**112. Path Sum**

Easy

1948497Add to ListShare

Given a binary tree and a sum, determine if the tree has a root-to-leaf path such that adding up all the values along the path equals the given sum.

**Note:** A leaf is a node with no children.

**Example:**

Given the below binary tree and sum = 22,

**5**

**/** \

**4** 8

**/** / \

**11** 13 4

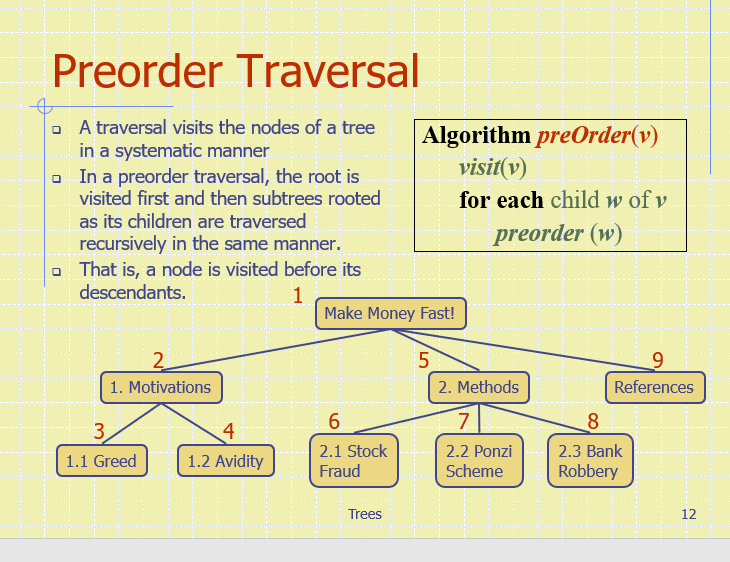
/ **\** \

7 **2** 1

那我们啥时候能知道是不是等于SUM

自然是到底的时候，

所以应该是自上到下



class Solution {

public boolean hasPathSum(TreeNode root, int sum) {

if(root==null)

return false;

if(root.val == sum && root.left==null && root.right==null) //VISIT自己这个点，也就是当他reach到底部的时候，是不是等于SUM

return true;

return hasPathSum(root.left , sum-root.val) || hasPathSum(root.right,sum-root.val);

如果不是底部，遍历所有子节点，sum=sum-root.val

}

}

Iterative方法

用一个Stack记录每个节点加后的值

class Solution {

public boolean hasPathSum(TreeNode root, int sum) {

if(root==null)

return false;

Stack<TreeNode> node\_stack=new Stack();

Stack<Integer> sum\_stack=new Stack();

node\_stack.push(root);

sum\_stack.push(root.val);

while(!node\_stack.isEmpty()){

TreeNode current\_node=node\_stack.pop();

int current\_sum=sum\_stack.pop();

if(current\_node.left==null&&current\_node.right==null&&current\_sum==sum)

return true;

if(current\_node.left!=null){

node\_stack.push(current\_node.left);

sum\_stack.push(current\_sum+current\_node.left.val);

}

if(current\_node.right!=null){

node\_stack.push(current\_node.right);

sum\_stack.push(current\_sum+current\_node.right.val);

}

}

return false;

}

}

