研讨报告 2

姓名：郑博远　学号：2154312　日期：2022 年 11 月 10 日

1. 研讨内容

1. 1 二叉排序树

它或者是一棵空树，或者是具有下列性质的二叉树：

若左子树不空，则左子树上所有结点的值均小于它的根节点的值；

若右子树不空，则右子树上所有结点的值均大于它的根节点的值；

左、右子树也分别为二叉排序树。

（1）请给出一棵二叉排序树；

（2）如何利用该二叉排序树得到一个递升序列？

（3）如何利用该二叉排序树得到一个递减序列？

组内讨论：

（1）如下图所示，图 1 是一棵而二叉排序树：

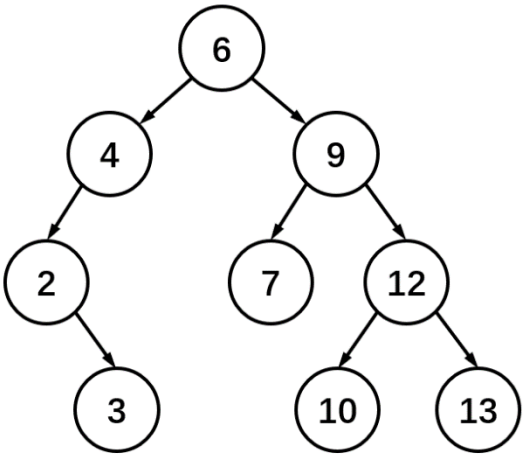

图 1　二叉排序树

上述二叉排序树满足二叉排序树性质：非空的左子树所有结点值均小于它的根节点的值；非空的右子树所有结点的值均大于它的根节点的值。

(2) 中序遍历二叉排序树, 即以左子树→根节点→右子树的顺序遍历排序树。遍历图 1 中的二叉排序树得到序列 2 3 4 6 7 9 10 12 13，是递升序列;

(3) 逆中序遍历二叉排序树, 以右子树→根节点→左子树的顺序遍历排序树。遍历图 1 中的二叉排序树得到序列 13 12 10 9 7 6 4 3 2，是递减序列;

个人思考：

（1） 二叉排序树的插入

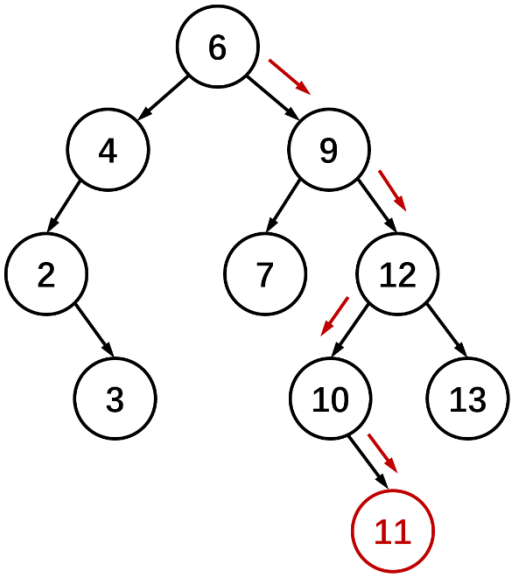

图 2　二叉排序树的插入

从根结点开始遍历, 若当前结点元素值小于要插入的元素, 则遍历其左子树; 若当前结点元素值大于要插入的元素，则遍历其右子树。直到遍历的子树为空, 则将元素插入到该位置。如图 2 所示，在原有二叉排序树中插入新元素 11。

（2） 二叉排序树的删除

若当前删除的结点没有左右子树，则直接将其删除即可;

若当前删除的结点只有左子树或右子树, 则将该结点删除, 其子树接到该处;

