INT10 2W4 _ 二 叉 树

定 义 ： 二 叉 树 是 一 种 数 据 的 结 构 ， 通 常 其 中 元 素 的 部 分 定 义 如 下

节点的度：一个节点含有的子树的个数称为该节点的度。

树的度：一棵树中，最大的节点的度称为树的度。

叶节点或终端节点：度为零的节点称为叶节点。

父亲节点或父节点：若一个节点含有子节点，则这个节点称为其子节点的父节点

孩子节点或子节点：一个节点含有的子树的根节点称为该节点的子节点。

兄弟节点：具有相同父节点的节点互称为兄弟节点。

节点的层次：从根开始定义起，根为第1层，根的子节点为第2层， 以此类推。

树的高度或深度：树中节点的最大层次称为树的深度或高度。

堂兄弟节点：父节点在同一层的节点互为堂兄弟。

节点的祖先：从根到该节点所经分支上的所有节点。

子孙： 以某节点为根的子树中任一节点都称为该节点的子孙。

森林： 由m（m>=0）棵互不相交的树的集合称为森林。

二叉树的分类：

二叉搜索树（Binary Search Tree）：在二叉搜索树中，每个节点的所有左子节点的值都小于或等于该节点的值，而所有右子节点的值都大于或等于该节点的值。
平衡二叉树（Balanced Binary Tree）：平衡二叉树是一种特殊的二叉搜索树，其中每个节点的两个子树的高度差最多为1。AVL树和红黑树是平衡二叉树的两个例子。
堆（Heap）：堆是一种特殊的完全二叉树，其中每个节点的值都大于或等于（在最大堆中）或小于或等于（在最小堆中）其子节点的值。
哈夫曼树（Huffman Tree）：哈夫曼树是一种优化用于数据压缩的二叉树，其中频率最高的元素位于树的最低深度。
B树和B+树：这些是用于数据库和文件系统的自平衡二叉搜索树的扩展。

本周我们学习的内容只是简单的二叉概念，并没有提及某样特定的树，但是我感觉应该下周会大概率讲解堆和二叉搜索

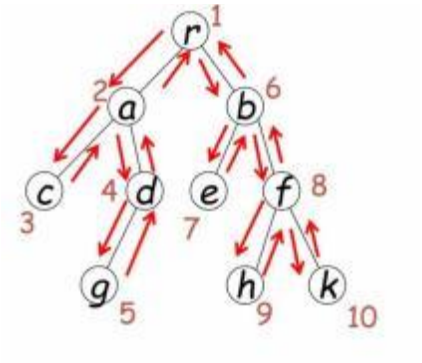
时间复杂度：二叉树是二分思想的体现，因此在大多数情况下他和我们归并排序中排序的部分相同，为O（logn），但是在极端情况下也会出现时间复杂度为O

（n）的特殊情况，比如在二叉搜索中你将一个从1到5的整数数组插入树时你会发现他退化成了一个链表，在这种情况下我们将会得到O（n）的时间复杂度

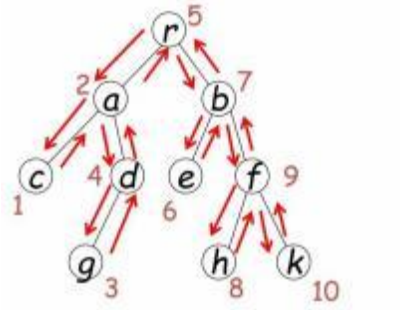
为了避免这种情况，可以使用一种叫做“平衡二叉搜索树” 的数据结构（例如 AVL 树或红黑树）

二叉树的遍历：这里我们首先需要知道他的遍历其实就是我们之前学习的BFS和DFS的思想体现，在DFS层次我们存在前序，中序和后序，而在BFS层次我们只存在一个叫做层次遍历的东西

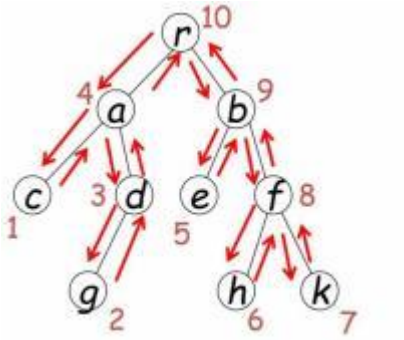
1.前序遍历：前序遍历就是首先访问根节点，然后递归地做前序遍历左子树和右子树。遍历的顺序是：**根节点 -> 左子树 -> 右子树，同时他也存在回溯这一 现象（所有DFS都有），如图所示**



2. **中序遍历：**首先递归地做中序遍历左子树，然后访问根节点，最后递归地做中 序遍历右子树。遍历的顺序是：**左子树 -> 根节点 -> 右子树**。这一遍历方式 会让我们得到一个升序的序列，遍历的示意如下



3.**后序遍历：**先递归地做后序遍历左子树和右子树，然后访问根节点。遍历的顺 序是：**左子树 -> 右子树 -> 根节点**



4. 层次遍历：层次遍历也被称为广度优先遍历。在这种遍历方法中，我们首先访问根节点，然后访问所有同一层次的节点，从左到右。然后是下一层的节点，以此类推，具体和我之前的那张一石激起千层浪的图差不多。