

106. Construct Binary Tree from Inorder and Postorder Traversal

106 Construct Binary Tree from Inorder and Postorder Traversal

Description

Given inorder and postorder traversal of a tree, construct the binary tree.

Note:

You may assume that duplicates do not exist in the tree.

Solution

- Tree
- Depth-first Search
- Array

Depth-first Search

```
1 /**
2  * Definition for a binary tree node.
3  * struct TreeNode {
4  *     int val;
5  *     TreeNode *left;
6  *     TreeNode *right;
7  *     TreeNode(int x) : val(x), left(NULL), right(NULL) {}
8  * };
9  */
10 class Solution {
11
12 public:
13     TreeNode* buildTree(vector<int>& inorder, vector<int>& postorder){
14         return helper(inorder,0,inorder.size(),postorder,0,postorder.size());
15     }
16
17 private:
18     TreeNode* helper(vector<int>& inorder,int i,int j,vector<int>& postorder,int ii,int jj)
19     {
20         // 每次取postorder的最后一个值mid, 将其作为树的根节点
21         // 然后从inorder中找到mid, 将其分割为两部分, 左边作为mid的左子树, 右边
22         // 作为mid的右子树
23         // tree:      8 4 10 3 6 9 11
24         // Inorder   [3 4 6] 8 [9 10 11]
25         // postorder [3 6 4]  [9 11 10] 8
26
27         if(i >= j || ii >= jj)
28             return NULL;
29
30         int mid = postorder[jj - 1];
31
32         auto f = find(inorder.begin() + i,inorder.begin() + j,mid);
33
34         int dis = f - inorder.begin() - i;
35
36         TreeNode* root = new TreeNode(mid);
37         root->left = helper(inorder,i,i + dis,postorder,ii,ii + dis);
38         root->right = helper(inorder,i + dis + 1,j,
39                             postorder,ii + dis,jj - 1);
40
41         return root;
42     }
43 }
44 };
```

