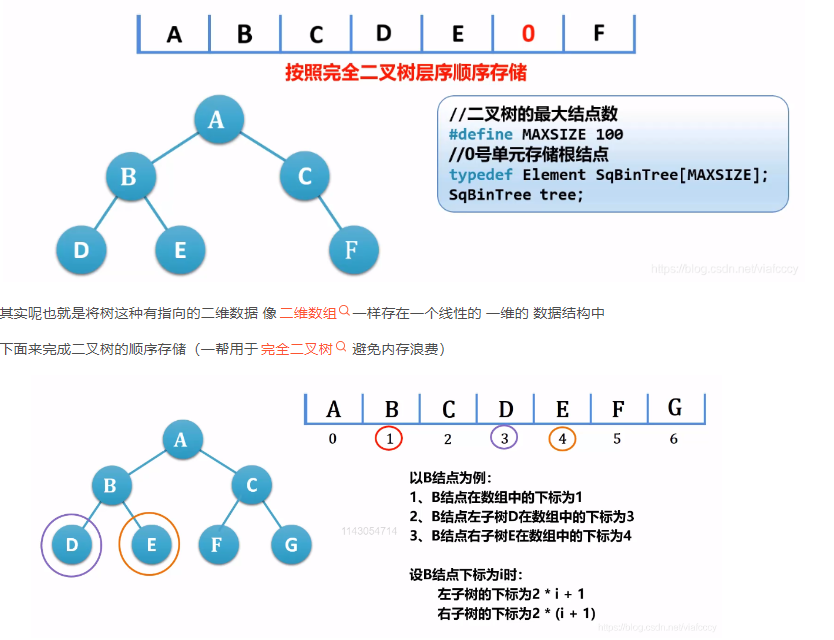
**实 验 报 告**

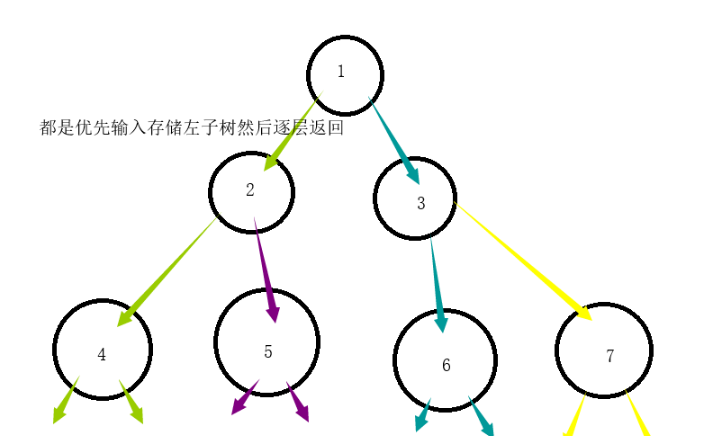
|  |  |  |
| --- | --- | --- |
| **课程名称**：数据结构 | **班级**：软工21101 | **实验成绩**： |
| **实验名称**：二叉树的建立与应用 | **学号**：2105006207 | **批阅教师签字：董傲霜** |
| **实验编号**：实验七 | **姓名**：方福涛 | **实验日期：**2022 年 10 月 28日 |
| **指导教师**：董傲霜 | **组号**：wuyanzu | **实验时间**： 时 分－ 时 分 |

**一、实验设计思想**

1. 说明二叉树的抽象数据类型的特征；
2. 结点是二叉树的基础,通常主要用结点保存与应用有关的信息.
3. 作为二叉树的表示,还需要记录二叉树的结构信息,至少需要保证能检查结点的父子关系,例如,能从一个结点找到其左/右子结点
4. 说明二叉树的顺序存储结构的设计思想。

用一组地址连续的存储单元依次**自上而下、自左至右 （即存层序序列）** 存储完全二叉树的结点。**仅适用于完全二叉树**（在排序和查找中经常用到）。因为对非完全二叉树可能对存储空间造成极大的浪费，比如单支树。





