**Project 7 线索二叉树**

学号 2014211116 姓名 骆金参 上交时间 2015-11-21

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 题号 | 程序逻辑(40) | 算法新颖性(20) | 代码规范 (20) | 实验报告(20) | 总分 |
| 得分 |  |  |  |  |  |

1. **问题描述**

线索二叉树的中序线索化，中序遍历二叉线索树T(头结点)的非递归算法

**2. 算法描述**

**/\* 中序遍历进行中序线索化 \*/**

**//Exercise 1: Thread the binary three from inOrder traversal**

**void InThreading(BiThrTree p)**

**{**

**if(p) {**

**InThreading(p->lchild); /\* 递归左子树线索化 \*/**

**if(!p->lchild) { /\* 没有左孩子 \*/**

**p->LTag=Thread; /\* 前驱线索 \*/**

**p->lchild=pre; /\* 左孩子指针指向前驱 \*/**

**}**

**if(!pre->rchild) {/\* 前驱没有右孩子 \*/**

**pre->RTag=Thread; /\* 后继线索 \*/**

**pre->rchild=p; /\* 前驱右孩子指针指向后继(当前结点p) \*/**

**}**

**pre=p; /\* 保持pre指向p的前驱 \*/**

**InThreading(p->rchild); /\* 递归右子树线索化 \*/**

**}**

**}**

**/\* 中序遍历二叉线索树T(头结点)的非递归算法 \*/**

**//Exercise 2: Implement non-recursive inOrder traversal using Threaded Binary tree**

**Status InOrderTraverse\_Thr(BiThrTree T)**

**{**

**BiThrTree p;**

**p=T->lchild; /\* p指向根结点 \*/**

**while(p!=T) { /\* 空树或遍历结束时,p==T \*/**

**while(p->LTag==Link) p=p->lchild;**

**if(!visit(p->data)) return ERROR; /\* 访问其左子树为空的结点 \*/**

**while(p->RTag==Thread&&p->rchild!=T) {**

**p=p->rchild;**

**visit(p->data); /\* 访问后继结点 \*/**

**}**

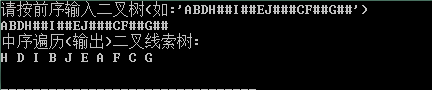
**p=p->rchild;**

**}**

**return OK;**

**}**

**3.测试结果**



**4. 分析与评论**

线索二叉树利用二叉链表中的空指针域，存放指向结点在某种遍历次序下的前趋和后继结点的指针，比二叉树更深入和复杂，感觉很混乱、、、