**吉首大学 软件学院**

**《数据结构与算法》实验报告**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **实验名称** | **树的基本操作** | | | **实验地点** | **3320** |
| **指导老师** | **张延亮** | **实验时间** | **20181130** | **提交时间** | **20181130** |
| **班　　级** | **17级5班** | **姓　　名** | **周文龙** | **学　　号** | **2017401320** |

|  |
| --- |
| 1. **实验目的** 2. 学会实现二叉树结点结构和对二叉树的基本操作。 3. 通过对二叉树遍历操作的实现，理解二叉树各种操作。 4. 学会利用递归方法编写对二叉树等类似递归数据结构进行处理的算法。 5. **实验内容**   (1)编写程序，输入二叉树的结点值（用#表示空指针或空树），构建二叉树。  (2)编写程序，采用中序遍历的递归和非递归算法对此二叉树进行遍历。  (3)编写程序，统计二叉树中结点的个数、统计二叉树中叶子结点的个数、计算二叉树的高度   1. **实验原理及结果**   **#include <stdio.h>**  **#include <stdlib.h>**  **typedef char ElemType;**  **typedef struct BiTNode**  **{**  **ElemType data;**  **struct BiTNode \*lchild;**  **struct BiTNode \*rchild;**  **}BiTNode,\*BiTree;**  **int leafcount;**  **int allcount;**  **void CreateBiTree(BiTree \*T)**  **{**  **char c;**  **scanf("%c",&c);**  **if(c=='#')**  **\*T = NULL;**  **else**  **{**  **\*T = (BiTNode\*)malloc(sizeof(BiTNode));**  **if(!T)**  **exit(0);**  **(\*T)->data = c;**  **CreateBiTree(&(\*T)->lchild);**  **CreateBiTree(&(\*T)->rchild);**  **}**  **}**  **void InOrderTraverse(BiTree T)**  **{**  **if(T == NULL)**  **return;**  **InOrderTraverse(T->lchild);**  **printf("%c",T->data);**  **allcount++;**  **InOrderTraverse(T->rchild);**  **}**  **int IsEmpty\_BiTree(BiTree \*T)**  **{**  **if(\*T == NULL)**  **return 1;**  **else**  **return 0;**  **}**  **void LevelOrderTraverse(BiTree T)**  **{**  **int front = 0;**  **int rear = 0;**  **BiTree BiQueue[20];**  **BiTree tempNode;**  **if(!IsEmpty\_BiTree(&T))**  **{**  **BiQueue[rear++] = T;**  **while(front!=rear)**  **{**  **tempNode = BiQueue[front++];**  **if(!IsEmpty\_BiTree(&(tempNode->lchild)))**  **BiQueue[rear++] = tempNode->lchild;**  **if(!IsEmpty\_BiTree(&(tempNode->rchild)))**  **BiQueue[rear++] = tempNode->rchild;**  **printf("%c",tempNode->data);**  **}**  **}**  **}**  **void Leafcount(BiTree T)**  **{**  **if(T)**  **{**  **if(T->lchild==NULL&&T->rchild==NULL)**  **leafcount++;**  **Leafcount(T->lchild);**  **Leafcount(T->rchild);**  **}**  **}**  **int TreeDeep(BiTree T)**  **{**  **int deep = 0;**  **if(T)**  **{**  **int leftdeep = TreeDeep(T->lchild);**  **int rightdeep = TreeDeep(T->rchild);**  **deep = leftdeep+1 > rightdeep+1 ? leftdeep+1 : rightdeep+1;**  **}**  **return deep;**  **}**  **int main()**  **{**  **BiTree T = NULL;**  **CreateBiTree(&T);**  **printf("中序遍历结果：");**  **InOrderTraverse(T);**  **printf("\n");**  **Leafcount(T);**  **printf("二叉树中结点个数：%d\n",allcount);**  **printf("二叉树中叶子结点个数：%d\n",leafcount);**  **int d = TreeDeep(T);**  **printf("二叉树的高度：%d\n",d);**  **return 0;**  **}**     1. **实验心得** |