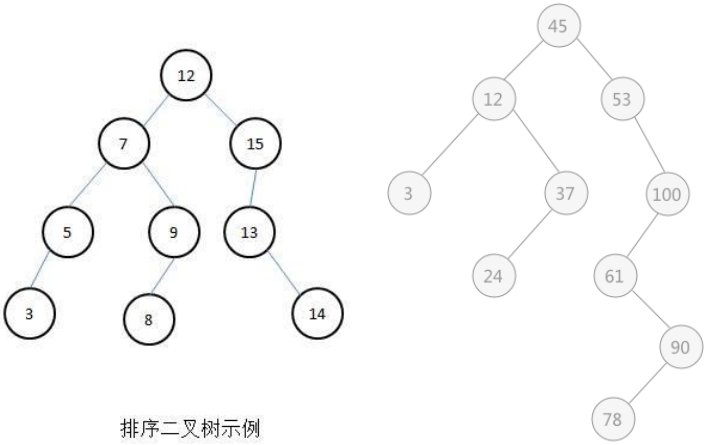
1、 二叉排序树

（1）



排序二叉树示例

（2）

中序遍历

我们知道对于二叉树来说，遍历的方式一般有三种：前序遍历、中序遍历和后序遍历。但对于二叉搜索树，比较常用的是中序遍历，因为通过该种方式遍历出来的元素属于一个递增序列。

排序二叉树的左子节点必然小于等于父节点，右子节点必然大于等于父节点，中序遍历先遍历左子，然后是自身，然后是右子，所以肯定是递增。

（3）

将上述的中序遍历的值放入一个栈中，等全部遍历完全后，再从使得数据出栈，这样就可以得到一个递减序列。

2、 树形结构在文件管理中的运用

（1） 文件管理本身就可以理解为树形结构的概念。如图：

```
Microsoft Windows [版本 10.0.19041.450]
(c) 2020 Microsoft Corporation. 保留所有权利。

C:\Users\cheng>D:

D:\>cd darknet\darknet-master\

D:\darknet\darknet-master>tree
卷 SOFTWARE 的文件夹 PATH 列表
卷序列号为 72F2-2543
D:.
├─cfg
├─data
│  └─labels
├─examples
├─include
├─python
├─scripts
└─src

D:\darknet\darknet-master>
```

文件夹可以被看做有子节点的节点，而文件可以被看做叶子节点，因为它没有子节点。

(2) 采用递归的方法统计节点的个数。当树为空时，叶子结点个数为0
当某个节点的左右子树均为空时，表明该结点为叶子结点，返回1
当某个节点有左子树，或者有右子树时，或者既有左子树又有右子树时，说明该节点不是叶子结点，因此叶结点个数等于左子树中叶子结点个数 加上 右子树中叶子结点的个数

二叉树的叶子节点个数的操作可以使用先序遍历，中序遍历，后序遍历中的任何一种，只需要在将访问操作变成判断该节点是否为叶子节点（既不存在左孩子、也不存在右孩子的节点）
如果是叶子节点，那么累加和加一
如果不是叶子节点，则继续递归判断
文件的数目即为叶节点的个数，再用上述的总节点减去叶节点的个数即为文件夹的个数。

（3） 删除即为删除该节点，以及如果它有子节点的话，也将其一并删除
复制即为新建一个树，根节点为要复制的该节点，如果该节点下面有子节点，则一并复制到新节点里。
移动即为，先复制，再删除，再增加到所要到的新节点的位置上。

(4) 地址路径和目录树的映射即为，按照地址路径的推进，逐层深入对应名称的目录树的各个子节点。反之，给定子节点后，逐层往上找到自己的父节点，一直到根节点，即可从下往上找到对应的地址路径名称。

3、

方法1：可以用哈希表的方法对1千万条分成若干组进行边扫描边建散列表。第一次扫描，取首字节，尾字节，中间随便两字节作为Hash Code，插入到hash table中。并记录其地址和信息长度和重复次数，1千万条信息，记录这几个信息还放得下。同Hash Code且等长就疑似相同，比较一下。相同记录只加1次进hash table，但将重复次数加1。一次扫描以后，已经记录各自的重复次数，进行第二次hash table的处理。用线性时间选择可在O（n）的级别上完成前10条的寻找。分组后每份中的top10必须保证各不相同，可hash来保证，也可直接按hash值的大小来分类。

方法2：可以采用从小到大排序的方法，根据经验，除非是群发的过节短信，否则字数越少的短信出现重复的几率越高。建议从字数少的短信开始找起，比如一开始搜一个字的短信，找出重复出现的top10并分别记录出现次数，然后搜两个字的，依次类推。对于对相同字数的比较常的短信的搜索，除了hash之类的算法外，可以选择只抽取头、中和尾等几个位置的字符进行粗判，因为此种判断方式是为了加快查找速度但未能得到真正期望的top10，因此需要做标记；如此搜索一遍后，可以从各次top10结果中找到备选的top10，如果这top10中有刚才做过标记的，则对其对应字数的所有短信进行精确搜索以找到真正的top10并再次比较。

方法3：可以采用内存映射的办法，首先1千万条短信按现在的短信长度将不会超过1G空间，使用内存映射文件比较合适。可以一次映射（当然如果更大的数据量的话，可以采用分段映射），由于不需要频繁使用文件I/O和频繁分配小内存，这将大大提高数据的加载速度。其次，对每条短信的第i（i从0到70）个字母按ASCII嘛进行分组，其实也就是创建树。i是树的深度，也是短信第i个字母。

该问题主要是解决两方面的内容，一是内容加载，二是短信内容比较。采用文件内存

映射技术可以解决内容加载的性能问题（不仅仅不需要调用文件I/O函数，而且也不需要每读出一条短信都分配一小块内存），而使用树技术可以有效减少比较的次数。