**实 验 报 告**

**课程名称：** 数据结构

**实验项目：** 二叉树的生成与操作

**实验仪器：** PC机

**学 院：** 计算机学院

**专 业：** 数据科学与大数据

**班级姓名：** 大数据1902/尹露

**学 号：** 2019011182

**日 期：** 2020年11月20日

**指导教师：** 范艳芳

**同组成员：** 无

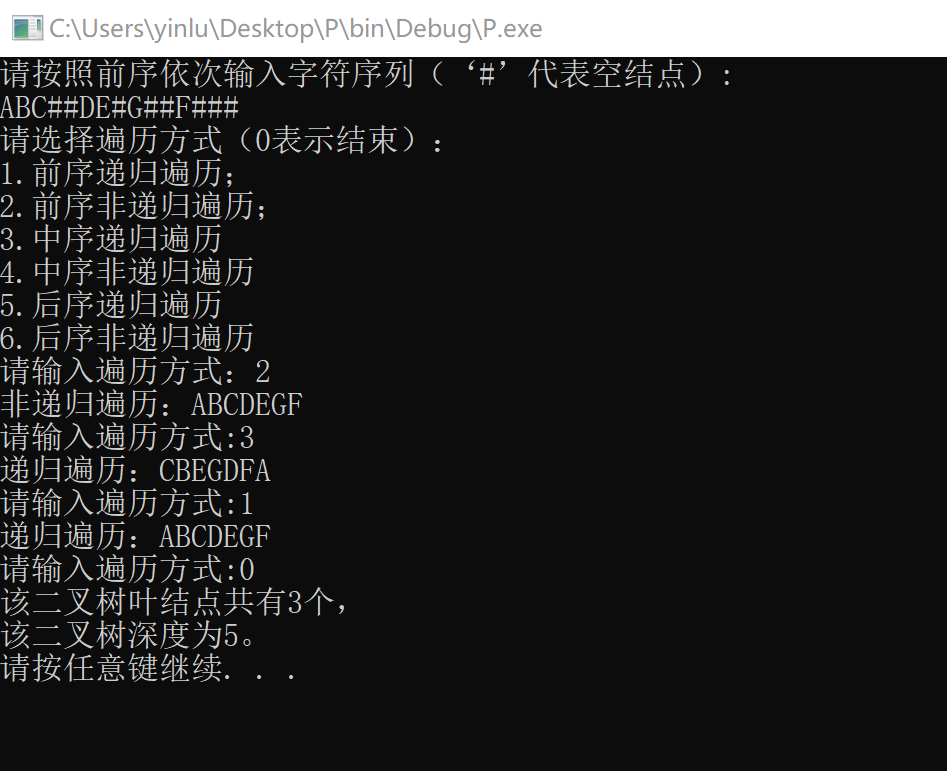
**成 绩：**

1. **实验目的**

本次实验的主要目的在于帮助学生掌握二叉树的建立和存储实现、二叉树的遍历思想。

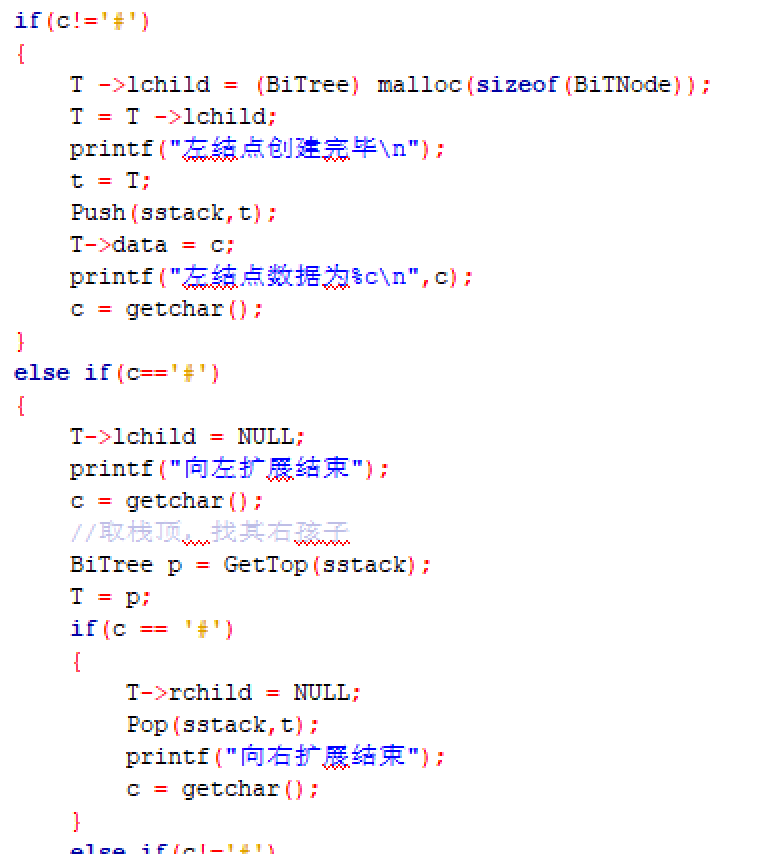
1. **实验内容**
2. 由用户输入字符序列（为某二叉树的扩展二叉树的前序遍历序列），生成该序列对应的二叉树，并用二叉链表存储，在此基础上利用递归和非递归方法进行该二叉树的遍历。
3. 提供三种次序的递归遍历方法：前序遍历、中序遍历、后序遍历，并输出遍历结果。
4. 提供两种次序的非递归遍历方法：前序遍历、中序遍历。用到的栈操作可使用在实验二中定义的栈。
5. 应提供用户操作界面，以便于由用户选择进行何种方式的遍历。
6. 通过非递归方法实现后序遍历。
7. 计算二叉树中叶节点的数量，计算二叉树的深度。
8. **设计思想**
9. 不管是创建二叉树，遍历还是统计叶结点、计算深度，都需要依次遍历左子树和右子树，所以全部使用递归方法，方便操作，但是较难理解，占用内存较大。
10. 创建时，依次读取字符型数据，将其值赋给当前结点的data。
11. 在非递归遍历时，要使用栈。指针指向的结点要先存储，之后取栈顶元素或者让栈顶元素出栈，才能继续访问这个结点或者是其右子树。
12. 在计算树深时，递归计算其左、右子树的深度，最大深度+1即为该二叉树的深度。
13. **算法框架**
14. 创建二叉树：用户输入为前序遍历顺序，getchar()依次读取输入的字符，若为’#’，则当前树为空；若不为’#’，则为其分配结点空间，将其值赋给当前结点的data。创建完根节点，依次去创建其左子树和右子树，递归实现。
15. 前序递归遍历： 当前结点不空，则访问该结点，依次访问其左子树和右子树。左子树中，也是先访问根节点，再依次访问左子树和右子树。依次递归。
16. 中序递归遍历：同样当前结点不空，则访问其左子树再访问该结点，访问右子树。
17. 后序递归遍历：则先访问左右子树，最后访问根结点。
18. 前序非递归遍历：
19. 访问结点后，一路向左至最左端，且全部入栈。直至结点为空。
