下面是一个二人小游戏：桌子上有M堆扑克牌；每堆牌的数量分别为Ni(i=1…M)；两人轮流进行；每走一步可以任意选择一堆并取走其中的任意张牌；桌子上的扑克全部取光，则游戏结束；最后一次取牌的人为胜者。   
现在我们不想研究到底先手为胜还是为负，我只想问大家：   
——“先手的人如果想赢，第一步有几种选择呢？”

**Input**

输入数据包含多个测试用例，每个测试用例占2行，首先一行包含一个整数M(1<M<=100)，表示扑克牌的堆数，紧接着一行包含M个整数Ni(1<=Ni<=1000000，i=1…M)，分别表示M堆扑克的数量。M为0则表示输入数据的结束。

**Output**

如果先手的人能赢，请输出他第一步可行的方案数，否则请输出0，每个实例的输出占一行。

**Sample Input**

3

5 7 9

0

**Sample Output**

1

题解：考察的是对尼姆博弈本质的理解。如果先手能赢则异或值不为0，先手的第一步肯定是通过

　　　修改一个数的值把这些数的异或值变为0。假设有n个数，设其中一个数为a是要减少的数，

　　　那么a的值必须大于除了a以外的n-1个数的异或值才可以，只有大于才能通过a的减少使得

　　　所有数的异或值为0，如果等于那说明所有数的异或值为0，先手必败。如果小于那么无论怎么减少a都无法使得总异或值为0，二进制的表示中一定会有至少一个1存在着。

　　　所以遍历每个数，看哪些数是刚才所述的符合条件的a，统计这样的数有多少个即为最终结果。

#include<stdio.h>

#include<iostream>

#include<cstring>

#include<set>

#include<map>

typedef long long ll;

using namespace std;

int main()

{

//freopen("input.txt","r",stdin);

int m;

int a[110];

while(cin>>m && m)

{

int ans(0);

for(int i=0;i<m;i++)

{

cin>>a[i];

ans^=a[i];

}

int cnt(0);

for(int i=0;i<m;i++)

{

if(a[i]>(ans^a[i]))

cnt++;

}

cout<<cnt<<"\n";

}

return 0;

}