若当前删除的结点既有左子树，又有右子树，则左子树保持不变，在右子树中寻找最小的元素 (递归遍历左子树直到为空)。将该元素改到删除结点的位置,

即完成结点的删除。如图 3 所示，删除元素 9，将右子树中最小元素 10 放到删除结点 9 的位置。

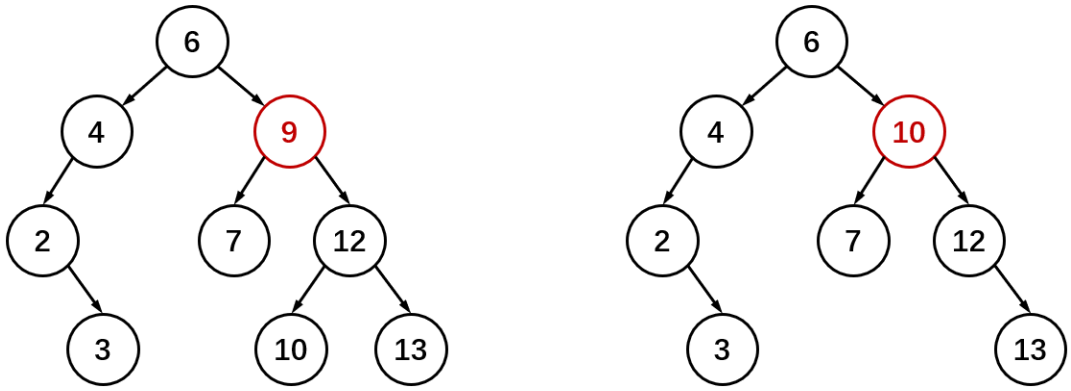

图 3　二叉排序树的删除

（3）　二叉排序树的优化

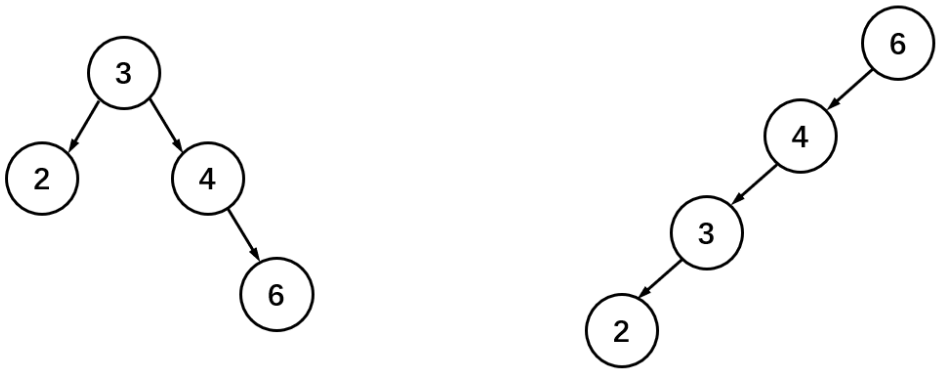

图 4　退化成链表的二叉树

当二叉树比较平衡时，每次查找的时间复杂度为 O(logn)；但若当如果二叉树原本是一个单调性比较好的串，那么存储的时候会退化成链表，使得依次遍历的查找时间复杂度上升到 O(n)。此时我们可以通过平衡二叉树（AVL 树）优化。

