**《数据结构》**

**第四次实验报告**

专 业： 空间信息与数字技术

班 级： 18级空间信息与数字技术

学号姓名： 季文哲

指导教师： 袁莹

成 绩：

**计算机与信息工程学院**

**2019 年 11月 29 日**

**一、实验目的：**

1.掌握二叉树的逻辑结构；

2.掌握二叉树的二叉链表存储结构；

3.验证二叉树的二叉链表存储及遍历操作；

4.掌握树的逻辑结构；

5.掌握树的孩子兄弟存储结构；

6.验证树的孩子兄弟存储结构及遍历操作。

**二、实验内容：**

1.建立一棵含有n个结点的二叉树，采用二叉链表存储；  
2.输出前序、中序和后续遍历该二叉树的遍历结果；

3.采用孩子兄弟表示法建立一棵树；

4.基于树的孩子兄弟表示法实现前序和后序遍历树的操作。

**三、代码实现：**

Bitee.h

#ifndef BiTree\_H

#define BiTree\_H

struct BiNode

{

char data;

BiNode\* lchild, \* rchild;

};

class BiTree

{

public:

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

~BiTree() { Release(root); }

void PreOrder() { PreOrder(root);}

void InOrder() { InOrder(root); }

void PostOrDer() { PostOrder(root); }

private:

BiNode\* root;

BiNode\* Creat(BiNode\* bt);

void Release(BiNode\* bt);

void PreOrder(BiNode\* bt);

void InOrder(BiNode\* bt);

void PostOrder(BiNode\* bt);

};

#endif

Bitree.cpp

#include<iostream>

using namespace std;

#include"Bitree.h"

BiNode\* BiTree::Creat(BiNode\*bt)

{

char ch;

cout << "请输入创建一棵二叉树的结点数据" << endl;

cin >> ch;

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

else {

bt = new BiNode;

bt->data = ch;

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

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

}

return bt;

}

void BiTree::Release(BiNode\* bt)

{

if (bt != NULL) {

Release(bt->lchild);

Release(bt->rchild);

delete bt;

}

}

void BiTree::PreOrder(BiNode\* bt)

{

if (bt == NULL)return;

else {

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

PreOrder(bt->lchild);

PreOrder(bt->rchild);

}

}

void BiTree::InOrder(BiNode\* bt)

{

if (bt == NULL)return;

else {

InOrder(bt->lchild);

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

InOrder(bt->rchild);

}

}

void BiTree::PostOrder(BiNode\* bt)

{

if (bt == NULL)return;

else {

PostOrder(bt->lchild);

PostOrder(bt->rchild);

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

}

}

Bitree\_main.cpp

#include<iostream>

using namespace std;

#include"Bitree.h"

int main()

{

BiTree T;

cout << "------前序遍历------" << endl;

T.PreOrder();

cout << endl;

cout << "------中序遍历------" << endl;

T.InOrder();

cout << endl;

cout << "------后序遍历------" << endl;

T.PostOrDer();

cout << endl;

return 0;

}

Tree.h

#ifndef Tree\_H

#define Tree\_H

const int Max = 20;

struct TNode

{

char data;

TNode\* firstchild, \* rightsib;

};

class Tree

{

public:

Tree();

~Tree() { Release(root); }

void PreOrder() { PreOrder(root); }

void PostOrder() { PostOrder(root); }

private:

TNode\* root;

void Release(TNode\* bt);

void PreOrder(TNode\* bt);

void PostOrder(TNode\* bt);

};

#endif

Tree.cpp

#include<iostream>

using namespace std;

#include "Tree.h"

#include <stdio.h>

Tree::Tree()

{

TNode\* Q[Max] = { NULL };

int front = -1, rear = -1;

char ch1 = '#', ch2 = '#';

TNode\* p = NULL, \* q = NULL;

cout << "请输入根结点：";

cin >> ch1;

p = new TNode; p->data = ch1;

p->firstchild = p->rightsib = NULL;

root = p;

Q[++rear] = p;

cout << "请输入结点对，以空格分隔：";

fflush(stdin);

getchar(); ch1 = getchar(),getchar(); ch2 = getchar();

while (ch1 != '#' || ch2 != '#')

{

p = new TNode; p->data = ch2;

p->firstchild = p->rightsib = NULL;

Q[++rear] = p;

while (front != rear)

{

q = Q[front + 1];

if (q->data != ch1)

front++;

else

{

if (q->firstchild == NULL)

q->firstchild = p;

else

{

q = q->firstchild;

while (q->rightsib != NULL)

q = q->rightsib;

q->rightsib = p;

}

break;

}

}

cout << "请输入结点对，以空格分隔：";

fflush(stdin);

getchar(); ch1 = getchar(); getchar(); ch2 = getchar();

}

}

void Tree::Release(TNode\* bt)

{

if (bt == NULL)return;

else

{

Release(bt->firstchild);

Release(bt->rightsib);

delete bt;

}

}

void Tree::PreOrder(TNode\* bt)

{

if (bt == NULL)return;

else

{

cout << bt->data;

PreOrder(bt->firstchild);

PreOrder(bt->rightsib);

}

}

void Tree::PostOrder(TNode\* bt)

{

if (bt == NULL)return;

else

{

PostOrder(bt->firstchild);

PostOrder(bt->rightsib);

cout << bt->data;

}

}

Tree\_main.cpp

#include<iostream>

using namespace std;

#include"Tree.h"

int main()

{

Tree t1;

t1.PreOrder();

cout << endl;

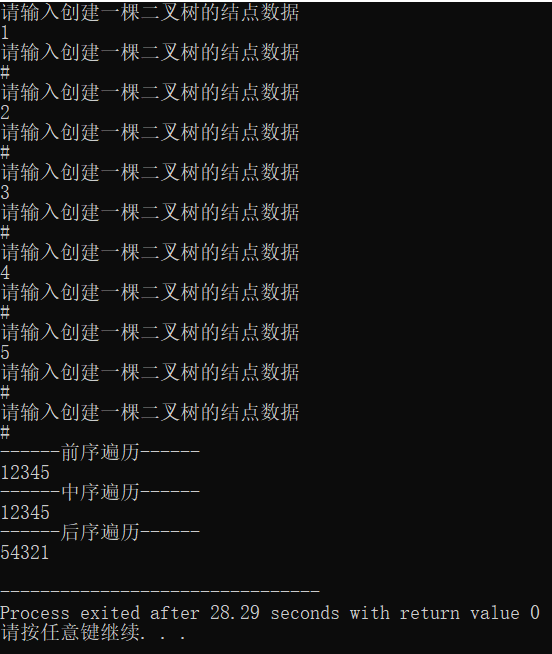
t1.PostOrder();

cout << endl;

return 0;

}

**四、结果截图：**

****

