A Simple Game

~~防AK题，快润！~~

注意到 xxcdsg 想要最小化最终得分，那么在他的回合，他一定会将所有序列取完。

证明：为简化表述，本段提到的所有的得分均表示从 xxcdsg 开始的回合之后的得分总和。假设在 xxcdsg 的回合内，剩余的序列的异或和为 S_1，而 xxcdsg 如果没有取完所有序列，假设他选择的子序列的异或和为 A，剩下未选择的序列的异或和为 B。根据异或的性质，有 $S_1=A\oplus B$。下一回合，因为 Taibo 会最大化游戏得分，Taibo 可以选择剩余所有的序列，或者在选择其他子序列有更优答案的情况下选择其他子序列，所以得分一定不会小于 $A+B$。因此，对 xxcdsg 来说，选择全部序列的得分为 S_1，选择其他子序列的得分 S_2 至少为 $A+B$。因为 $S_1=A\oplus B\le A+B\le S_2$，为了最小化得分，xxcdsg 会选择全部序列。

根据这一事实，我们可以把问题转化为：把一个序列划分为两个子序列 A_1,A_2，使得这两个子序列的异或和相加最大（易知此时这两个子序列是否为空已经没有影响了）。假设 s,s_1,s_2 分别表示原序列，A_1,A_2 的异或和。那么，最终得分为 $s_1+s_2=s_1+(s\oplus s_1)$。

因为 $a_i<2^{20}$，根据异或的性质可知 $s,s_1<2^{20}$。因此，我们可以先求出整个序列的异或和 s，然后使用异或线性基枚举所有可能的 s_1 并维护得分的最大值，时间复杂度为 $O(nk+k2^k)$，其中 k 为输入序列的最大位数。

```
#include<bits/stdc++.h>
#define Nmax 200010
#define int long long
#define IOS ios::sync_with_stdio(false);cin.tie(nullptr);cout.tie(nullptr);
using namespace std;
int n,xor_sum,a[Nmax],b[25];
void build()
{
    xor_sum=0;
    for(int i=0;i<=20;i++)
        b[i]=0;
    for(int i=1;i<=n;i++)
    {
        xor_sum^=a[i];
        int tmp=a[i];
        for(int j=20;j>=0;j--)
        if((1ll<<j)&tmp)
        {
            if(b[j]==0)
            {
                b[j]=tmp;
                break;
            }
            else
                tmp^=b[j];
        }
    }
}
bool find(int x)
{
    int res=x;
    for(int i=20;i>=0;i--)
    if((1ll<<i)&res)
        res^=b[i];
    return res==0;
}
signed main()
{
    IOS;
    cin>>n;
    for(int i=1;i<=n;i++)
```

```
        cin>>a[i];
    build();
    int ans=0;
    for(int i=0;i<=(1ll<<20);i++)
    if(find(i))
        ans=max(ans,i+(xor_sum^i));
    cout<<ans;
}
```