平衡二叉树的特点如下：

1. 它是一棵空树或它的左右两个子树的高度差的绝对值不超过 1；

2. 左右两个子树都是一棵平衡二叉树。

在平衡二叉树的建立上，有如下方法：

1. 左旋：新插入节点在最小不平衡子树根节点的右儿子的右子树上。

根节点成为右儿子点的左子树；右儿子原本的左子树成为根节点的右子树。

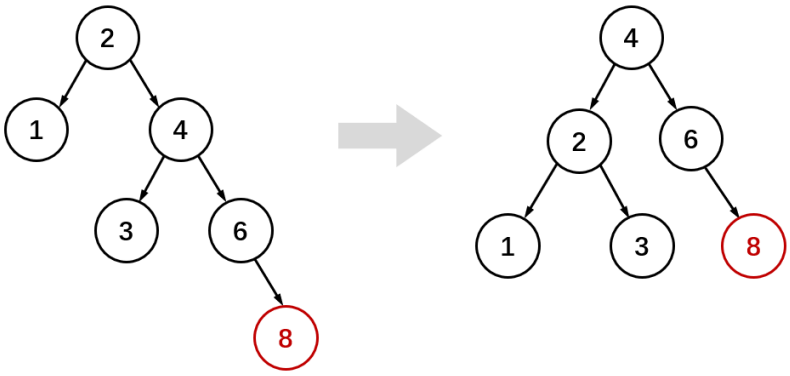

图 5　左旋

2. 右旋：新插入节点在最小不平衡子树根节点的左儿子的左子树上。

根节点成为左儿子点的右子树；左儿子原本的右子树成为根节点的左子树。

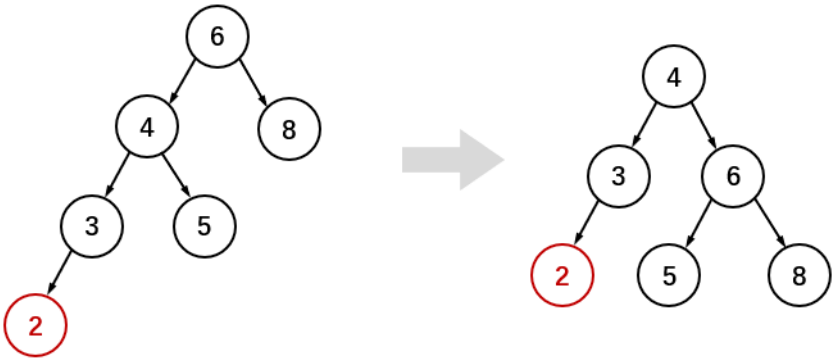

图 6　右旋

3. 双旋：新插入节点在最小不平衡子树根节点的左儿子的右子树上（或反之）。

以左儿子为根节点进行左旋，再按原始的根节点右旋（或反之）。

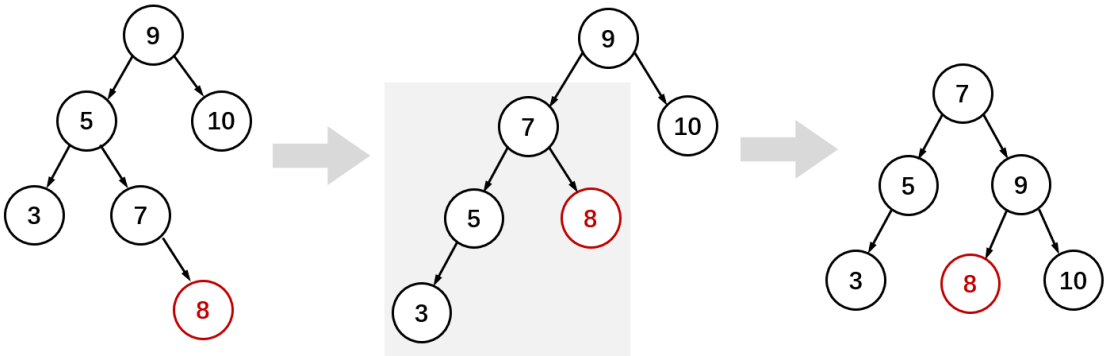

图 7　双旋

1. 2 树形结构在文件管理中的应用

(1) 怎样利用树形结构来管理文件目录，并能够将文件和文件夹区分；

(2) 如何统计一个节点下文件夹和文件的数目；

(3) 从目录树的管理上看，要实现文件夹或文件的删除、复制、移动，请描述算法实现的思路；

(4) 地址路径和目录树结构怎么映射，给定一个地址路径（例如：D:\Program Files\Microsoft Office\Office14），怎么实现定位。反之，给定一个节点，获得相应的路径地址。请描述算法实现的思路。

组内讨论：

（1） 用于文件管理的数据结构如下：

```
struct node{
    TElemType data;
    bool is_file;
    node* parent;
    node* first_child;
    node* next_sibling;
}
```

```
D:\>tree root /f
卷 Data 的文件夹 PATH 列表
卷序列号为 000000F0 2A32:E908
D:\ROOT
|  file1
|
├─dir1
|  |  file2
|  |  file3
|  |
|  └─dir3
|     |  file6
|     |  file7
|     |
|     └─dir5
└─dir2
   |  file4
   |  file5
   |
   └─dir4
      |  file8
      |
      └─dir6
```

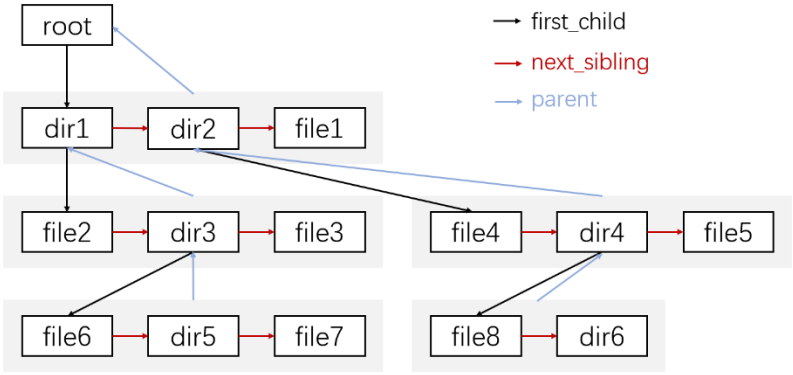

图 8 文件夹数据结构存储示意图

如伪代码所示，每个结点代表一个文件夹/文件的信息。在结点中用一个 bool 变量 is_file 存储其是文件夹或是文件。每个结点有一个 parent 指针指向包含其的上级文件夹，一个 next_sibling 指针指向该文件夹内其下一个文件/文件夹。若该节点存储的是文件夹（非空），则其 first_child 指针指向其包含的第一个文件/文件夹。具体存储方式如图 8 右侧所示。

