2020 年 8 月 14 日

1.P1030 求先序排列

算法思路：

后序遍历的最后一个字母是树根，在中序遍历中找到这个字母，左边的字符串是左子树，右边的字符串是右子树，递归~

代码：

```
#include <iostream>
#include <string>
#include <algorithm>
using namespace std;

void digui(string s1,string s2){
    if(s1.length() == 0)    return;
    int index = s1.find(s2[s2.length()-1]);
    cout << s1[index];
    digui(s1.substr(0,index),s2.substr(0,index));
    digui(s1.substr(index+1),s2.substr(index,s2.length()-1-index));
}

int main(){
    string s1,s2;
    cin >> s1 >> s2;
    digui(s1,s2);
}
```

Accepted截图：

备注：

1. substr：

```
    //substr有2种用法：
    //假设：string s = "0123456789";

    string sub1 = s.substr(5); //只有一个数字5表示从下标为5开始一直到结尾：sub1 = "56789"

    string sub2 = s.substr(5, 3); //从下标为5开始截取长度为3位：sub2 = "567"
```

2.P4913 【深基16.例3】二叉树深度

算法思路：

按照左子树，右子树递归，每一层有1个高度，结果取max（左子树的高度，右子树的高度）

代码：

```
#include <iostream>

using namespace std;

struct tree
{
    int left,right;
}TREE[1000010];

int dfs(int node){
    if(TREE[node].left == 0 && TREE[node].right == 0)   return 1;

    int lh = 1, rh = 1;
    if(TREE[node].left != 0)
        lh += dfs(TREE[node].left);
    if(TREE[node].right != 0)
        rh += dfs(TREE[node].right);

    return lh >= rh ? lh : rh;
}

int main(){
    int n;
    cin >> n;
    for(int i = 1; i <= n; i++){
        cin >> TREE[i].left >> TREE[i].right;
    }
    int height = dfs(1);
    cout << height;
```

```
    return 0;
}
```

Accepted截图：