求子串 查找 调用自身

如果是空串，返回空子树

abcdefg

cbdaegf

```
BiTNode *createtree(string s1,string s2){
    if(!s1.size()) return nullptr;
    BiTNode * root =new BiTNode(s1[0]);
    int index=s2.find(s1[0]); // a--3 b--1
    // 根左边是左子树
    leftpre =s1.substr(1,index); // bcd c
    leftin = s2.substr(0,index); // cbd c
    root->lchild=createtree(leftpre,leftin); // b变为根
    rightpre=s1.substr(index+1);
    rightin=s2.substr(index+1);
    root->rchild=createtree(rightpre,rightin);
    return root;
}
```

先序和后序不行

中序和后序可以

后序最后一个是根，中序遍历取

搞懂。

A

左DBEH

取四个 BDEH

leftpre=

线索二叉树——简单知道，就是指针

Huffman编码

创建加上权值

权值的顺序和先序一致

```
if(!T->lchild && !T->rchild) {
    cin>>T->w;
}
```

构建最小二叉树

从下向上构建，优先构建权值最小的树

规定左子树权值小于右子树

笔试是计算题，不考代码

```
select (HT,i,&s1,&s2){
for(int k=0;k<i;k++){
    if(min1>HT[k].w &&HT[k].parent==-1){
        min2=min1;
        s2=s1;
        min1=HT[k].w;
        s1=k;
    }
    else if ()
}
}
```

```
string decode (string s,int n,Htree HT){
    int k=2*n-2;
    string res;
    for(auto x:s){
        if (x=='0') k=HT[k].lchild;
        else k=HT[k].rchild;
        if(叶子结点) {res+=HT[k].data;
                  k=2*n-2;
                 }
    }
}
```

交换左右子树

调用swap函数，对指针也适用

计算每个节点的左子树高度+右子树高度——最大距离

最大距离节点——层次遍历HK

之前内容的应用

后序遍历的非递归实现