

**数据结构实验报告**

**学 院** 信息工程学院

**专 业** 计算机类（大类招生）

**班 级** 2016240210

**姓 名** 瞿强鑫

**学 号** 2016902094

**指导教师** 史昕

**完成时间** 2017.6.15

1. **实验目的**

1.掌握二叉树的逻辑结构；

2.掌握二叉链表存储结构；

3.验证二叉树的二叉链表存储和遍历操作；

**二、实验设备（环境）及要求**

电脑一台，VC++环境下运行。

1. **实验内容（步骤）与方法**

内容：1.建立一棵含有n个结点的树，采用二叉链表结构

2.输出前序、中序、后序、层序遍历结果

方法：

在本次实验中，采用C++的模板机制，建立结点结构体，并声明了二叉树类， 通过调用前序遍历、中序遍历、后序遍历三个递归算法达到对二叉树的遍历操作。 通过队列的一系列特性实现了对二叉树的层序遍历。

//结点结构体

**struct BiNode**

{

DataType data;

BiNode<DataType> \* lchild,\* rchild;

};

//定义二叉树类

**template<class DataType>**

**class BiTree**

{

public:

BiTree(){root=Creat(root);}

~BiTree(){Release(root);}

void PreOrder(){PreOrder(root);}

void InOrder(){InOrder(root);}

void PostOrder(){PostOrder(root);}

void LeverOrder();

private:

BiNode<DataType> \* root;

BiNode<DataType> \* Creat(BiNode<DataType> \* bt);

void Release(BiNode<DataType> \* bt);

void PreOrder(BiNode<DataType> \* bt);

void InOrder(BiNode<DataType> \* bt);

void PostOrder(BiNode<DataType> \* bt);

};

//建立二叉链表

**template<class DataType>**

**BiNode<DataType> \*BiTree<DataType>::Creat(BiNode<DataType> \* bt)**

{

char ch;

cout<<"请输入结点信息(输入#结束)：";

cin>>ch;

if(ch=='#') bt=NULL;

else{

bt=new BiNode<DataType>;bt->data=ch;

bt->lchild=Creat(bt->lchild);

bt->rchild=Creat(bt->rchild);

}

return bt;

}

//释放二叉链表

**template<class DataType>**

**void BiTree<DataType>::Release(BiNode<DataType> \* bt)**

{

if(bt!=NULL)

{

Release(bt->lchild);

Release(bt->rchild);

delete bt;

}

}

//前序遍历

**template<class DataType>**

**void BiTree<DataType>::PreOrder(BiNode<DataType> \* bt)**

{

if(bt==NULL) return;

else {

cout<<bt->data<<" ";

PreOrder(bt->lchild);

PreOrder(bt->rchild);

}

}

//中序遍历

**template<class DataType>**

**void BiTree<DataType>::InOrder(BiNode<DataType> \* bt)**

{

if(bt==NULL) return;

else {

InOrder(bt->lchild);

cout<<bt->data<<" ";

InOrder(bt->rchild);

}

}

//后序遍历

**template<class DataType>**

**void BiTree<DataType>::PostOrder(BiNode<DataType> \* bt)**

{

if(bt==NULL) return;

else {

PostOrder(bt->lchild);

PostOrder(bt->rchild);

cout<<bt->data<<" ";

}

}

//层序遍历

**template<class DataType>**

**void BiTree<DataType>::LeverOrder()**

{

BiNode<DataType> \* Q[20];

int front,rear;

front=rear=-1;

if(root==NULL) return;

Q[++rear]=root;

BiNode<DataType> \* q;

while(front!=rear)

{

q=Q[++front];

cout<<q->data;

if(q->lchild!=NULL) Q[++rear]=q->lchild;

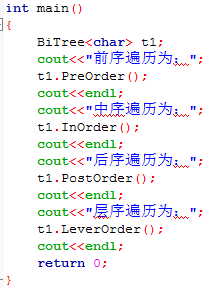
if(q->rchild!=NULL) Q[++rear]=q->rchild;

}

}

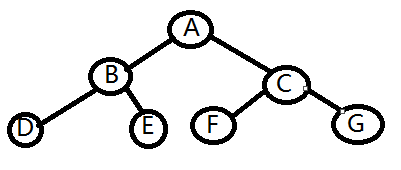
1. **实验结果与分析**

如图1，是main函数详细内容：



**图1**

如图二，建立如图2的二叉树：



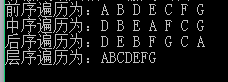
**图2**

按照如图三所示，输入二叉树各元素，建立成功！



**图3**

前序、中序、后序、层序遍历结果如图4，四种遍历结果都达到预期目标。



**图4**

**五、实验心得与体会**

这次的实验比较顺利，发生的错误较少，已经很少出现基础错误了。期 间发现了还是有技术性的错误：对已经声明的指针申请空间的时候，我忘了 给指针后面附加模板类名造成了错误：bt=new BiNode<DataType>；在声明 新的数组时，我再次犯了同样的错误：BiNode<DataType> \* Q[20]。此外， 我发现自己对类的使用还不够熟练，不能迅速翻译出对应编码，而少了一些 类的关键码。

其次，我发现了在输入二叉树各结点数据信息时，对过程的把握需要深 刻认识，掌握前序遍历从而才能实现对二叉树信息的输入和输入结束#的控 制。对每个叶子结点都需要两个#放置在子树表示结束，这操作起来很是麻 烦。

最后，这次的实验规模较小，实现起来较为容易，希望我能够掌握扎实 二叉树的知识，学好、用好。