山东大学网络空间安全学院

数据结构与算法 课程实验报告

|  |  |  |  |
| --- | --- | --- | --- |
| 学号：202400460042 | 姓名：江李阳 | | 班级：1班 |
| 1. 实验题目：实现二叉树前序、中序、后序遍历的递归算法；统计二叉树度为1的节点个数（递归和非递归算法）； | | | |
| 实验学时：2 | | 实验日期：2025.3.28 | |
| 实验目的： 根据二叉树、队列的数据结构特点，熟练运用递归算法，实现二叉树的遍历和对度为一的节点的统计。 | | | |
| 硬件环境：内存：16384MB 处理器：13th Gen intel(R) Core(TM) i5-13500HX (20 CPUs),~2.5GHz | | | |
| 软件环境：windows系统，visual studio | | | |
| 实验步骤与内容：  分析问题:  题目一要求使用递归算法实现对二叉树的遍历，递归的好处在于代码简洁，实现简单，并不需要知道具体的实现，但需要确定好递归调用的顺序。前序顺序为：根左右，在访问到根节点就输出数据，然后分别对将左子树和右子树当作新的根节点参数，调用前序输出，此时会将根节点的左子树当作新的根节点，输出数据，让后将这个左子树的左右子树作为参数，传入前序输出函数。退出条件为节点为空，这样的循环递归调用，就能实现对整棵树的前序遍历。中序遍历和后续遍历的思路类似，只需要调整一下数据输出和递归调用的顺序。  题目二要求统计二叉树树度为1的节点的个数，基本思路是遍历二叉树的每一个节点，判断每个节点是否满足树度为1，满足就令总数+1，不满足就忽略。  一个节点满足树度为1，即这个节点的左孩子为空且右孩子不为空，或左孩子不为空且右孩子子为空。代码思路为：遍历节点->判断是否符合条件->得出结果。   1. 递归实现时，先判断节点，符合就让计数变量增加；然后分别将当前节点的非空左、右孩子当作参数传入这个函数，计数结果加等于递归调用的结果，最终遍历完整棵树，计数结果就保留了根节点和所有左右子树中树度为1的节点个数。 2. 非递归实现时，利用已有的队列的结构。判断方式相同，不同点在于遍历节点时，首先将根节点入队，然后在循环里，首先将队列头元素出队，判断这个头元素代表的节点是否符合要求并做相应计数处理，最后将这个节点的左右非空孩子入队。循环的判断条件就是队列是否为空。这样能保证遍历完整棵树的所有节点，并且最后返回正确结果。   代码实现：  按照上面的思路编写相应代码，在利用队列时要创建并初始化一个临时队列，出队操作时要有临时节点变量来接收出队元素。测试几组数据得到预期结果，问题成功解决。  分析计算二叉树深度代码：   1. 递归实现时先声明计数变量为-1，判断节点不为空则让分别将左右孩子作为参数，递归的调用这个计算深度的函数，结果分别赋值给临时计数变量。函数最后返回两个标记左右子树中深度更大的那个值加一，代表整棵树的深度。 2. 非递归实现中，首先创建一个队列用来储存节点，并把根节点入队。外层循环令len等于队列长度，确保内层循环每次能将同一层的都出队后，再进项下一步操作，同时让计算深度的变量deep递增。内层循环声明临时节点储存出队元素，并把这个出队元素代表的节点的非空左右孩子入队，确保能遍历所有节点。最后遍历完所有节点，deep按树的层数递增，最后返回代表树的深度。 | | | |
| 结论分析与体会：通过这个实验，利用递归和非递归的方法实现对二叉树的基本操作，让我对进一步掌握二叉树的结构特点和操作技巧。对比使用递归和非递归算法，配合队列使用，让我对这些实现有了更深刻的认识。递归实现代码简洁，但逻辑性强，数据太大可能会出现调用栈溢出现象；非递归实现代码冗长，但逻辑严谨，能处理更大的输入数据，两种方法各有适用场景。 | | | |