

同时，因为线性基是一个集合，所以也可以通过合并集合的方式进行快速合并

hdu 3949 Xor ( 线性基 解第k小子集异或和）

题意： T组测试，每次测试给n个long long 型数字，做Q次询问第k小的子集异或和值。

线性基的异或和是无法进位，A[i] > all{ 0……i-1的所有线性基的异或组合}，

实际相当于二进制，求第k最小，实际求k的二进制表达中为1的项位置.

(第i位不为0即对应于线性基中第i位不为0的基在要选的组合中。

必须注意的是： 线性基无法凑成子集异或和为0的情况。所以我们在建线性基组的时候，一定要判定是否在添加过程中，没有参与A中线性基的构建，直接被现有的基消为0，就表示有为0的情况。

(我们的二进制遍历是专门设计解决第k小异或和不为0的子集异或和，所以有0的情况，就是找第k-1小异或和）

如果二进制完后，k不为0，即表示询问范围超出了结果的个数，输出-1。

#include<bits/stdc++.h>

#define ll long long

#define inf 1e18

#define mn 100005

using namespace std;

ll ans,n,m,flag=0;

ll a[mn],tmp[mn];

void ins(ll x)

{ll i;

for(i=m;~i;i--)

if(x&(1ll<<i))

if(a[i]==0){a[i]=x;return ;}

else x^=a[i];

flag=1;//表示线性基里至少有一个

}

bool check(ll x)

{ll i;

for(i=m;~i;i--)

if(x&(1ll<<i))

if(a[i]==0)return 1;

else x^=a[i];

return 1;

}

ll qmax()

{ll ret=0,i;

for(i=m;~i;i--)

ret=max(ret,ret^a[i]);

return ret;

}

ll qmin()

{ll i;

if(flag)return 0;

for(i=0;i<=m;i++)//找最小值从低位到高位

if(a[i])return a[i];

}

ll query(ll k)

{ll res=0,cnt=0,i,j;

k-=flag;//因为是从第0位开始的，有元素的话就-1

if(!k)return 0;

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

{

for(j=i-1;~j;j--)

if(a[i]&(1ll<<j))a[i]^=a[j];

if(a[i])tmp[cnt++]=a[i];

}

if(k>=(1ll<<cnt))return -1;

for(i=0;i<cnt;i++)

if(k&(1ll<<i))res^=tmp[i];

return res;

}

int main()

{

ll x,y,z,i,j,k;

char ch;

cin>>n;

m=60;//这个一般取决于log(值域)的大小

for(i=1;i<=n;i++)

{

scanf("%lld",&x);

ins(x);

}

cout<<qmax();

return 0;

}