(2) 若采用链表方式存储每一个节点下的文件夹和文件，则需要遍历链表来计算文件夹与文件的数目，可以用 is_file 变量进行区分计数，时间复杂度 O(n)；若采用 vector 存储每一个节点下的文件夹和文件，则 vector 的 size 即为文件夹与文件的个数；也可以在数据结构中增加记录其节点下文件和文件夹数据的变量，使时间复杂度达到 O(1)，提高查询效率。

(3) 算法实现思路如下：

删除操作：使用后根遍历的方式递归寻找到所有的子文件、文件夹逐一进行删除释放。若当前删除的文件/文件夹为其上级文件夹的第一个文件/文件夹，则将上级文件夹的 first_child 指向删除结点的 next_sibling；否则将其前一个结点的 next_sibling 指向删除结点的 next_sibling；

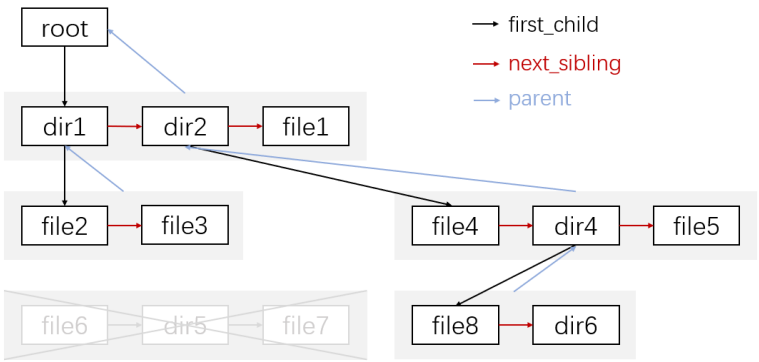

图 9　删除 dir3 文件夹

复制操作：使用先根遍历的方式递归复制所有的子文件与文件夹。若当前插

入的文件/文件夹为其上级文件夹的第一个文件/文件夹，则将上级文件夹的 first_child 指向插入的文件/文件夹结点；否则将其链表中最后一个结点的 next_sibling 指向当前插入的文件/文件夹结点；

移动操作：只需操作与当前移动结点相关的指针即可。若当前移动文件/文件夹插入后为其上级文件夹的第一个文件/文件夹，则将上级文件夹的 first_child 指向插入的文件/文件夹结点；否则将其链表中最后一个结点的 next_sibling 指向当前插入的文件/文件夹结点；

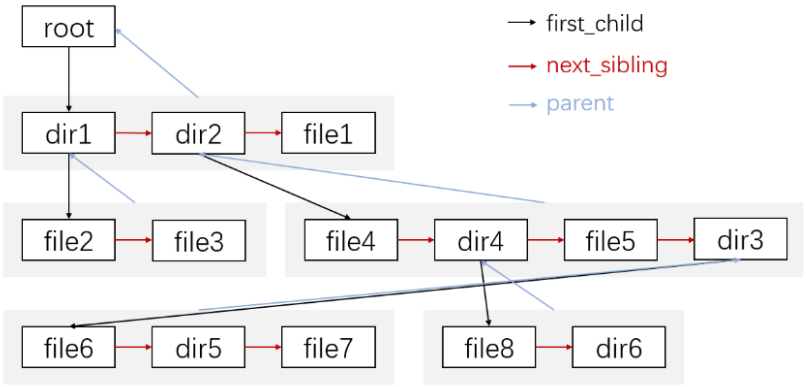

图 10 将 dir3 文件夹移动到 dir2 文件夹下

（4）地址路径遍历：若为绝对路径则从根节点开始逐个进行匹配，遍历便能寻找到对应的节点；若为相对路径则从当前目录或其上级目录开始遍历；

获得路径地址：从当前结点开始，不断访问其 parent 结点直到到达根结点。可以用栈的方式压栈进行存储，最后出栈直到栈为空即可打印路径地址。

个人思考：

在存储当前结点下的所有文件/文件夹时，也可以使用二叉排序树根据字典序来存储，这样能将每个文件夹下查找对应子文件/文件夹的时间复杂度优化到 O(logn)，比起链表存储的 O(n)更高效。

