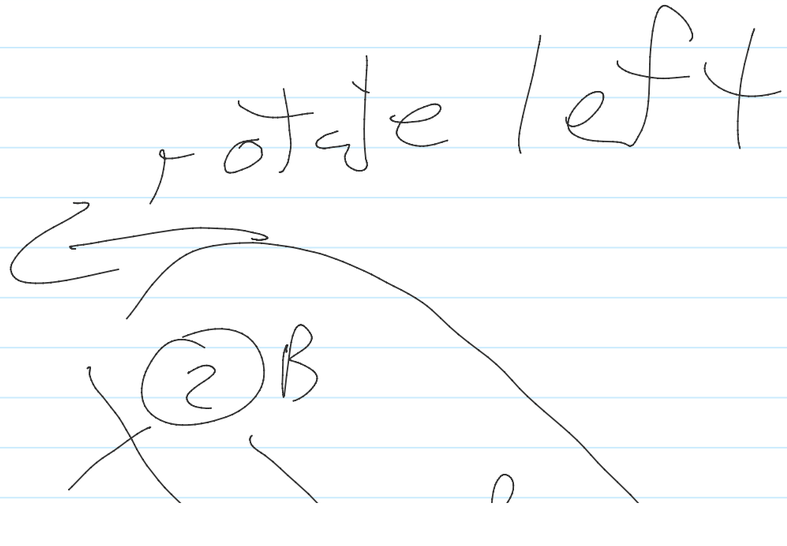

R
3
6
R
8
R
10
R
5
R

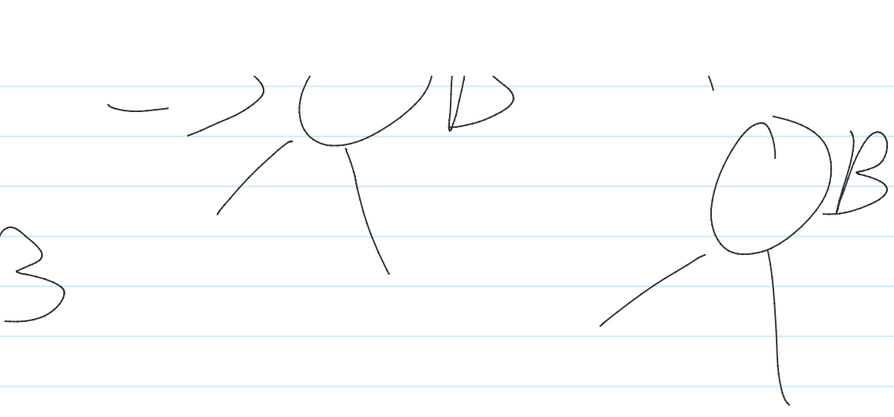

B
R
R

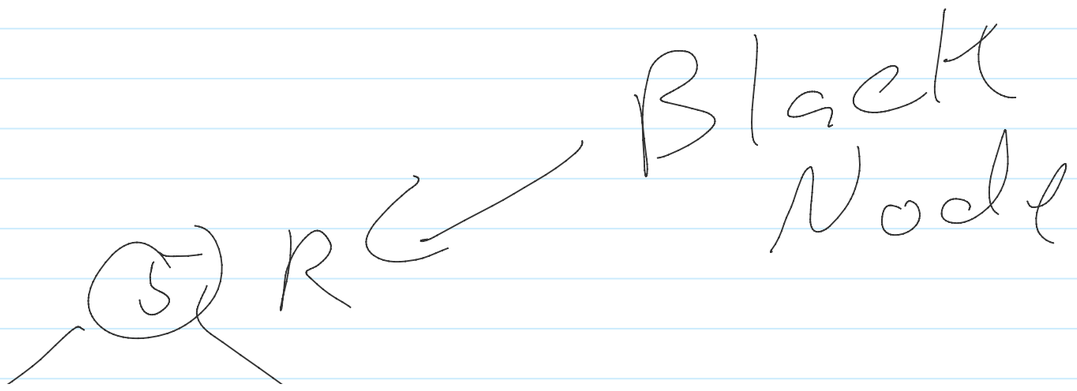

R
R
R

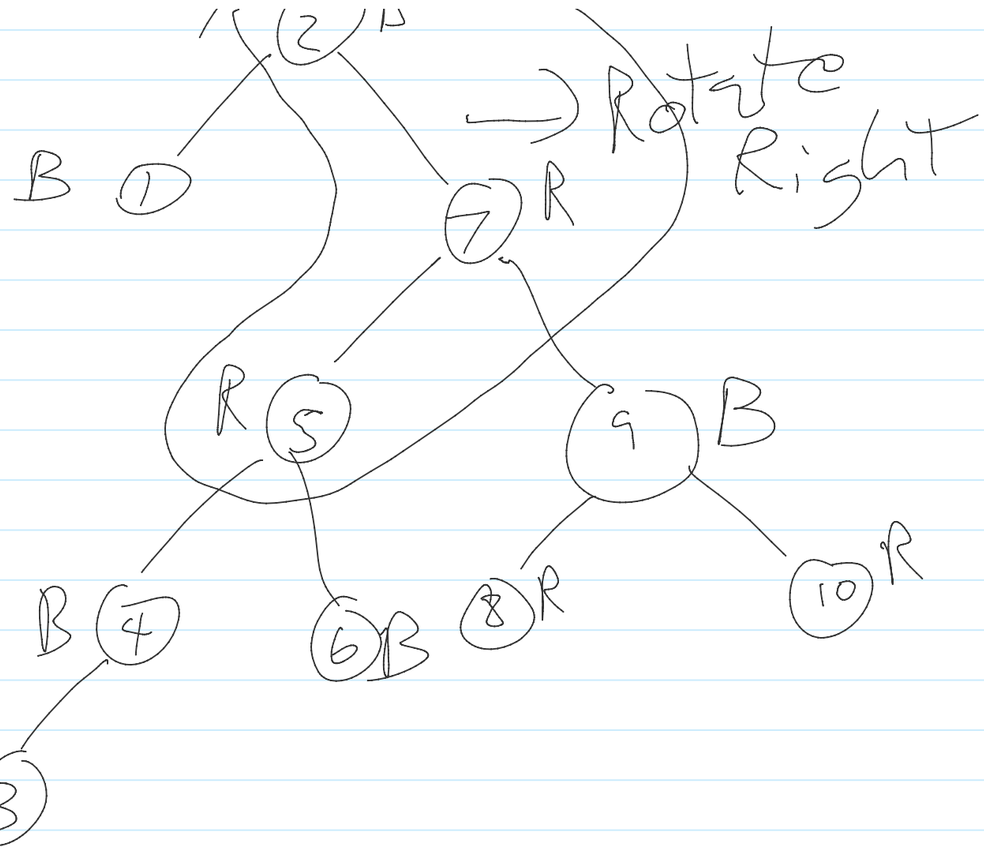

2
B
B
1
7
R
R
5
9
B
B
4
6
B
8
R
10
R
