

二叉树前序后序中序遍历

https://blog.csdn.net/m0_50617544/article/details/128749340

九是否随机的称呼 于 2023-05-12 11:30:00 发布

```
1  #include<cstdio>
2  #include<iostream>
3  #include<vector>
4  #include<string>
5  using namespace std;
6  struct nod {
7      int val = -9;
8      nod *l=NULL, *r=NULL;
9  };
10 int pre[] = {1, 2, 3, 4, 5, 6};
11 int in[] = {3, 2, 4, 1, 6, 5};
12
13 //不能产生树的情况
14 // int pre[] = {1, 2, 3, 4, 6, 7, 5};
15 // int in[] = {2, 3, 1, 7, 4, 5, 6};
16 void printvec(vector<int> a, string t){
17     cout<<t<<"\t";
18     for(int i=0; i<a.size();i++){
19         cout<<a[i];
20         if(i!=(a.size()-1))
21             cout<<" ";
22     }
23     cout<<endl;
24 }
25 /*
26     1
27   2   5
28 3   4   6
29 */
30
31 vector<int> postarr;
```

内容来源: csdn.net

作者昵称: 九是否随机的称呼

原文链接: https://blog.csdn.net/m0_50617544/article/details/128749340

作者主页: https://blog.csdn.net/m0_50617544

```

32 void poster(int root, int start, int end) {
33     if(start > end) return;
34     int k = start;
35     while(k <=end && pre[root]!=in[k]) k++;
36     if(k > end) {
37         printf("//不能产生树的\n"); //来判断是否可以产生树
38         return;
39     }
40     poster(root+1, start, k-1);
41     poster(root + k - start + 1, k+1, end);
42     postarr.push_back(pre[root]);
43 }
44 nod* gentree(nod *root, int preroot, int start, int end) {
45     if(start > end) return NULL;
46     if(root==NULL) {
47         root = new(nod);
48         root->val = pre[preroot];
49     }
50     int k = start;
51     while(k <=end && pre[preroot]!=in[k]) k++;
52     if(k > end) {
53         printf("//不能产生树的\n"); //来判断是否可以产生树
54         return NULL;
55     }
56     root->l = gentree(root->l, preroot+1, start, k-1);
57     root->r = gentree(root->r, preroot + k - start + 1, k+1, end);
58     return root;
59 }
60 void preorder(nod *root) {
61     if(root==NULL) return;
62     postarr.push_back(root->val);
63     preorder(root->l);
64     preorder(root->r);
65 }
66 void inorder(nod *root) {
67     if(root==NULL) return;
68     inorder(root->l);
69     postarr.push_back(root->val);
70     inorder(root->r);

```

内容来源: csdn.net

作者昵称: 九是否随机的称呼

原文链接: https://blog.csdn.net/m0_50617544/article/details/128749340

作者主页: https://blog.csdn.net/m0_50617544

```

71 }
72 void postorder(nod *root) {
73     if(root==NULL) return;
74     postorder(root->l);
75     postorder(root->r);
76     postarr.push_back(root->val);
77 }
78 int main() {
79     poster(0, 0, 5); //0pre 0inl 5inr
80     printvec(postarr, "generate postarr:");
81
82     nod *root = NULL;
83     root = gentree(root, 0, 0, 5);
84     postarr.clear();
85     preorder(root);
86     printvec(postarr, "preorder:");
87
88     postarr.clear();
89     inorder(root);
90     printvec(postarr, "inorder:");
91
92     postarr.clear();
93     postorder(root);
94     printvec(postarr, "postorder:");
95     return 0;
96 }

```

PAT/binarytree at master · ZouJiu1/PAT (github.com)

<https://github.com/ZouJiu1/PAT/tree/master/binarytree>

内容来源: csdn.net

作者昵称: 九是否随机的称呼

原文链接: https://blog.csdn.net/m0_50617544/article/details/128749340

作者主页: https://blog.csdn.net/m0_50617544