

二叉树先序、中序、后序遍历的非递归实现

收藏人: 勤奋不止 [+关注](#) [与TA对话](#)

2012-05-10 | 阅: 1982 转: 32 | 来源 [A](#) [分享](#) [分享到微信](#) [转藏到我的图书馆](#)

二叉树先序、中序、后序遍历的非递归实现

Filed in: [C++学习](#), [数据结构与算法](#)在网上看了一些用非递归实现先序中序后序 [Add comments](#)
遍历二叉树的代码，都很混乱，while、if各种组合嵌套使用，逻辑十分不清晰，把我也搞懵了。想了大半天，写了大半天，突然开了窍，实际上二叉树的这三种遍历在逻辑上是十分清晰的，所以才可以用递归实现的那么简洁。既然逻辑如此清晰，那么用非递归实现也应该是清晰的。

自认为自己的代码比网上搜到的那些都清晰得多，好理解得多。

稍微解释一下：

先序遍历。将根节点入栈，考察当前节点（即栈顶节点），先访问当前节点，然后将其出栈（已经访问过，不再需要保留），然后先将其右孩子入栈，再将其左孩子入栈（这个顺序是为了让左孩子位于右孩子上面，以便左孩子的访问先于右孩子；当然如果某个孩子为空，就不用入栈了）。如果栈非空就重复上述过程直到栈空为止，结束算法。

中序遍历。将根节点入栈，考察当前节点（即栈顶节点），如果其左孩子未被访问过（有标记），则将其左孩子入栈，否则访问当前节点并将其出栈，再将其右孩子入栈。如果栈非空就重复上述过程直到栈空为止，结束算法。

后序遍历。将根节点入栈，考察当前节点（即栈顶节点），如果其左孩子未被访问过，则将其左孩子入栈，否则如果其右孩子未被访问过，则将其右孩子入栈，如果都已经访问过，则访问其自身，并将其出栈。如果栈非空就重复上述过程直到栈空为止，结束算法。

其实，这只不过是保证了先序中序后序三种遍历的定义。对于先序，保证任意一个节点先于其左右孩子被访问，还要保证其左孩子先于右孩子被访问。对于中序，保证任意一个节点，其左孩子先于它被访问，右孩子晚于它被访问。对于后序，保证任意一个节点的左孩子右孩子都先于它被访问，其中左孩子先于右孩子被访问。如是而已。

代码里应该体现得比较清楚。这里不光给出了非递归版本，也给出了递归版本。

```
#include <iostream>
#include <stack>

using namespace std;

struct TreeNode
{
    int data;
```

最新文章

- cocos2d-x中Lua类型强转问题
- 【Lua】为什么 Lua 里没有 continue
- Lua性能优化技巧
- lua中table如何安全移除元素
- 关于如何释放lua table 占用的内存
- Lua | Cocos2d

[更多](#)

热门文章

- 那些感动过你我的电影音乐
- 【强烈推荐】德国MM减肥法
- 中小学教师职称改革方案
- 50条一句话搞笑语录：泼出去的水、我...
- 中国模特张莫霏
- 《古玩指南全编》
- 小学奥数解题技巧大全100讲
- 中纪委推荐新年第一书：《历史的教
- 《宽心谣》一诗，越看越宽心
- 脑梗塞及脑血管的保健
- 【清一色宽800 性感迷人美女大图30幅...
- 广场舞：动人心扉的声音30首

我要转藏

官方微信

[更多>>](#)

Ad7.

新技术

快速找回流失客户

点击了解

```
TreeNode* left;
TreeNode* right;
int flag;

};

typedef TreeNode *TreePtr;

TreePtr CreateTree()
{
    TreePtr root = new TreeNode;
    cout<<"input the data :\n";
    int n;
    cin>>n;
    if (n == -1)
    {
        return NULL;
    }
    else
    {
        root->data = n;
        root->flag = 0;
        root->left = CreateTree();
        root->right = CreateTree();
    }
    return root;
}

void PreOrderRecursion(TreePtr p)
{
    if (p == NULL)
    {
        return;
    }
    cout<<p->data<<" ";
    PreOrderRecursion(p->left);
    PreOrderRecursion(p->right);
}

void InOrderRecursion(TreePtr p)
{
    if (p == NULL)
    {
        return;
    }
    InOrderRecursion(p->left);
    cout<<p->data<<" ";
    InOrderRecursion(p->right);
}

void PostOrderRecursion(TreePtr p)
{
    if (p == NULL)
    {
```

