数据结构作业(第七次)

PB20111686 黄瑞轩

6.38

前序遍历时为：根→左子树→右子树；后序遍历时为：左子树→右子树→根，则对前序遍历做如下改造：

①先访问右子树再访问左子树，变为根→右子树→左子树

②将答案反过来，变为左子树→右子树→根

```
deque<int> PostorderTraverse(TreeNode* root){
    stk<TreeNode*> stk;
    deque<int> ans;
    if(root) stk.push(root);
    while(!stk.empty()){
        TreeNode* tmp = stk.top();
        ans.push_front(tmp->val);
        stk.pop();
        if(tmp->left) stk.push(tmp->left);
        //栈先进后出，这里先加入左子树再加入右子树，就相当于执行任务的时候先看右子树再看左子树
        if(tmp->right) stk.push(tmp->right);
    }
    return ans;
}
```

6.47

```
vector<int> FloorTraverse(TreeNode* root){
    queue<TreeNode*> Q;
    vector<int> ans;
    if(root) Q.push(root);
    while(!Q.empty()){
        TreeNode* tmp = Q.front();
        Q.pop();
        ans.push_back(tmp->val);
        if(tmp->left) Q.push(tmp->left);
        if(tmp->right) Q.push(tmp->right);
    }
    return ans;
}
```

6.58

题目说以`*x`为根、只有左子树，则默认`x`和不是`nullptr`，且`x->LTag == 0`，即`x`确实有不空的左子树。同理，如果说`*p`有左子树，说明`p->LTag == 0`。

```
void Insert(TreeNode* p, TreeNode* x){
    TreeNode* x_1 = x;
    while(x_1!=nullptr && x_1->LTag==0){    // 表示x_f的左指针指向的是左孩子
        x_1 = x_1->left;
    }
    if(p->LTag == 0){    // p原来有左子树，则要找到这个左子树的第一个元素
        TreeNode* p_1 = p;
        while(p_1!=nullptr && p_1->LTag==0){
            p_1 = p_1->left;
        }
        x_1->left = p_1->left;
        x->RTag = 0;
        x->right = p->left;
        p_1->left = x;
        p_1->LTag = 0;
    }
    else{    // p原来没有左子树
        x_1->left = p->left;
        x->RTag = 1;
        p->LTag = 0;
        x->right = p;
        p->left = x;
    }
}
```