3．对这个区间上的数做 bit map 映射，再遍历 bit map，找到一个没出现的数即可。

找到 100 亿个 URL 中重复的 URL 以及搜索词汇的 top *K* 问题

【题目】

有一个包含 100 亿个 URL 的大文件，假设每个 URL 占用 64B，请找出其中所有重复的 URL。

【补充题目】

某搜索公司一天的用户搜索词汇是海量的（百亿数据量），请设计一种求出每天最热 top 100 词汇的可行办法。

【难度】

士 ★☆☆☆

【解答】

原问题的解法使用解决大数据问题的一种常规方法：把大文件通过哈希函数分配到机器，或者通过哈希函数把大文件拆成小文件。一直进行这种划分，直到划分的结果满足资源限制的要求。首先，你要向面试官询问在资源上的限制有哪些，包括内存、计算时间等要求。在明确了限制要求之后，可以将每条 URL 通过哈希函数分配到若干机器或者拆分成若干小文件，这里的“若干”由具体的资源限制来计算出精确的数量。

例如，将 100 亿字节的大文件通过哈希函数分配到 100 台机器上，然后每一台机器分别统计分给自己的 URL 中是否有重复的 URL，同时哈希函数的性质决定了同一条 URL 不可能分给不同的机器；或者在单机上将大文件通过哈希函数拆成 1000 个小文件，对每一个小文件再利用哈希表遍历，找出重复的 URL；或者在分给机器或拆完文件之后，进行排序，排序过后再看是否有重复的 URL 出现。总之，牢记一点，很多大数据问题都离不开分流，要么是哈希函数把大文件的内容分配给不同的机器，要么是哈希函数把大文件拆成小文件，然后处理每一个小数量的集合。

补充问题最开始还是用哈希分流的思路来处理，把包含百亿数据量的词汇文件分流到

不同的机器上，具体多少台机器由面试官规定或者由更多的限制来决定。对每一台机器来说，如果分到的数据量依然很大，比如，内存不够或其他问题，可以再用哈希函数把每台机器的分流文件拆成更小的文件处理。处理每一个小文件的时候，哈希表统计每种词及其词频，哈希表记录建立完成后，再遍历哈希表，遍历哈希表的过程中使用大小为 100 的小根堆来选出每一个小文件的 top 100（整体未排序的 top 100）。每一个小文件都有自己词频的小根堆（整体未排序的 top 100），将小根堆里的词按照词频排序，就得到了每个小文件的排序后 top 100。然后把各个小文件排序后的 top 100 进行外排序或者继续利用小根堆，就可以选出每台机器上的 top 100。不同机器之间的 top100 再进行外排序或者继续利用小根堆，最终求出整个百亿数据量中的 top 100。对于 top K 的问题，除哈希函数分流和用哈希表做词频统计之外，还经常用堆结构和外排序的手段进行处理。

40 亿个非负整数中找到出现两次的数和所有数的中位数

【题目】

32 位无符号整数的范围是 0～4294967295，现在有 40 亿个无符号整数，可以使用最多 1GB 的内存，找出所有出现了两次的数。

【补充题目】

可以使用最多 10MB 的内存，怎么找到这 40 亿个整数的中位数？

【难度】

尉 ★★☆☆

【解答】

对于原问题，可以用 bit map 的方式来表示数出现的情况。具体地说，是申请一个长度为 4294967295×2 的 bit 类型的数组 bitArr，用 2 个位置表示一个数出现的词频，1B 占用 8 个 bit，所以长度为 4294967295×2 的 bit 类型的数组占用 1GB 空间。怎么使用这个 bitArr 数组呢？遍历这 40 亿个无符号数，如果初次遇到 num，就把 bitArr[num*2 + 1]和 bitArr[num*2]设置为 01，如果第二次遇到 num，就把 bitArr[num*2+1]和 bitArr[num*2]设置为 10，如果第三次遇到 num，就把 bitArr[num*2+1]和 bitArr[num*2]设置为 11。以后再遇