大学生兼职网	42寸液晶电视	厨房电器
车险报价		加盟奶茶店
		台湾旅游
肾病食谱	金属探测仪价格	上班族兼职
燃气灶十大品牌	投资创业	女装马甲
房间装修	农村致富好项目	
吸粪车价格		烤鱼培训
		净水器滤芯
洗衣机十大品牌	苹果5	

关闭

关闭

```

        return;
    }
    PostOrderRecursion(p->left);
    PostOrderRecursion(p->right);
    cout<<p->data<<" ";
}

void PreOrderNoRecursion(TreePtr p)
{
    cout<<"PreOrderNoRecursion\n";

    stack<TreeNode> stk;
    TreeNode t = *p;
    stk.push(t);

    while (!stk.empty())
    {
        t = stk.top();
        stk.pop();
        cout<<t.data<<" ";

        if (t.right != NULL)
        {
            stk.push(*t.right);
        }

        if (t.left != NULL)
        {
            stk.push(*t.left);
        }
    }
}

void InOrderNoRecursion(TreePtr p)
{
    cout<<"InOrderNoRecursion\n";

    stack<TreeNode> stk;
    TreeNode t = *p;
    stk.push(t);

    while (!stk.empty())
    {
        if (stk.top().flag == 0)
        {
            stk.top().flag++;
            if (stk.top().left != NULL)
            {
                stk.push(*stk.top().left);
            }
        }
        else
        {
            t = stk.top();

```

```

        stk.pop();
        cout<<t.data<<" ";
        if (t.right != NULL)
        {
            stk.push(*(t.right));
        }
    }
}

void PostOrderNoRecursion(TreePtr p)
{
    cout<<"PostOrderNoRecursion\n";

    stack<TreeNode> stk;
    TreeNode t = *p;
    stk.push(t);

    while (!stk.empty())
    {
        if (stk.top().flag == 0)
        {
            stk.top().flag++;
            if (stk.top().left != NULL)
            {
                stk.push(*(stk.top().left));
            }
        }
        else if (stk.top().flag == 1)
        {
            stk.top().flag++;
            if (stk.top().right != NULL)
            {
                stk.push(*(stk.top().right));
            }
        }
        else
        {
            cout<<stk.top().data<<" ";
            stk.pop();
        }
    }
}

int main()
{
    TreePtr t = CreateTree();

    cout<<"PreOrderRecursion\n";
    PreOrderRecursion(t);
    cout<<"\n";

    cout<<"InOrderRecursion\n";

```

```
InOrderRecursion(t);
cout<<"\n";

cout<<"PostOrderRecursion\n";
PostOrderRecursion(t);
cout<<"\n";

PreOrderNoRecursion(t);
cout<<"\n";

InOrderNoRecursion(t);
cout<<"\n";

PostOrderNoRecursion(t);
cout<<"\n";

}
```

来自： [勤奋不止](#) > 《数据结构和算法》 [+关注](#) [与TA对话](#)

上一篇： [静态库和动态库的总结](#)
下一篇： [连通图的割点、割边\(桥\)、块、缩点，有向图的强连通分量](#)

转藏到我的图书馆

献花(0)

分享到微信

以文找文

分享：

类似文章

- [二叉树的非递归遍历](#)
- [算法1二叉树的遍历方法](#)
- [向量及单链表实现栈的界面](#)
- [ucos原理分析](#)
- [uc/os—II下的九个C语言文件功能函数大全....](#)
- [二叉树三种遍历的非递归算法](#)
- [二叉树 非递归遍历 栈实现（前、中后序）](#)
- [面试题](#)

更多

热搜文章

- [绝代双骄大放异彩:一张图看懂我国的重要...](#)
- [古德先贤故事五则 教我们做好人好事的...](#)
- [回忆曾经的美好 老照片：那些年我们这样...](#)
- [安妮宝贝的美丽语录，年轻人必看的情感哲理](#)
- [两千年来的政治隐语](#)
- [190座城市雾霾排名，你的城市第几？](#)
- [让人感动——一家人的爱在善意的谎言里延...](#)
- [史上最牛数学动态图,让你陷入数学的魅力...](#)

猜你喜欢

激励员工名言100句

成功人生的七大靠山

惊! 北京一布局让世界倒退30年

你的时间在那里,成就就在那里

如何克服焦虑心情

销量：115件

销量：27件

销量：1128件

¥ 256.00

¥ 1199.00

¥ 236.00



发表评论：

您好，请 [登录](#) 或者 [注册](#) 后再进行评论

其它帐号登录：  