二叉树算法设计练习

- 设计算法求二叉树的结点个数
- 设计算法按前序次序打印二叉树中的叶子结点
- 设计算法求二叉树的深度

二叉树算法设计练习

设计算法求二叉树的结点个数

```
void Count(BiNode *root)
{
    if (root == NULL) return;
    else {
        Count(root->lchild);
        count++;      //count为全局量并已初始化为0
        Count(root->rchild);
    }
}
```

二叉树算法设计练习

设计算法按前序次序打印二叉树中的叶子结点

```
void PreOrder(BiNode *root)
{
    if (root == NULL) return;
    else {
        if (!root->lchild && !root->rchild)
              cout<<root->data;
        PreOrder(root->lchild);
        PreOrder(root->rchild);
    }
}
```

二叉树算法设计练习

设计算法求二叉树的深度

```
int  Depth(BiNode *root)
{
    if (root == NULL) return 0;
    else {
          //hl, hr为全局量并已初始化为0
         hl= Depth(root->lchild);
         hr= Depth(root ->rchild);
         return max(hl, hr)+1;
    }
}
```

二叉树算法设计练习 – 进阶

- 判断两个二叉树是否相同
- 交换二叉树所有左右子树
- 二叉树中查找数据元素x
- 二叉树中查找数据元素x的双亲
- 二叉树中删除数据元素x为根的子树

二叉树算法设计练习 – 进阶

判断两个二叉树是否相同

```
int judgebitree(bitree *bt1,bitree *bt2)
{ if (bt1==0 && bt2==0) // bt1==0 与  bt1==Null等价
        return(1);
  else if (bt1==0 || bt2==0 || bt1->data!=bt2->data)
        return(0);
  else{ temp1=judgebitree(bt1->lchild,bt2- >lchild);
        temp2= judgebitree(bt1->rchild,bt2->rchild);
        return(temp1*temp2);
      }
}
```

二叉树算法设计练习 – 进阶

交换二叉树所有左右子树

```
void SwapsSubTree(BinaryTree *root)
{ //后序遍历思想
        if (root==NULL) return;
        SwapsSubTree(root ->left);
        SwapsSubTree(root ->right);

       BinaryTree temp = root ->left;
        root ->left = root ->right;
        root ->right = temp;
}
```

二叉树算法设计练习 – 进阶

二叉树中查找数据元素x

```
void Search（ Tnode * root，elemtype x)
{      //int flag=0 ;TNode * myNode=null; 为全局量
       //前序遍历思想
       if (flag>0) return
       if (root==Null) return
if (root->data == x)
     {   myNode=p; flag=1;   };
else {
    Search(root->leftchild, x);
    Search(root->rightchild, x);
      }
}
```

二叉树算法设计练习 – 进阶

二叉树中查找数据元素x的双亲

```
void Parent  ( Tnode * root,  elemtype x)
{      //int flag=0 ;TNode * myNode=null; 为全局量
       if (flag>0) return
       if (root==Null) return
       if (root->leftchild!=Null)
              if (root->leftchild->data==x)
                     {   myNode=root; flag=1;   };
       if (root->rightchild!=Null)
              if (root->rightchild->data==x)
                     {   myNode=root; flag=1;   };
if (flag==0)
     {      Parent(root->leftchild, x);
            Parent(root->rightchild, x);
      }
}
```

二叉树算法设计练习－进阶

二叉树中删除数据元素x为根的子树

```
int DelTree（ Tnode * root，elemtype x)
{    //int flag=0 ;TNode * myNode=null; 为全局量
     Parent（ root，  x）；        //查找数据元素x的双亲
     if flag==0  return 0;
      if  （myNode->leftchild->data==x)
           {  p= myNode->leftchild; myNode->leftchild=null;}
       else
           {  p= myNode->rightchild; myNode->rightchild=null;}
     Release(p)；  //见教材  后序递归释放以p 为根节点子树
     return 1;
}
```