**二、程序说明**

1. 给出二叉树的顺序存储结构定义及其属性含义的说明；

typedef struct {

char element[MAXSIZE + 1];//静态数组存储元素

int maxNum;//二叉树中结点的最大编号

}BiTree；

1. 给出你设计的函数的结构说明（函数，参数，执行的结果）；  
    void initBiTree(BiTree\*T)//初始化

void clearBiTree(BiTree\*T)//适配器设计

void createBiTree(BiTree\*T)//创建树

void outputBiTree(BiTree\*T)//输出

void preOrder(BiTree\*T,int rnum)//先序遍历二叉树

void inOrder(BiTree\*T,int rnum)//中序遍历二叉树

void PostOrder(BiTree root)//后序遍历二叉树

int leaf(BiTree\* T,int rnum)// 计算叶子数

1. 说明二叉树遍历操作实现步骤（处理步骤，可用文字也可画流程图）。

 先序遍历：

1.先访问根节点

2.再先序遍历左子树

3.最后先序遍历右子树

中序遍历：

1.先中序遍历左子树

2.再访问根节点

3.最后中序遍历右子树

后序遍历：

1.先后序遍历左子树

2.再后序遍历右子树

3.最后访问根节点

**三、实验环境**

Windows10、devc++

**四、实验过程分析**

实验调试过程中的问题及改正的事件举例说明（只要调程序就不可能没有错误，现象、原因、修正方法，可以截图但不能过多）。

分析：

1.二叉树是否为空？

2.二叉树当前节点时候为空

3.插入当前节点，需要判断是否为空

**五、实验结果总结**

1、阐述二叉树遍历操作的算法时间复杂度和空间复杂度。

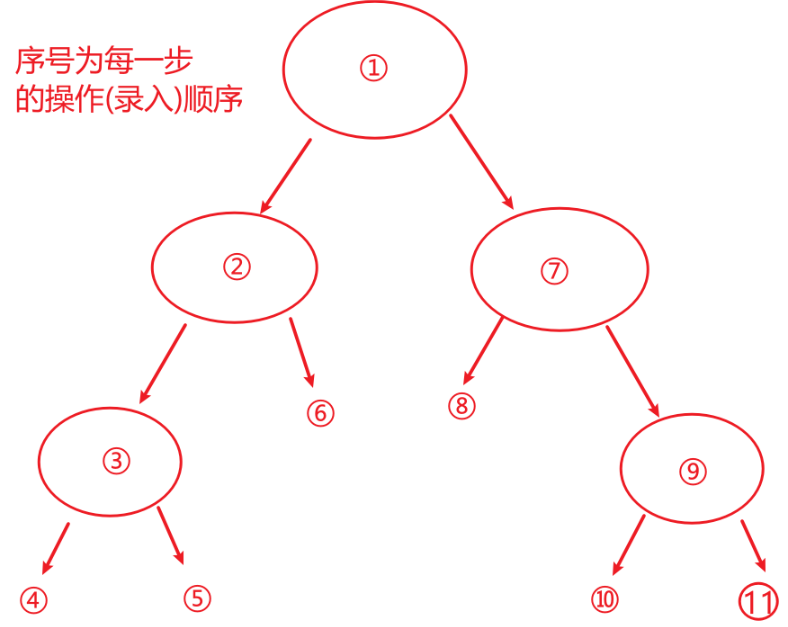
非递归/递归遍历：

时间复杂度：O(N)

空间复杂度：O(N)

2、给出你的测试方法和测试截图（不能超过三个截图）；

这里假设我们要录入的数据分别是1，2，3，4因为每次都是从左子树开始录入的，所以我们的录入顺序应该是1 2 3 0 0 0 4 0 0。这个很好理解，第一个0代表3的左子树为空，因此就不会继续沿着这条路往下走了，而是转向右边，第二个0代表3的左子树也为空，因此我们沿着原路返回到2，因为2的左子树我们已经录入了，右子树现在还没有处理，因此我们现在要处理2的右子树，第三个0代表的是2的右子树也是空，于是相同的过程，我们又返回到上一级，录入右子树的值，这时右子树不为空，OK，录入右子树的数据，然后又继续往左子树开始录，第四个0代表4的左子树为空，于是转向右子树，第五个0代表4的右子树也为空，我们还是会返回到上一级，返回上一级发现，上一级的左右子树都已经满了，继续往上找，发现没有上一级了，代表现在已经回溯到根结点了，此时代表数据已经全部录入完毕。图解如下，重点是理解。



**六、附录**

1. 意见和建议（没有可不写）。

2. 思考题：

回答以下问题：

1. 什么是完全二叉树？

一棵深度为k的有n个结点的二叉树，对树中的结点按从上至下、从左到右的顺序进行编号，如果编号为i（1≤i≤n）的结点与满二叉树中编号为i的结点在二叉树中的位置相同，则这棵二叉树称为完全二叉树。

1. 完全二叉树的高度和结点个数之间的关系？

1.具有n个(n>0)结点的完全二叉树的高度h为：log2(n+1)

2. 具有n个(n>0)结点的完全二叉树的高度h为：h = log2n + 1

**七、打分表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 项目 | 考核点 | 分数 | **得分** | 备注 |
| 程序 | 逻辑是否正确  程序可读性  创新点 | 50 | **0** |  |
| 报告完整性 | 实验过程阐述是否完整  测试数据设计是否合理  运行结果是否正确 | 40 | **0** |  |
| 调试问题及解决方法 | 是否对调试过程问题进行阐述 | 5 | **0** |  |
| 思考题目 | 回答是否正确 | 5 | **0** |  |
| 合计 |  | 100 | **0** |  |