TREES

INORDER SUCCESSOR OF BINARYTREE

Exchange the Leaf Nodes
Sum of the Longest Bloodline of a Tree
Remove Half Nodes
Leaves to DLL
Check if Tree is Isomorphic
***Vertical sum (Special Algo)

http://www.geeksforgeeks.org/lowest-common-ancestor-in-a-binary-search-tree/
http://www.geeksforgeeks.org/diagonal-sum-binary-tree/

http://www.geeksforgeeks.org/diameter-of-a-binary-tree/
Serialize and Deserialize a Binary Tree

*****Diameter of tree

Views of tree:

- [ ] https://www.geeksforgeeks.org/print-right-view-binary-tree-2/
- [ ] https://www.geeksforgeeks.org/print-nodes-top-view-binary-tree/
- [ ] https://www.geeksforgeeks.org/bottom-view-binary-tree/
- [ ] https://www.geeksforgeeks.org/print-left-view-binary-tree/
- [ ] https://www.geeksforgeeks.org/print-binary-tree-vertical-order/
- [ ] Diagonal Traversal of tree

1. Maximum sum path
2. Root to leaf paths sum
3. http://www.geeksforgeeks.org/find-maximum-path-sum-two-leaves-binary-tree/

Enumeration of Binary Trees

A Binary Tree is labeled if every node is assigned a label and a Binary Tree is unlabeled if nodes are not assigned any label.

Below two are considered same unlabeled trees

```
   o            o
 /  \         /  \
o    o       o    o
```

Below two are considered different labeled trees

```
   A            C
 /  \         / \
B    C       A    B
```

How many different Unlabeled Binary Trees can be there with n nodes?

For n = 1, there is only one tree

```
  o
```

For n = 2, there are two trees

```
  o     o
 /       \
o         o
```

For n = 3, there are five trees

```
   o     o         o        o     o
  /       \       / \     /        \
 o         o     o   o   o          o
/           \              \       /
o            o              o     o
```

The idea is to consider all possible pair of counts for nodes in left and right subtrees and multiply the counts for a particular pair. Finally add results of all pairs.

For example, let T(n) be count for n nodes.

T(0) = 1 [There is only 1 empty tree]

T(1) = 1

T(2) = 2

T(3) = T(0)*T(2) + T(1)*T(1) + T(2)*T(0) = 1*2 + 1*1 + 2*1 = 5

T(4) = T(0)*T(3) + T(1)*T(2) + T(2)*T(1) + T(3)*T(0)
     = 1*5 + 1*2 + 2*1 + 5*1
     = 14

The above pattern basically represents n'th Catalan Numbers.

SERIES

First few catalan numbers are 1 1 2 5 14 42 132 429 1430 4862,...

$$T(n) = \sum_{i=1}^{n} T(i-1)T(n-i) = \sum_{i=0}^{n-1} T(i)T(n-i-1) = C_n$$

Here,
T(i-1) represents number of nodes on the left-sub-tree
T(n–i-1) represents number of nodes on the right-sub-tree
n’th Catalan Number can also be evaluated using direct formula.

T(n) = (2n)! / (n+1)!n!

```
class Solution {
public:
    int numTrees(int n) {
        int g[n+1];
        for(int i=0;i<n+1;i++){
            g[i]=0;
        }
        g[0]=1;
        g[1]=1;
        for(int i=2;i<n+1;i++){
            for(int j=0;j<i;j++){
                g[i]+=g[j]*g[i-j-1];
            }
        }
        return g[n];
    }
};
```