北京邮电大学软件学院

2016-2017学年第1学期实验报告

课程名称：\_算法与数据结构\_

实验名称：\_树及其应用\_

实验完成人： \_刘浩博\_

姓名：\_刘浩博\_ 学号：\_2015212086\_ 成绩：\_\_\_\_\_\_\_\_\_\_\_

指导老师： \_贾红娓\_

日期： 2016 年 11 月 24 日

⼀.**实验⽬的**

树是⼀种应⽤极为⼴泛的数据结构，也是这门课程的重点。它们的特点在于⾮线性。

⼴义表本质上是树结构。本章实验继续突出了数据结构加操作的程序设计观点，但根

据这两种结构的⾮线性特点，将操作进⼀步集中在遍历操作上，因为遍历操作是其他

众多操作的基础。遍历逻辑的（或符号形式的）结构，访问动作可是任何操作。本次

实验希望帮助学⽣熟悉各种存储结构的特征，以及如何应⽤树结构解决具体问题（即

原理与应⽤的结合）。

**⼆. 实验内容**

**二叉树的建⽴立与遍历**

**[问题描述]**

　　建⽴⼀棵⼆叉树，并对其进⾏遍历（先序、中序、后序），打印输出遍历结果。

**[基本要求]**

　　从键盘接受输⼊（先序），以⼆叉链表作为存储结构，建⽴⼆叉树（以先序来建⽴），

并采⽤递归算法对其进⾏遍历（先序、中序、后序），将遍历结果打印输出。

**[测试数据]**

ABCффDEфGффFффф（其中ф表⽰空格字符）则输出结果为

先序：ABCDEGF

　　中序：CBEGDFA

　　后序：CGBFDBA

**三. 实验环境**

Dev-C++ 5.11

**四. 实验结果**

达到实验要求，程序可完成实验目的，且无错误出现。

**五. 附录**

**BiTree\_Traverse.cpp:**

#include"stdio.h"

#include"stdlib.h"

#include<iostream>

using namespace std;

typedef struct BiTNode{

char data;

struct BiTNode \*lchild,\*rchild;

}BiTNode,\*BiTree;

int CreateBiTree(BiTree &T){

char ch;

scanf("%c",&ch);

if(ch==' ') T=NULL;

else{

if(!(T=(BiTree)malloc(sizeof(BiTNode)))) exit(0);

T->data=ch;

CreateBiTree(T->lchild);

CreateBiTree(T->rchild);

}

return 0;

}

int Visit(char e){

cout<<e;

return 1;

}

int InOrderTraverse(BiTree T,int(\*Visit)(char e)){

if(T){

if(InOrderTraverse(T->lchild,Visit))

if(Visit(T->data))

if(InOrderTraverse(T->rchild,Visit))

return 1;

return 0;

}

else return 1;

}

int PreOrderTraverse(BiTree T,int(\*Visit)(char e)){

if(T){

if(Visit(T->data))

if(PreOrderTraverse(T->lchild,Visit))

if(PreOrderTraverse(T->rchild,Visit))

return 1;

return 0;

}

else return 1;

}

int PostOrderTraverse(BiTree T,int(\*Visit)(char e)){

if(T){

if(PostOrderTraverse(T->lchild,Visit))

if(PostOrderTraverse(T->rchild,Visit))

if(Visit(T->data))

return 1;

return 0;

}

else return 1;

}

int main(){

BiTree T;

cout<<"Please input a binary tree,in Pre Order:";

CreateBiTree(T);

cout<<"Pre Order:";

PreOrderTraverse(T,Visit);

cout<<endl<<"In Order:";

InOrderTraverse(T,Visit);

cout<<endl<<"Post Order:";

PostOrderTraverse(T,Visit);

return 0;

}

**结果截图：**

