班级_____ _____ _____ _____

实验题目： 树和二叉树的关键算法实现

一、 实验目的

能够运用高级程序设计技术实现树和二叉树及其关键算法

二、实验原理

1.树和二叉树的基本原理

```
typedef struct node
{
    int data;
    struct node *Lson,*Rson;
}Bnode,*Bptr;
```

2.关键算法设计原理及性能影响因素分析

树的遍历主要有两种，一个是深度优先遍历，一个是广度优先遍历。深度优先遍历又有三种：前序、中序、后序遍历。

三、实验方案设计

1.存储方案设计

即用一个链表来存储一棵二叉树，二叉树中每一个结点用链表的一个链结点来存储。

包含数据域 data，左指针域 lchild，右指针域 rchild。

```
typedef struct  BiTNode
{
    ElemType data;//数据域
    struct BiTNode *Ichild, *rchild;//左、右孩子指针
}BiTNode, *BiTree
```

2.算法的设计和实现

（1）、先序遍历

```c
void PreOrder(BiTtree T){
    if(T!=NULL){
        visit(T);//访问根结点
        PreOrder(T->Ichild);//递归遍历左子树
        PreOrder(T->rchild);//递归遍历右子树
    }
}
```

（2）、中序遍历

```c
void InOrder(BiTtree T){
    if(T!=NULL)
    {
        InOrder(T->Ichild//递归遍历左子树
        visit(T);//访问根结点
        InOrder(T->rchild);//递归遍历右子树
    }
}
```

（3）、后序遍历

```c
void PostOrder(BiTtree T)
{
    if(T!=NULL)
    {
        PostOrder(T->Ichild);//访问根结点
        PostOrder(T->rchild);//递归遍历左子树
        visit(T);//递归遍历右子树
    }
}
```

（4）、二叉树的遍历

```c
void InOrder(BiTtree T)
{
  //二叉树中序遍历的非递归算法，算法需要借助一个栈
  InitStack (S) ;//初始化栈
  Bitree p=T;//p是遍历指针
  while(p||!IsEmpty(S))//栈不空或p不空时循环
  {
  if(p)//根指针进栈，遍历左子树
  {
    Push(S,p);//每遇到非空二叉树先向左走
    p=p->lchild;
  }
    else
    {
    Pop(S,p);//根指针退栈，访问根结点，遍历右子树
    visit(p);//退栈，访问根结点
    p=p->rchild;//再向右子树走
    }
  }
}
```

3、测试用例

输入用例：无

期待输出：二叉树的遍历结果