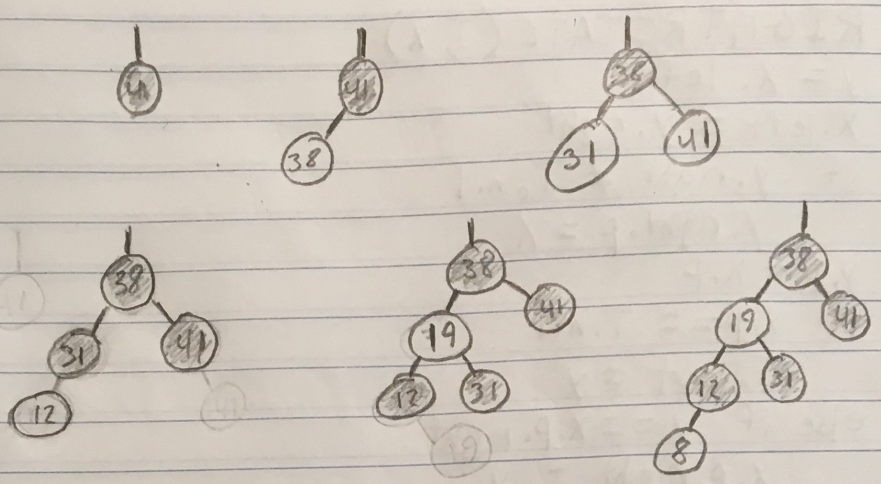
Jake Schinto

● = black
○ = red

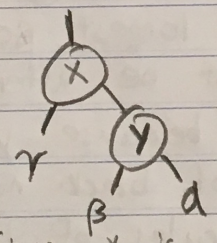
Homework #4

1.



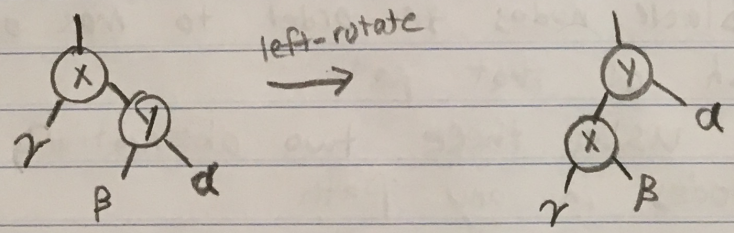
2. • Shortest path would be only black nodes S = black nodes
• Longest path would be black and red alternating, so for every black node there is one more red node or L = 2S
• L/S = 2S/S = 2
• For any tree L/S ≤ 2

3. a.



• if X = red
• Y and γ must be black
• β and α could be either depending on what is in the subtrees
• Since X is red, both children must be black
• There is no rule for black parents, so the children could be either

b.

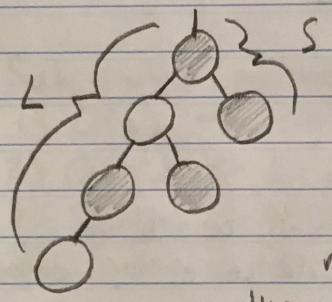


4. This is impossible because while an all black tree would be a valid tree, there would have to be other trees that contain at least one red that works as well and would be used due to preferance of red nodes in the insert method.

5. RIGHTROTATE(T, x)

```
1.  y = x.left
2.  x.left = y.right
3.  if y.right ≠ T.nil
4.      y.right.p = x
5.  y.p = x.p
6.  if x.p == T.nil
7.      T.root = y
8.  else if x == x.p.right
9.      x.p.right = y
10. else
11.     x.p.left = y
12. y.right = x
13. x.p = y
```

6.



- In order for Longest path to be the longest possible path it must be alternating red and black, because you want the least amount of black nodes possible without breaking the rules of the tree
- For the shortest path, it would need to be all black nodes in order to max out the required bh of that path
- Therefore, using these two observations

S = black nodes in any path

L = black nodes in any path + each red node since there is a red for every black in L,

L = black nodes in any path x 2

- So, by dividing L/S it gives 2/1 = (2)
  - Or L is at most 2x as long as S