HOMEWOEK 3

Runlin Hou
ECE, School Of Graduate Studies
Rutgers University
hourunlinxa@gmail.com

Question 1

Since we are supposed to implemente the symbel table based on the 2-3 tree, we should implemente a 2-3 tree first. In this section, I will implemente a red-black tree to fullfill our requirment.

For each node, we set the following attributes. `key, value, color, l_child, r_child, parent, tomb.`

According to the requirment, we don't have to maintian the balance when we insert a new node to the red node. But still, the tree need some basic functions.

- `isEmpty()` can help to decide if this tree is empty.
- `search()` returns a pack of two values, the fisrt value is a node and the second value is a boolean
- `insert()` changes the value of an existed key and insert a new node if the key not existed
- `delete()` mark a node to be tomb

Question 2

The average path length of sorted insertion and random insertion

node	1	2	4	8	16	32	64	128	256	512
srt	1	2	4	8	16	32	64	128	256	512
rnd	1.0	2.0	4.0	3.6	4.8	7.7	7.8	8.8	9.9	11.2

As we can see that, sorted insertion always have only one leaf node at bottom. So the path length always equals to the amount of nodes.

As for random insertion, since the nodes will not be inserted in order, the tree will show a more flat pattern. And the length can be fit to $log_2 N$, where N=number of nodes.

Question 3

To be more ascurate, I do not use the unbalanced tree in Q1. Instead, I write two python packages `LLRB_tree` and `RB_tree` to create the Left-Lean-Red-Black Tree and Red-Black Tree, since these are two different kinds of RB_tree with a significant rule being different. The LLRB tree can only have red node on its left child. So I assume the RB tree will have much more red nodes than LLRB tree. I took 100 trails for ench amount of node and then take the average value of the red nodes contained in each tree. And the result is shown as below.

Total Nodes	10^4	10^5	10^6
Red Nodes	5706	57297	572967
LLRed Nodes	2572	25465	254064

As we can see, the red node contained in each tree shows a linear relatoinship with the total amount of nodes. The ragular Red-black Tree has 57% red nodes and Left-Lean-Red-Black Tree has 25% red nodes.

Question 4

For this section, I also use package `LLRB_tree` to implemente the Left-Lean-Red-Black Tree. To save some time, I set the step size to be 500 and run each trail 1000 times to get the average avlue. Following are parts of the result, as we can see, since we insert the nodes randomly, the deviations are quite small and the tree are actually shows a flat pattern. The depth are close to $log_2 N$.

Nodes	500	1000	2000	4000	6000	8000	10000
Avg	8.2	9.2	10.2	11.3	11.8	12.3	12.6
Std	1.60	1.63	1.66	1.69	1.70	1.71	1.72

Question 5

For $rank(k)$, it returns the amount of keys that is smaller than k. For $select(k)$, it returns the key that have k keys smaller than it.

It is easy to find out that the given dataset is from 1 to 1000, so I assume the result would be

$rank(7) = 6$ and $select(7) = 8$

program shows the same result

rank(7) = 6

select(7) = 8