树与二叉树题解

找树根与孩子

```cpp
#include<bits/stdc++.h>
using namespace std;
int n,m,tree[101]={0};
int main()
{
    int i,x,y,root,maxroot,sum=0,j,Max=0;
    cin>>n>>m;//节点和边的数目
    for(i=1;i<=m;i++)
    {
        cin>>x>>y;
        tree[y]=x;//y是x的孩子
    }
    for(i=1;i<=n;i++)//找出树根
    {
        if(tree[i]==0)// i的父亲节点为0，即没有父亲节点
        {
            root=i;
            break;
        }
    }
    for(i=1;i<=n;i++)//找孩子最多的节点maxroot
    {
        sum=0;
        for(j=1;j<=n;j++)
        {
            if(tree[j]==i) sum++;
        }
        if(sum>Max)
        {
            Max=sum;
            maxroot=i;
        }
    }
    cout<<root<<endl<<maxroot<<endl;
    for(i=1;i<=n;i++)//maxroot的孩子
    {
        if(tree[i]==maxroot) cout<<i<<" ";
    }
    return 0;
}
```

小球

题意：模拟小球在二叉树上的移动

```cpp
#include<bits/stdc++.h>
using namespace std;
int d,m,i,flag;
```

```
/*每个小球都会落到结点上，只能往左或者往右，我们分析小球的奇偶性，
  发现：若小球编号是奇数，则是往左落下的第（I+1）/2个小球；
        若小球是偶数，则是往右落下的第I / 2个小球。
以此判断来模拟最后一个小球的路线。*/
int main()
{
    scanf("%d%d",&d,&m);
    flag=1;
    i=1;
    while (i<d)
    {
        if (m%2==1)
        {
            i++;
            flag=flag*2;
            m=(m+1)/2;
        }
        else
        {
            i++;
            flag=flag*2+1;
            m=(m+1)/2;
        }
    }
    printf("%d",flag);
}
```

二叉树遍历

建议手动模拟一下，如何通过中序和层次遍历建树，总结规律后，再看一下代码。

版本一：

```
#include<bits/stdc++.h>
using namespace std;
string s1,s2;
void calc(int l1,int r1,int l2,int r2)
{
    int i,j;
    for (i=l2;i<=r2;i++) //先找根
    {
        int b=0;
        for (j=l1;j<=r1;j++)
            if (s2[i]==s1[j])
            {
                cout<<s1[j];
                b=1;
                break;
            }
        if (b==1) break;
    }
    if (j>l1) calc(l1,j-1,1,r2); //递归左子树
    if (j<r1) calc(j+1,r1,1,r2); //递归右子树
}
int main()
{
    cin>>s1>>s2;
```

```
    calc(0,s1.length()-1,0,s2.length()-1);
    return 0;
}

```

版本二：

```
#include<bits/stdc++.h>
using namespace std;
string zx,ac;
void build(int l,int r){
//找出中序遍历区间[l,r]在按层遍历的根位置m，左子树区间[l,m-1]右子树区间[m+1,r]
for(int i=0;i<ac.length();i++){
    int m=zx.find(ac[i]);
    if(m>=l&&m<=r){
    cout<<ac[i];
    if(m>l)build(l,m-1);
    if(m<r)build(m+1,r);
    break;//已找到根，后面不再尝试
}
}
}
int main(){
    cin>>zx>>ac;
    build(0,zx.length()-1);
    return 0;
}

```

二叉树输出

先根据先序遍历和中序遍历，建立一个二叉树，然后用递归对二叉树的每一个结点进行统计，最后输出每个结点的那个值即可

```
#include<bits/stdc++.h>
using namespace std;
string xx,zx;
int a[200],n;
int fa(int l,int r){//找出中序遍历子树区间[l,r]中根节点位置并求出其长度
    int t,m;
    for(int i=0;i<n;i++){
        t=zx.find(xx[i]);
        if(t>=l&&t<=r){m=i;break;}//m存子树根在中序遍历的位置
    }
    if(t>l) a[m]+=fa(l,t-1);//加上左子树根节点长度
    if(t<r) a[m]+=fa(t+1,r);//加上右子树根节点长度
    if(l==r)a[m]=1;//若为叶子节点长度为1
    return a[m];
}
int main(){
    cin>>xx>>zx;
    n=xx.length();
    fa(0,n-1);
    for(int i=0;i<n;i++){
```

```
      for(int j=0;j<a[i];j++)
        cout<<xx[i];
    cout<<endl;
    }
    return 0;
}

```

扩展二叉树

真·建树，然后输出即可

```
#include<bits/stdc++.h>
using namespace std;
struct node{//二叉树孩子表示法
char data;int lchild,rchild;
}n[10005];
int root,i=-1,cnt;
string s;
void build(int &r){//参数引用传递(相当于r的值会被buid修改)
    if(s[++i]!='.'){
    n[++cnt]=(node){s[i],0,0};
    r=cnt;
    build(n[r].lchild);
    build(n[r].rchild);
}
}
void printzx(int r){//中序遍历
    if(r){
        printzx(n[r].lchild);
        cout<<n[r].data;
        printzx(n[r].rchild);
    }
}
void printhx(int r){//后序遍历
    if(r){
        printhx(n[r].lchild);
        printhx(n[r].rchild);
        cout<<n[r].data;
}
}
int main(){
    cin>>s;
    build(root);
    printzx(root);
    cout<<endl;
    printhx(root);
    return 0;
}
```