1. 3 有一千万条短信，有重复，以文本文件（ASCII）形式保存，一行一条，请找出重复出现最多的前 10 条。

组内讨论：

方法 1 ： 用字典树来记录每个短信文件的信息，每一个结点有当前字母和出现次数两个元素，在每一个字符串的结束之处使当前的结点出现次数加 1。同时维护一个大小为 10 的结构体数组，记录出现次数前 10 短信的字符串和出现次数，每遍历到新短信就更新前 10 的结构体数组。统计后能直接输出前 10 条。

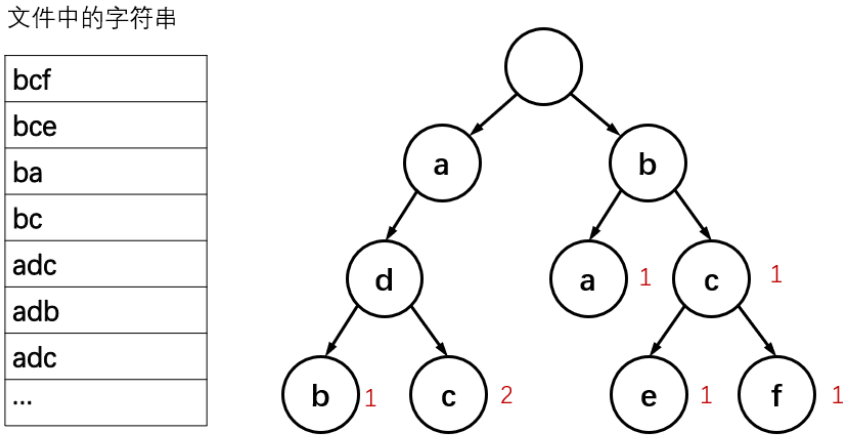

图 11　对短信文件中字符串建立字典树

方法 2：同样维护一个只存储出现次数前 10 的数组。使用哈希表的方式，让每一个短信字符串都对应唯一的哈希值，每次 O(1)访问来存储该字符串的总出现次数。通过此方式，可以使总的时间复杂度降低到 O(n)。

个人思考：

对于两个内容不同的短信字符串，存在 MOD 后其哈希值相同的情况，便会导致哈希冲突。可以采用双 hash 的方式降低出现哈希冲突的概率；也可以采用链地址法将相同哈希值的不同字符串用链表存储来避免冲突。

2. 研讨总结与建议

本次研讨课中，我很荣幸作为我们小组的代表参加了问题讨论结果的汇报。本次研讨的三道题目虽然数量不多，但是在难度上较第一次研讨更高，给了我们更多的思考空间。在讨论过程中，我们小组热情十分高涨，积极提出自己看法的同时，也在网络上查找了一些资料以供参考。本次研讨课营造了良好的讨论氛围，既有组内的看法交流，也有不同组之间互相分享解题思路，让我在巩固所学的知识点之余也能够更深入地去思考问题。

对于之后的研讨课，我有几点小建议：

1. 每组讲解形式上，可以不必拘泥于每一个分享者都轮流一次性分享所有的题目。可以以题目为单位，每个小组进行轮流汇报并补充，这样讨论的针对性更强，分享者与台下的同学思路也更连贯清晰；

2. 讨论课上思路众多，同学们可能无法当堂全部吸收；且同学们分散在各个教室，有的思路只被部分同学了解。可以将比较优秀的思路整理成文档发送在班级的 QQ 群中，让所有同学都能够更深入地了解到优秀的解法思路，有所收获；

3. 对于部分难度较大或者涉及背景比较复杂的讨论题目，可以在研讨课前事先公布。可以选择小组轮流或者自愿报名的方式，对应小组在课前做好准备工作，在上课时向其他同学以介绍的方式进行知识的传授；

4. 在研讨课前，可以通过共享文档的方式邀请同学们填写自己在学习该章节时所遇到的困惑之处，或是想要提出的认为有讨论价值的问题。老师或者助教学长学姐们可以挑选同学们比较有共鸣的问题或是含金量比较高的问题作为研讨课的讨论题目。在每组轮流讨论的过程中，也可以鼓励同学们即时提出自己困惑之处，让讲解的同学及时解答。