Binary Tree

March 20 ,2020

1 Binary Tree Implementation

1.1 Tree Node Declaration

```
struct Node{
        int data;
        Node* left;
        Node* right ;
         Node(int x){
                data = x;
                left =NULL;
                right = NULL;
                }
};
```

1.2 Taking a Binary Tree as Input

The first line of the input denotes n the number of nodes in the tree. Each of the next n-1 lines are of the format : a b L or a b R

a b L : means node with index b is the left child of node with index a

a c R : means node with index c is the right child of node with index a

And the next n lines will contain the value node from 1 to n .

Assume that node 1 is always the root node.

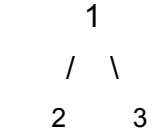
E.g. Input

3

1 2 L

1 3 R

101 86 99



Node with index 1 will have value 101

Node with index 2 will have value 86

Node with index 3 will have value 99

```
int n;
cin>>n;
Node* arr[n+1];  // For 1 based indexing
for(int i=0;i<n+1;i++){
        arr[i] = new Node(1000); //1000 is just a random number(initial initialization)
                }

for(int i=0;i<n-1;i++){
        int a,b;
        char c;
        cin>>a>>b>>c;
        if(c=='L')
        {
                arr[a]->left=arr[b];
        }
        else if(c=='R')
        {
                arr[a]->right=arr[b];
        }
}

for(int i=1;i<=n;i++){
        cin>>arr[i]->data;
}

Node* root=arr[1];
return root;
```

2 Tree Traversal

2.1 PostOrder Traversal

```
void PostOrder(struct Node* node)
{
    if (node == NULL)
        return;

    // first recur on left subtree
    PostOrder(node->left);

    // then recur on right subtree
```

```
    PostOrder(node->right);

    // now deal with the node
    cout << node->data << " ";
}
```

2.2 Inorder Traversal

```
void Inorder(struct Node* node)
{
    if (node == NULL)
        return;

    Inorder(node->left);

    cout << node->data << " ";

   Inorder(node->right);
}
```

2.3 PreOrder Traversal

```
void PreOrder(struct Node* node)
{
    if (node == NULL)
        return;

    /* first print data of node */
    cout << node->data << " ";

    /* then recur on left sutree */
    PreOrder(node->left);

    /* now recur on right subtree */
    PreOrder(node->right);
}
```

2.4 Level Order Traversal

```
void LevelOrder(node* root)
{

        queue<node*> q;
        q.push(root);
        while(q.size())
        {
                node* curr = q.front();
                q.pop();
                if(curr==NULL)
```

```
                continue;
            cout  << curr->data << endl;
            q.push(curr->left), q.push(curr->right);
        }
    }
```

Example:

```
        1

    /       \

3             5

        /          \

     7              -1
```

Output: 1 3 5 7 -1

Question

- Inorder Traversal

  www.hackerrank.com/challenges/tree-postorder-traversal/problem

- Preorder Traversal

  www.hackerrank.com/challenges/tree-postorder-traversal/problem

- Postorder Traversal

  www.hackerrank.com/challenges/tree-postorder-traversal/problem

Questions:

- Find nth Node in inorder traversal using only O(1) memory.

- Print the “size” of every node in the tree.

- Print the maximum and minimum element in the subtree for every node in the tree.
    - Print in INORDER way.

- Height of Binary Tree

  https://www.hackerrank.com/challenges/tree-height-of-a-binary-tree/problem

- Find diameter of a binary tree

- Construct Tree from given Inorder and Preorder traversals

  https://www.hackerrank.com/challenges/tree-height-of-a-binary-tree/problem

- Print cousins of given node in a binary tree
- In-place convert given binary tree to its sum tree
- Find the distance between given pairs of nodes in a binary tree
- Level order Traversal https://www.hackerrank.com/challenges/tree-level-order-traversal/problem
- Top View of a Tree https://www.hackerrank.com/challenges/tree-top-view/problem
- Construct a tree from Inorder and Level order traversals https://www.geeksforgeeks.org/construct-tree-inorder-level-order-traversals/
- Construct a complete binary tree from given array in level order fashion https://www.geeksforgeeks.org/construct-complete-binary-tree-given-array/
- Party https://www.geeksforgeeks.org/construct-complete-binary-tree-given-array/
- Mirror Image https://www.hackerearth.com/problem/algorithm/mirror-image-1/
- Given the root node a binary tree, find the maximal left spine of the tree. The left spine is the simple path from the root that consists of only the leftmost edges at each level