Name : Ansari M.Saeem M.Saleem

Uid : 2019430001

Subject : AAC

Expt no : 2

Aim : Write a program to implement & Analysis Red Black Tree and to store the details of product (Id,Cost,Qty)

Aim:

Write a program to implement & Analysis RBT and to store the details of product (id,cost,qty).

Objectives:

- Initialization Red Black Tree
- Perform primitive operations of RBT such as Insertion, Deletion and searching.
- Display the details of products (id,cost,qty) in RBT format.

Methodology:

A red–black tree is a kind of self-balancing binary search tree. Each node in RBT has an extra bit as compare to BST, and that bit is color (red or black) of the node.

Why RBT ?

Most of the BST operations (e.g., search, max, min, insert, delete.. etc) take O(h) time where h is the height of the BST. If we make sure that height of the tree remains O(Logn) after every insertion and deletion, then we can guarantee an upper bound of O(Logn) for all these operations. The height of a Red-Black tree is always O(Logn) where n is the number of nodes in the tree.

Properties:

- Binary Search Tree : every RBT should be binary search tree
- Color Property : every node is either red or black
- Root Node : root node of RBT is always black
- Internal Node : If a node is Red then its both children are black
- Black Depth Property : for each node, all path from that node to descendants external nodes contains the same no of black tree.

Time Complexity of operations:

Insertion: O(log n)

Deletion: O(log n)

Searching: O(log n)

Results:

Insertion:

```
guest-9q9d30@SPIT:~/Desktop/Red-Black-Tree-master$ python rb.py
1.Insert node
2.Delete node
3.Print Tree
4.Search
5.Quit

Enter your choice1
insert ID cost and Qty seperated by space3,42,2
1.Insert node
2.Delete node
3.Print Tree
4.Search
5.Quit
```

```
Enter your choice1
insert ID cost and Qty seperated by space2,12,3
1.Insert node
2.Delete node
3.Print Tree
4.Search
5.Quit

Enter your choice1
insert ID cost and Qty seperated by space5,32,3
1.Insert node
2.Delete node
3.Print Tree
4.Search
5.Quit

Enter your choice1
insert ID cost and Qty seperated by space7,45,2
1.Insert node
2.Delete node
3.Print Tree
4.Search
5.Quit

Enter your choice1
insert ID cost and Qty seperated by space9,12,3
1.Insert node
2.Delete node
3.Print Tree
4.Search
5.Quit
```

Deletion:

```
Enter your choice3
Left/Right-----(id,Cost,Qty)-------color

R----['(3, 42, 2)'](BLACK)
     L----['(2, 12, 3)'](BLACK)
     R----['(7, 45, 2)'](BLACK)
          L----['(5, 32, 3)'](RED)
          R----['(9, 12, 3)'](RED)
1.Insert node
2.Delete node
3.Print Tree
4.Search
5.Quit

Enter your choice2
enter node to be deleted5,32,3
1.Insert node
2.Delete node
3.Print Tree
4.Search
5.Quit

Enter your choice3
Left/Right-----(id,Cost,Qty)-------color

R----['(3, 42, 2)'](BLACK)
     L----['(2, 12, 3)'](BLACK)
     R----['(7, 45, 2)'](BLACK)
          R----['(9, 12, 3)'](RED)
1.Insert node
2.Delete node
3.Print Tree
4.Search
5.Quit
```

Search:

```
Enter your choice4
enter the node to be search 7,45,2
found
1.Insert node
2.Delete node
3.Print Tree
4.Search
5.Quit
```

Conclusion:

Here we can conclude that RBT have time complexity of O(log n) for all primitive operation(insertion, deletion, search).Time complexity of RBT is directly propositional to number of nodes present in the tree.