R
3

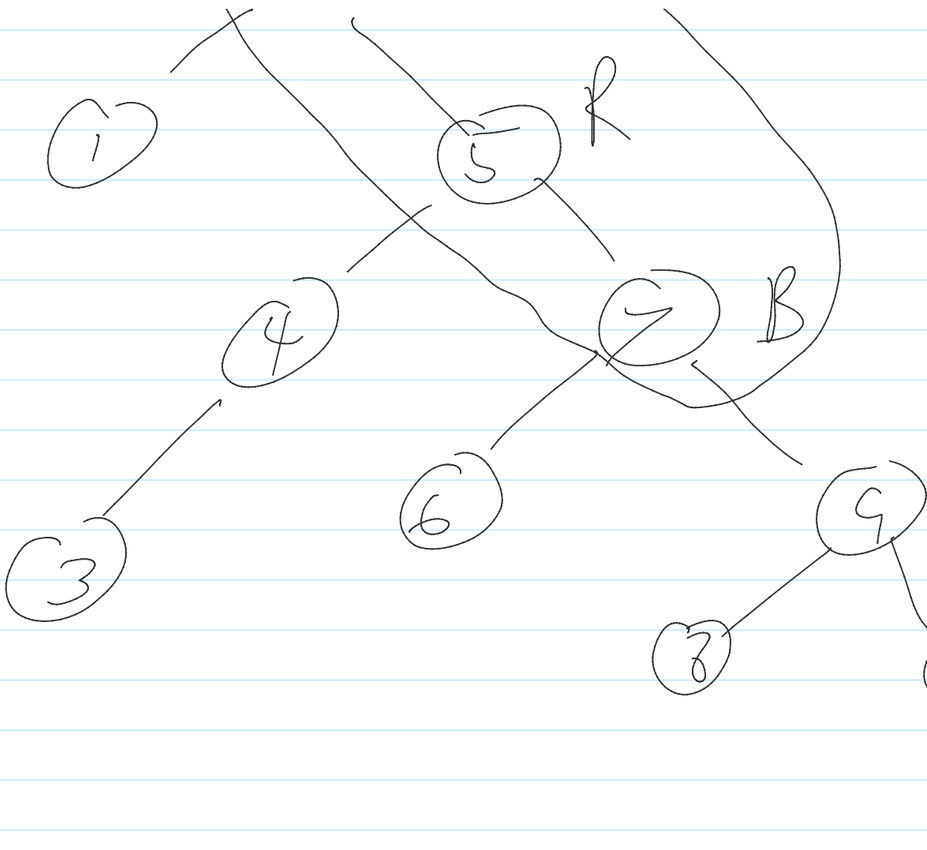

2
B


rotate left
2
B

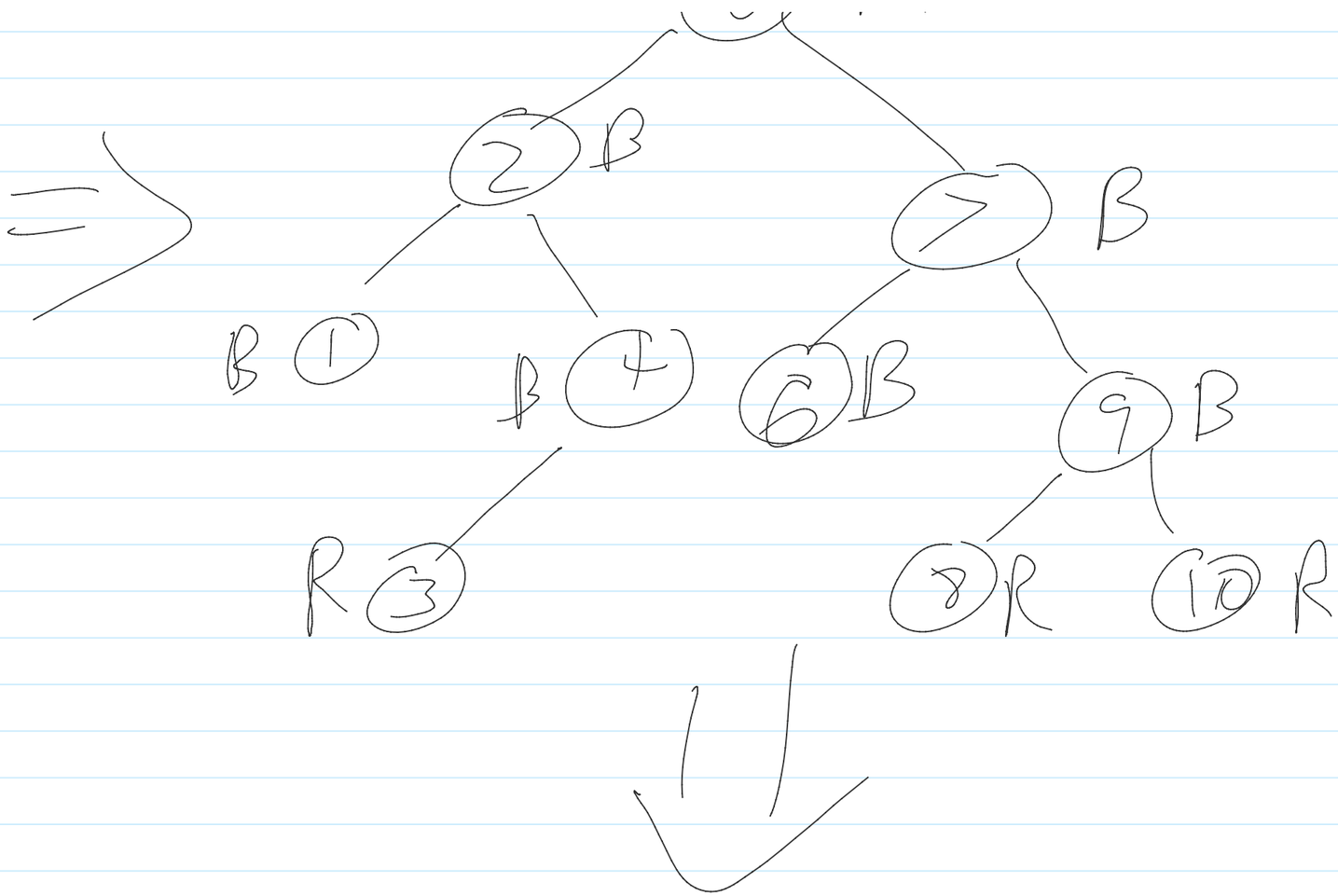


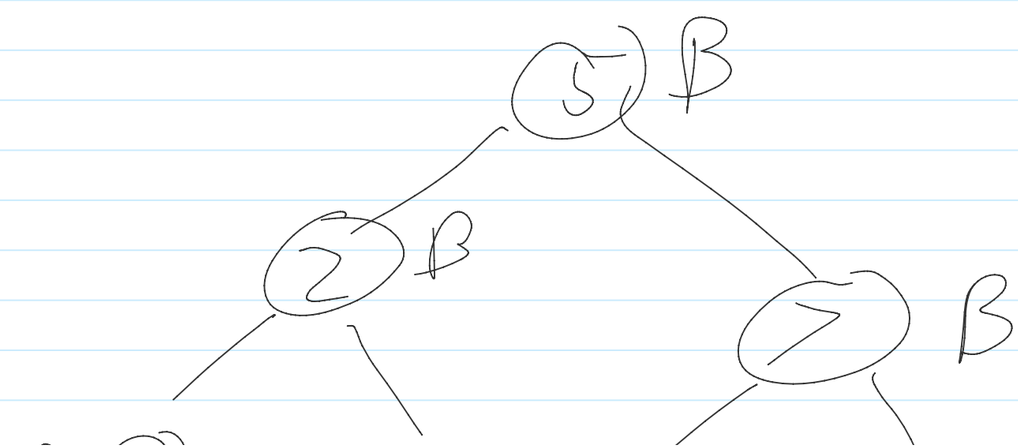

S
R
Black
Node



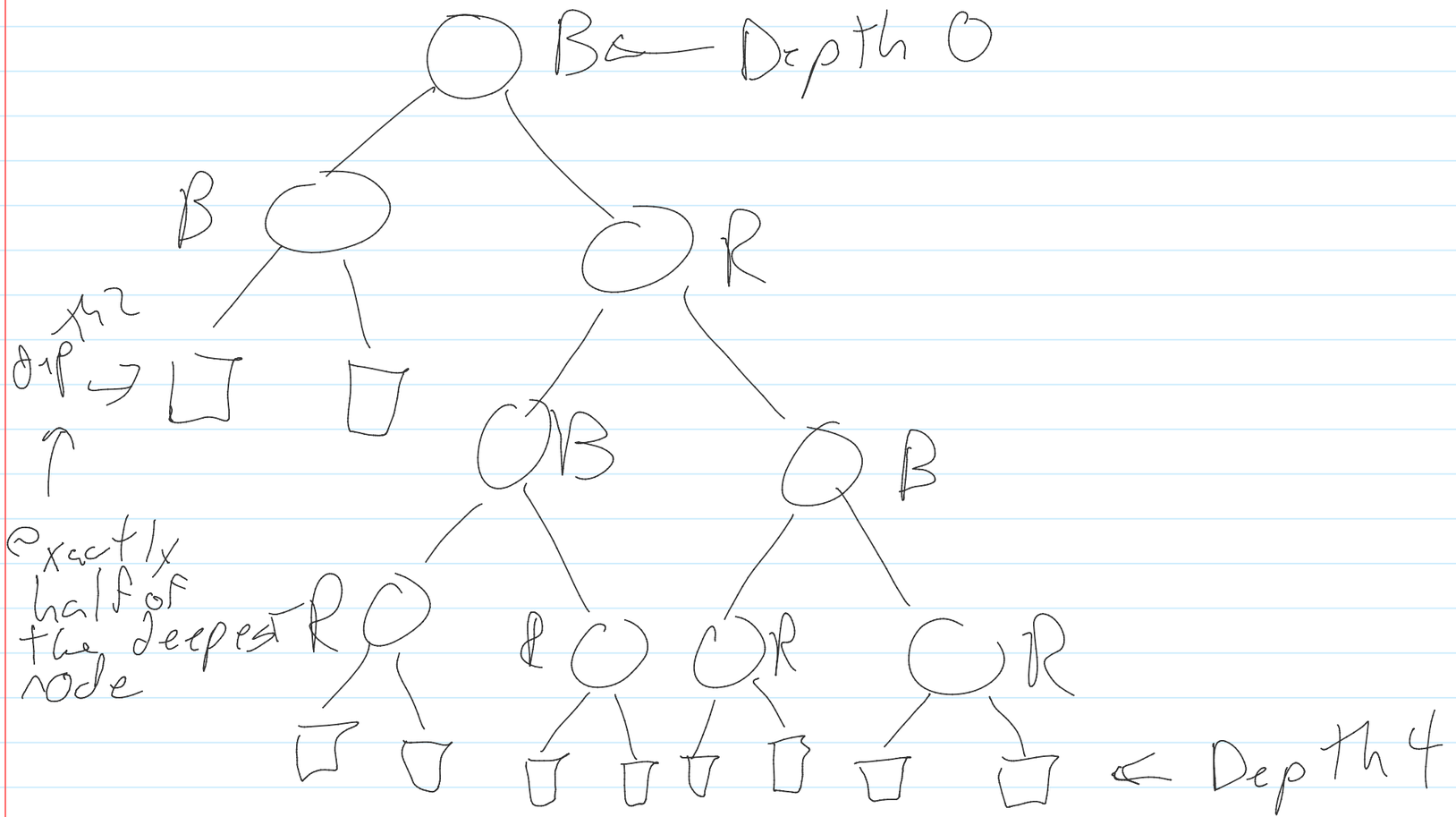


Prove or disprove
