Given the root of a tree, you are asked to find the most frequent subtree sum. The subtree sum of a node is defined as the sum of all the node values formed by the subtree rooted at that node (including the node itself). So what is the most frequent subtree sum value? If there is a tie, return all the values with the highest frequency in any order.

Examples 1

Input:

```
    5
   /  \
  2   -3
```

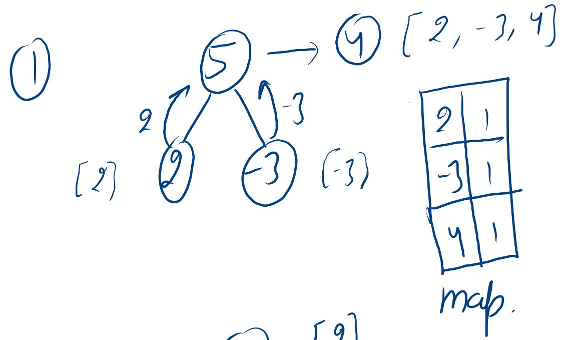
return [2, -3, 4], since all the values happen only once, return all of them in any order.

Examples 2

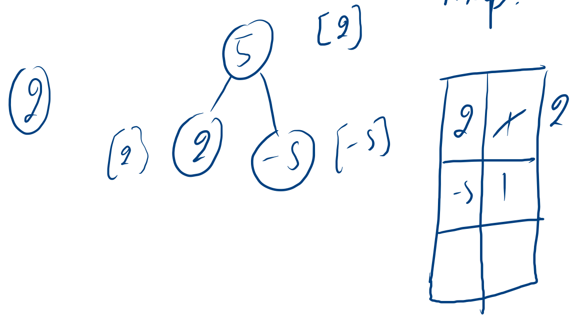
Input:

```
    5
   /  \
  2   -5
```

```
#define pb(val) push_back(val)

unordered_map<int,int> map;
int maxFreq=0;

vector<int> findFrequentTreeSum(TreeNode* node) {
    if(node==nullptr) return {};

    dfs(node);
    vector<int> ans;
    for(pair<int,int> key:map){                         // vo sarre element jiski freq, maxFreq ke
        if(key.second==maxFreq) ans.pb(key.first);       equal hai.

    }

    return ans;
}

int dfs(TreeNode* node) {
    if(node==nullptr) return 0;

    int sum=dfs(node->left) + dfs(node->right) + node->val;   // calculate sum.

    map[sum]++;                                         // 1. populate hashmap
    if(map[sum]>maxFreq) maxFreq=map[sum];              // 2. find maxFreq sum ele in hashmap

    return sum;
}
```

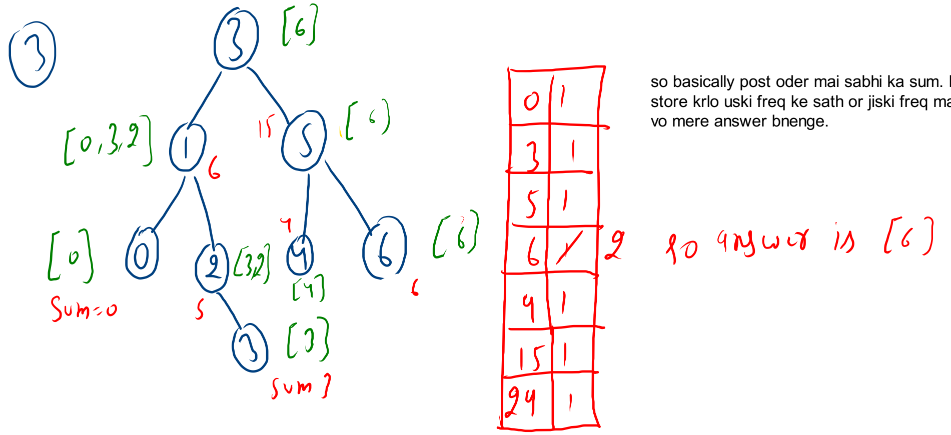


2	1
-3	1
4	1



2	~~1~~	2
-5	1	

here 2 have heighest frequency so answer vector contain 2.



0	1	
3	1	
5	1	
6	~~1~~	2
4	1	
15	1	
24	1	

so basically post oder mai sabhi ka sum. hashmap mai store krlo uski freq ke sath or jiski freq max and equal hogi vo mere answer bnenge.