1、有1亿个浮点数，请找出其中对大的10000个。提示：假设每个浮点数占4个字节，1亿个浮点数就要站到相当大的空间，因此不能一次将全部读入内存进行排序。  
解法：对于第一道题目可以发现如果一次读入，那么机器的内存肯定是受不了的，因此我们只有想其他方法解决，解决方式为了高效还是得符合一定的该概率解决，结果并不一定准确，但是应该可以作对大部分的数据。  
(1.我们可以把1亿个浮点数分组为100W个一组，这样就分为了100个组，第一次在每个组中找出最大的1W个数，第二次查询的时候就是100W个数中再找出最大的1W个数。  
PS:100W个数中再找出最大的1W个数用类似快排的思想搞定。  
\*对于第一道的思路一种更效率是：

(1)、读入的头10000个数，直接创建二叉排序树。O(1)

(2)、对以后每个读入的数，比较是否比前10000个数中最小的大。(N次比较)如果小的话接着读下面的数。O(N)

(3)、如果大，查找二叉排序树，找到应当插入的位置。

(4)、删除当前最小的结点。

(5)、重复步骤2，直到10亿个数全都读完。

(6)、按照中序遍历输出当前二叉排序树中的所有10000个数字。

 基本上算法的时间复杂度是O(N)次比较 算法的空间复杂度是10000(常数）  
-----------------------------------------------------------------------------------------------------------  
2、有一篇英文文章(也就是说每个单词之间由空格分隔)，请找出“csdn”着个单词出现的次数，要求效率最高，并写出算法的时间级。  
解法：第二道题目可以把单词看成一个N进制数，CSDN相当于('c'-'a')\*N^3+('s'-'a')\*N^2+('d'-'a')\*N+('n'-'a'),然后查找这个数出现的次数就是答案,也可以建立一颗字典树，然后去计数！  
PS:N可以取32，64等。  
-----------------------------------------------------------------------------------------------------------  
3.假设有1kw个身份证号，以及他们对应的数据。身份证号可能重复，要求找出出现次数最多的身份证号。  
解法：简单进行hash搞定,O(n),如果数据量再扩大我就不知道怎么搞了，用磁盘的话，IO数据是接受不了的。  
-----------------------------------------------------------------------------------------------------------  
4.百度每天都会接受数亿的查询请求, 如何在这么多的查询(Query)中找出高频的Query是一个不小的挑战. 而你的任务则更加艰巨, 你需要在极其有限的资源下来找出这些高频的Query.(使用内存不得多于1MB) 。输入文件是一行一个Query, 以文件结束符结尾。每个Query字节数L(一个汉字两个字节)满足：0

解法：hash，然后建立hash[103][100]的节点的表，每次找出出现次数最少的进行替换。

\*在处理海量数据时hash + 索引 + 倒排非常有用，在内存空间较小的情况下多通道排序也非常有用！另外，在遇到海量数据求高频问题时，还可以采用采样的手段来处理。

 -----------------------------------------------------------------------------------------------------------