7-2　是否完全二叉搜索树　(30分)

将一系列给定数字顺序插入一个初始为空的二叉搜索树（定义为左子树键值大，右子树键值小），你需要判断最后的树是否一棵完全二叉树，并且给出其层序遍历的结果。

输入格式：

输入第一行给出一个不超过20的正整数N；第二行给出N个互不相同的正整数，其间以空格分隔。

输出格式：

将输入的N个正整数顺序插入一个初始为空的二叉搜索树。在第一行中输出结果树的层序遍历结果，数字间以1个空格分隔，行的首尾不得有多余空格。第二行输出YES，如果该树是完全二叉树；否则输出NO。

输入样例1：

```
9
38 45 42 24 58 30 67 12 51
```

输出样例1：

```
38 45 24 58 42 30 12 67 51
YES
```

输入样例2：

```
8
38 24 12 45 58 67 42 51
```

输出样例2：

```
38 45 24 58 42 12 67 51
NO
```

```
#include<stdio.h>


typedef struct Node{
    int data;
    struct Node *left,*right;
}Node,*Tree;


Tree insert(Tree tree,int data);

Node* makenode(int data){
    Node* node;
    node=(Node*)malloc(sizeof(Node));
    node->left=node->right=NULL;
    node->data=data;
    return node;

}
```

```
Tree maketree(){
 int num;
 scanf("%d",&num);
 Tree tree;
 int data;
 if(num>0){

 scanf("%d",&data);
 tree=makenode(data);
 }else{
     return NULL;
 }
 for(int i=1;i<num;i++){
          scanf("%d",&data);
          tree=insert(tree,data);
 }
 return tree;

}

Tree insert(Tree tree,int data){
     if(!tree){
          tree=makenode(data);
     }else if(data<tree->data){
          tree->right=insert(tree->right,data);
     }else{
          tree->left=insert(tree->left,data);
     }
     return tree;
}


int middle(Tree tree){
     int count;
     Node* queue[20];
     int top,base;
     top=base=0;
     Node* node;
     queue[top++]=tree;
     int yezi=0;
     int wan=1;
     while(top!=base){
          node=queue[base++];
          if(yezi==1){
               if(node->left||node->right){
                    wan=0;
```

```
            }
        }
        if(!node->left&&node->right){
            wan=0;
        }
        if(node->left&&!node->right){
            yezi=1;
        }
        if(!node->left&&!node->right){
            yezi=1;
        }
        if(top==1){
        printf("%d",node->data);
        }else{
        printf(" %d",node->data);
        }

         if(node->left){
            queue[top++]=node->left;
        }

        if(node->right){
            queue[top++]=node->right;
        }

    }
    return wan;

}

int main(){
    Tree tree;
    tree=maketree();
    if(tree!=NULL){

    int flag=middle(tree);
    if(flag==0){
        printf("\nNO");
    }else{
        printf("\nYES");
    }
    }else{
        printf("NO");
    }


}
```