**Construct Binary Tree from Inorder and Postorder Traversal**

TreeNode<int>\* constructTree(vector < int > & postorder, int postStart, int postEnd,

vector < int > & inorder, int inStart, int inEnd, map < int, int > & mp) {

  if (postStart > postEnd || inStart > inEnd) return NULL;

  TreeNode<int>\* root = new TreeNode<int>(postorder[postEnd]);

  int elem = mp[root -> data];

  int nElem = elem - inStart;

  root -> left = constructTree(postorder, postStart, postStart + nElem - 1,

  inorder, inStart, elem - 1, mp);

  root -> right = constructTree(postorder, postStart + nElem, postEnd-1, inorder,

  elem + 1, inEnd, mp);

  return root;

}

TreeNode<int>\* getTreeFromPostorderAndInorder(vector<int>& postOrder, vector<int>& inOrder)

{

     // Write your code here.

       int postStart = 0, postEnd = postOrder.size() - 1;

  int inStart = 0, inEnd = inOrder.size() - 1;

  map < int, int > mp;

  for (int i = inStart; i <= inEnd; i++) {

    mp[inOrder[i]] = i;

  }

  return constructTree(postOrder, postStart, postEnd, inOrder, inStart, inEnd, mp);

}