1、一个数组中除了一个元素只出现一次外，剩余元素均出现了 2 次，找到这个出现一次的元素。要求：在 O(n)的时间 O(1)的空间。

思路：异或

2、长度为 n 的数组循环左移 k 位。要求：O(n)的时间 O(1)的空间。

思路：$[A^T B^T]^T=[B\ A]$

3、蓄水池抽样。

4、一个数组，大小为 n 里面存储的数的大小为 0-n-1，有些数出现多次，要求找出至少一个出现多次的数。要求：O(n)的时间，O(1)的空间。

思路：尝试将所有的元素 k 都放到其对应的地方(索引为 k 处 ,使得 v[k]=k)

```
1 while(i<=n)
2    if v[i]≠i
3       if v[v[i]]==v[i]
4          value=v[i] and break//该数出现了两次
5       else
6          exchange v[v[i]] and v[i]//会继续当前 i 的循环
7    else
8       i++
```

5、子集和问题，从集合 S 中找出所有和为 sum 的子集（类似于动态规划的问题，**区别在于：当分割成一个数和另一个和的时候，该和可能是不可分的，也就是有些情况是不存在的，需要加以区分**）

1. 对集合 S 进行排序
2. 找出集合中小于 sum 的最大元素的索引 n
3. 在索引 1-n 中（集合 S 的子集 Sub）中寻找和为 sum 的子集

```
List<Set> L
SubSum(S,sum,L)
1 sum<S[1]
2    return
3 int n=MaxBeneathSum(S)
4 if sum==S[n]
5    L.add(sum,sum)
4 for i=1 to n
5    SubSum(S,sum-S[i],List)
```