

数据结构实验提交代码的接口要求

1 实验二功能调用序号

序号	功能	键盘输入\n	屏幕输出\n
0	退出程序	0	无
29	setBTnum(num); 设置当前二叉树序号: 0<num 序号<10;	29 num	无
1	InitBiTree(T); 初始化二叉树	1	无
2	DestroyBiTree(T); 销毁二叉树	2	无
3	CreateBiTree(T,definition); 创建二叉树	3 def	无
4	ClearBiTree (T); 清空二叉树	4	无
5	BiTreeEmpty(T); 判定空二叉树	5	1 或 0
6	BiTreeDepth(T); 求二叉树深度	6	树深度
7	Root(T); 获得根结点	7	e0
8	Value(T,e); 获得当前结点的值	8	e0
9	Assign(T,&e,value); 结点赋值	9 e	无
10	Parent(T,e); 获得双亲结点	10	e 或 ^, 注 1
11	LeftChild(T,e); 获得左孩子结点	11	e 或 ^, 注 1
12	RightChild(T,e); 获得右孩子结点	12	e 或 ^, 注 1
13	LeftSibling(T,e); 获得左兄弟结点	13	e 或 ^, 注 1
14	RightSibling(T,e); 获得右兄弟结点	14	e 或 ^, 注 1
15	InsertChild(T,p,LR,c); 插入子树	15 L/R Tnum	无, 注 2
16	DeleteChild(T,p,LR); 删除子树	16 L/R	无
17	PreOrderTraverse(T,Visit()); 前序递归遍历	17	e 序列
18	InOrderTraverse(T,Visit()); 中序非递归遍历	18	e 序列
19	PostOrderTraverse(T,Visit()); 后序遍历	19	e 序列
20	LevelOrderTraverse(T,Visit()); 按层遍历	20	e 序列
31	CreateHTree(HTnum); 建立哈夫曼树	31 def	无
32	HTreeCode(string); 字符序列编码	32 string	code 序列
33	HTreeDecode(code); 解码为字符序列	33 code	string
34	StringCheck(string); 字符序列编码解码比对	34 string	1 或 0, 注 3

35	CodeCheck(code); 编码解码与编码比对	35 code	1 或 0, 注 3
----	----------------------------	------------	------------

注 1: 访问左右孩子和兄弟时, 如果存在, 则当前结点改变为所访问的结点, 如果左右孩子或兄弟不存在, 则当前结点仍保持, 不改变为空结点, 但是输出显示为空指针^;

注 2: L/R 表示采用 L, 或采用 R; Tnum 为子树的编号, 且该子树根的右孩子为空; 当子树 Tnum 插入到其他树后, 原来保存的指向该 Tnum 子树根结点 root 的指针改为 NULL;

注 3: 输入一个字符串/正确的编码串, 然后程序对输入进行编码/解码, 接着对得到的结果进行解码/编码, 即可以还原出原来最先输入的字符串/编码串, 程序自己检验还原的结果是否相符; 注意开始输入的编码串要确保是正确的, 即是我们前面输入进去的哈夫曼树对应的编码;

2 实验二数据格式约定

- 1) 数据结点的 data 为字符类型; 输出时, 若结点为空, 则输出^;
- 2) 判断是否正确, 若是, 输出 1, 非, 输出 0;
- 3) 二叉树/哈夫曼树的输入表达式 def 的格式为: 先序含空结点^的遍历序列;
- 4) 待编码的字符序列 string 的格式为: e1e2....., 字符之间没有空格, 除非本身含有空格;
- 5) 待解码的编码序列 code 的格式为: c1c2....., 编码串连续, 之间没有空格;
- 6) 选择当前二叉树/插入子树的功能中, num 为树的序号, Tnum 为子树所在的序号, 该子树的右子树为空;

3 功能实现的限制性说明

- 1) 确认使用二叉链表的功能来构建上述的二叉树, 构建哈夫曼树时可以增加扩展域;
- 2) 前序遍历使用递归方式实现;
- 3) 中序遍历使用非递归方式实现;
- 4) 构建哈夫曼树的功能 31 比构建二叉树功能 3 有更多的内容需要处理: 需要对叶子结点建立索引, 或形成类似线索指针的链表, 可以勾链形成类似三叉链表的可以快速找到双亲结点的指针等;
 - 因需要使用哈夫曼树进行编码, 可以通过遍历的方式找到叶子结点并构建索引, 便于今后编码的使用 (如可以使用类似线索指针的方式, 或者建立指针数组的方式);
 - 因需要使用哈夫曼树进行编码, 可以通过遍历的方式找到结点的双亲, 也可以专门扩展双亲指针域, 并在哈夫曼树构建后再来填充该指针域, 但是在二叉树的各项功能中均不能使用该指针域;

4 输入输出样例

样例 1: (层序遍历求深度)

用例序号	输入	输入含义	输出	输出含义
1	29 2 1	选择第 2 棵二叉树 初始化	0 4 A	不是空的二叉树 树的深度为 4 根结点为 A

	3 AD^EF^^G^^B^C^^ 5 6 7 20 2 0	创建二叉树 含空二叉树的格式 判断是否为空的二叉树 求树的深度 求根结点 按层序遍历二叉树 销毁二叉树 退出程序	ADBECFG	层序遍历的结果
--	---	---	---------	---------

样例 2：（访问左右孩子）

用例序号	输入	输入含义	输出	输出含义
1	29 3 1 3 AD^EF^^G^^B^C^^ 7 11 12 11 12 0	选择第 3 棵二叉树 初始化 创建二叉树 含空二叉树的格式 求根结点 求左孩子 求右孩子 求左孩子 求右孩子 退出程序	A D E F ^	根结点为 A 左孩子 D 右孩子 E 左孩子 F 右孩子空

样例 6：（哈夫曼树输入/后序遍历）

用例序号	输入	输入含义	输出	输出含义
1	29 2 1 31 akghiC^^U^^H^^j R^^S^^I^^mI^^N ^^znopqB^^G^^rP ^^sV^^tuvJ^^Q^^ wX^^Z^^K^^O^^x A^^yD^^L^^bcdeY ^^W^^fM^^F^^T^ ^ ^^ 6 19 2 0	选择第 2 棵二叉树 初始化 创建哈夫曼树 含空二叉树的格式 求树的深度 按后序遍历二叉树 销毁二叉树 退出程序	11 CUIHhRSj gElNmlkB GqPVJQv XZwuKtsr pOoADLy xnYWeMF fdTc bza	树的深度为 11 后序遍历的结果

F

- F



- F