博弈：

解题模型：

1.把原游戏分解成多个独立的子游戏，则原游戏的SG函数值是它的所有子游戏的SG函数值的异或。

即sg(x)=sg(a1)^sg(a2)^...^sg(an)。

SG值的计算：

1.可选步数为1~m的连续整数，直接取模即可，SG(x) = x % (m+1);

2.可选步数为任意步，SG(x) = x;

3.可选步数为一系列不连续的数，用模板计算。

*//f[]*：可以取走的石子个数

*//sg[]:0~n*的*SG*函数值

*//hash[]:mex{}*

**int** f[N],sg[N],hash[ N];

**void** getSG(**int** n)

{

memset(sg,0,**sizeof**(sg));

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

{

memset(hash,0,**sizeof**(hash));

**for**(**int** j=0; f[j]<=i && j < k; j++) *//k*是*f[]*的有效长度

hash[sg[i-f[j]]]=1;

**for**(**int** j=0; ; j++) *//*求*mes{}*中未出现的最小的非负整数

{

**if**(hash[j]==0)

{

sg[i]=j;

**break**;

}

}

}

}

**int** sg[N];*//sg*值数组

**int** XOR[N];*//*储存每一个二进制位上的和

**int** x;*//sg*值的临时变量

**int** cnt;*//*储存当前*sg*值有多少位的临时变量

**int** ma;*//*最大的*sg*值位数

**bool** cal(**int** n,**int** m)*//n*表示*sg*数组的大小*,*从*1*到*n,m*表示每次可以取*1*到*m*堆

{

memset(XOR, 0, **sizeof** XOR);

ma = -1;

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

{

x = sg[i];

cnt = 0;

**while** (x)

{

XOR[cnt] += x&1;

cnt++;

x >>= 1;

}

ma = max(ma, cnt);

}

**for** (**int** i = 0; i < ma; i++) **if** (XOR[i] % (m + 1)) **return** 1;*//*返回*1*表示先手必胜

**return** 0;*//*返回*0*表示先手必败

}