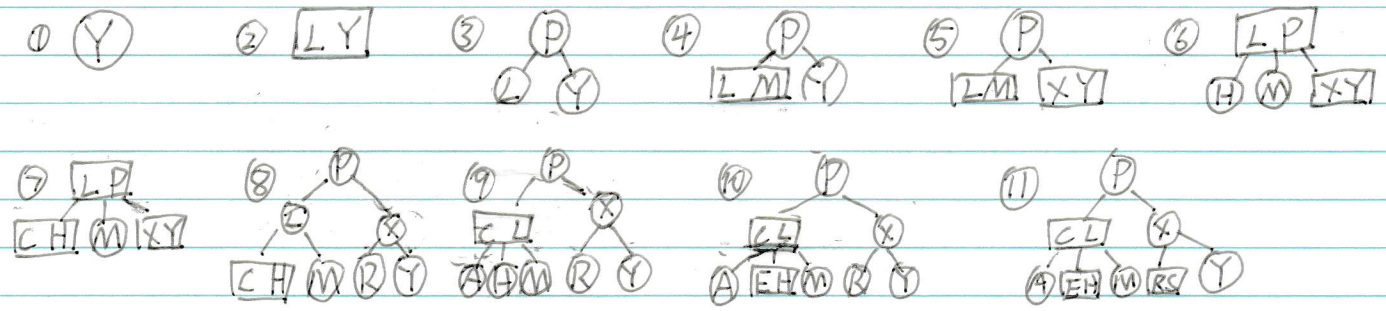
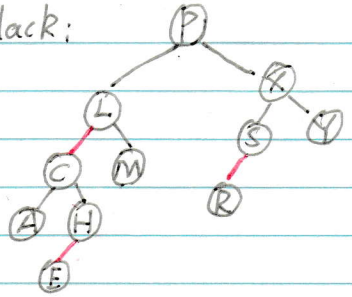
CSC 226 Assignment 2
Nelson Dai (V00815253)
Oct 12/2016

1. 


2. Suppose the AVL tree T has height h, after inserting k we get an unbalanced tree which means the node x with key k at height -1. Now in S' we have node z of height 1 y of height 0 x of hight -1. After the two rotations the height of x increas by 2: -1+2=1, the height of z decrease by 1: 1-1=0, and the height of y keeps the same as 0. Then the height of $S^*$ is 1 which is same as S.

red-black:




3. The minimum number is 0 since it would be a sorted sequence with no inversion
The maximum number of inversions in a list with n elements is $(n-1)+(n-2)+\cdots(n-n)$.
$= 1+2+3+\cdots+(n-1) = \frac{(n-1)(n+1-1)}{2} = \frac{(n-1)n}{2}$ or $0.5(n-1)n$.

4. By using red-black to track the sum of inversions, we add each element in the permutation into the tree. whenever a node is attached to the left subtree, we add the amount of nodes to the right of it to the sum after doing any rotation needed (size of right subtree+1) After the tree is complete, the sum will equal the number of inversions. Since we are going to call put() n times and Put() run in O(logn) and since the red-black tree is balanced binary tree, it requires no extra runing time. The running time is O(nlogn) the change will look like this:

```
if (cmp<0){
    h.left = put(h.left, key, val);
    inversions += size(h.right)+1;}
```

5. By doing the experiment I found out that the percentage of red edge in a red-black tree with random keys is always between 25% - 25.5%. But in realiey we know we can have no red edge in the tree or a red edge follows every black edge, which means the range of the percentage of red edge is 0 to 50% but the expect percentage should be 25% to 25.5%. → $\frac{n}{4}$%