[报告]J - I NEED A OFFER!

[source]

http://202.114.18.202:8080/judge/contest/view.action?cid=6165#problem/B

[Description]

Nim is a 2-player game featuring several piles of stones. Players alternate turns, and on his/her turn, a player's move consists of removing *one or more stones* from any single pile. Play ends when all the stones have been removed, at which point the last player to have moved is declared the winner. Given a position in Nim, your task is to determine how many winning moves there are in that position.

A position in Nim is called "losing" if the first player to move from that position would lose if both sides played perfectly. A "winning move," then, is a move that leaves the game in a losing position. There is a famous theorem that classifies all losing positions. Suppose a Nim position contains n piles having k_1, k_2, …, k_n stones respectively; in such a position, there are $k_1 + k_2 + \ldots + k_n$ possible moves. We write each k_i in binary (base 2). Then, the Nim position is losing if and only if, among all the k_i's, there are an even number of 1's in each digit position. In other words, the Nim position is losing if and only if the *xor*of the k_i's is 0.

Consider the position with three piles given by $k_1 = 7$, $k_2 = 11$, and $k_3 = 13$. In binary, these values are as follows:

```
 111
1011
1101
```

There are an odd number of 1's among the rightmost digits, so this position is not losing. However, suppose k_3 were changed to be 12. Then, there would be exactly two 1's in each digit position, and thus, the Nim position would become losing. Since a winning move is any move that leaves the game in a losing position, it follows that removing one stone from the third pile is a winning move when $k_1 = 7$, $k_2 = 11$, and $k_3 = 13$. In fact, there are exactly three winning moves from this position: namely removing one stone from any of the three piles.

[Solution]

是 nim 的一个变化。令 ans=a1^a2^...^an，第一步的方法数就是减少某一堆的值使 ans=0 的方法数，如果需要找出异或值为 0 的数，所以对于某堆石子 ai,现在对于 ans^ai，就是除了 ai 以外其他的石子的异或值，如果 ans^ai<=ai，那么对于 ai 的话，是可以减小到 ans^ai 的值，然后使得所有数的异或值为 0。

[Code]

```
#include<cstdio>
#include<cstdlib>
int main()
{
    int a[10000];
    int n,t;
```

```c
    int ans=0;
    while (scanf("%d",&n))
    {
        if (!n) break;
        ans=0;
        t=0;
        for (int i=1;i<=n;i++)
          {
            scanf("%d",&a[i]);
            t=t^a[i];
          }
        if (t==0)
          printf("0\n");
        else
          {
             for (int i=1;i<=n;i++)
               {
                  if ((t^a[i])<=a[i])
                    ans++;
               }
             printf("%d\n",ans);
          }

    }
    return 0;
}
```