The depth of the most shallow leaf node is at least 1/2 the depth of the most deep leaf node

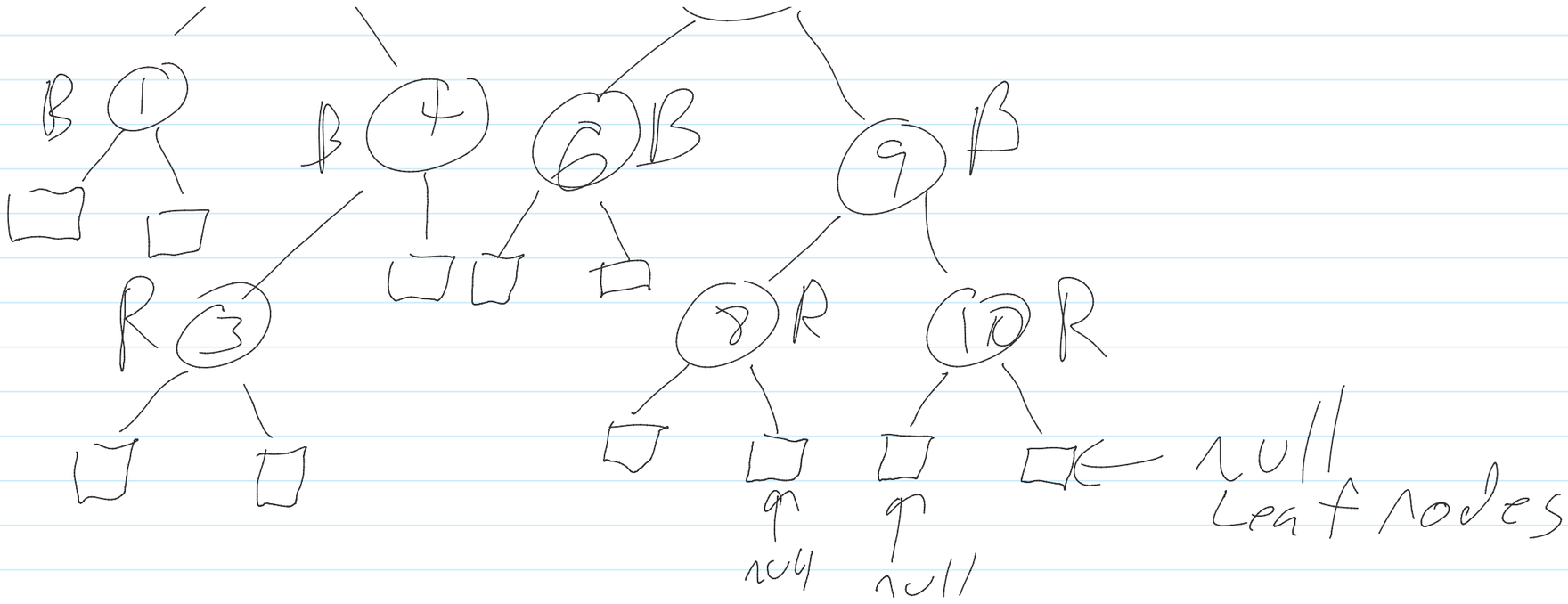

2 B
7 B
B 1
B 4
6 B
9 B
R 3
8 R
10 R
10


5 B
2 B
7 B

depth of the most deep leaf node



Consider the shallowest path
It must be all Black nodes
otherwise we could remove the red


B
1
B
4
6
B
9
B
R
3
8
R
10
R
null
null
null
Leaf nodes

otherwise we could remove the red node and make a shorter path

the other (longest) path would alternate red and black nodes

Since #B in Short = #of B in long

#of Nodes in Short = S then Depth = S - 1

B/C alternating #of Nodes in long = S + S - 1 Depth = S + S - 1 - 1

$= 2(S-1)$

This shows the longest Path cannot be more than twice the shortest path ▨