20. 此时栈顶元素出栈，去访问其右孩子及右子树。
21. 循环步骤1）。
22. 遍历结束标志为：当前访问结点为空，且栈空，无元素可取。
23. 中序非递归遍历：与前序类似，先一路向左入栈，结点为空时，元素出栈，访问根结点后，再遍历右子树。
24. 后序非递归遍历：
25. 不访问结点一路向左，全部入栈。
26. 结点为空时，取栈顶元素。
27. 若栈顶元素的右孩子存在，且未被访问（使用临时指针来记录上一个访问的结点），则访问其右子树；若其右子树已经被访问，则栈顶元素出栈，访问当前结点。
28. 循环步骤1.
29. 遍历结束标志为：当前访问结点为空，且栈空，无元素可取。
30. **调研报告**

1.运行截图：



1. 问题及解决
2. 在一开始时，创建二叉树使用的是非递归算法。发现不管怎样，情况总是考虑不完全——写了好几十行还是感觉是刚开始。

后来实在写不动了，在室友的提醒下，发现了书上的递归创建方法。这是这次试验最大的问题所在。



1. 在写非递归算法时，需要用到栈。在实验二中，我定义的栈为指针类型。所以在本次实验中，就尽量避免类型再次出现指针类型。但是发现，存入非指针类型时，当取栈顶元素或者元素出栈时，不能直接将其赋值给当前访问结点T，所以最后还是选择使用BiTree结点指针类型。
2. 在后序非递归遍历时，当左右的子树都遍历完成时，要将移动指针p置为NULL，再继续操作。
3. 在设计用户界面时，可能实操的是最多的。从一开始的无需用户选择全部输出，到后来的用户通过数字选择输出选项，再到最后的用户循环输入至0结束，一步一步来符合用户的输入习惯。
4. 对于程序中的临时结点或者是不再使用的结点，要及时释放内存。
5. **实验总结**
6. 理论总结：
7. 从一定程度来讲，二叉树的创建过程，相当于把树当做满二叉树来创建，有一个结点为空时，即代表这棵树为空。
8. 二叉树区别于树的特点：度数最大为2；必须区分左右子树。
9. 二叉树也存在顺序存储，但是用处不大。首先，对于本实验中的用户数据，只有左子树的情况，就会造成极大的内存浪费。
10. 这次实验的实操中，使用自己上次实验所定义的栈，在学习二叉树的同时复习了栈的相关知识。

（2）思考与创新

1）其实发现用递归还可以做很多好玩的事情，比如左右子树反转。使用第三变量分别赋值后，递归；比如输出根结点到某一结点的路径。

2）对于层序遍历，可以使用辅助队列。一个结点出队，若左右结点均存在，则都入队。后序可以实现这个。

3）其实我们做的相对来说比较少，还有线索二叉树、哈夫曼树，求哈夫曼编码等等。可以在课余时间实现。