博弈汇总：<http://www.cnblogs.com/frog112111/p/3199073.html>

博弈论 SG函数详解与一些题型：<https://blog.csdn.net/strangedbly/article/details/51137432>

石子类问题总结：<https://blog.csdn.net/gvolv/article/details/47758109>

SG函数模板：<http://www.cnblogs.com/frog112111/p/3199780.html>

博弈论各种题型总结：<https://www.cnblogs.com/Randolph87/p/5804798.html>





**package** 暑期集训;

**import** java.util.Scanner;

**public** **class** Main {

**static** **int** *N*=1001;

**static** **int**[] *F*=**new** **int**[*N*];

**static** **int**[] *vis*=**new** **int**[*N*];

**static** **int**[] *sg*=**new** **int**[*N*];

**public** **static** **void** memset(**int**[] sg,**int** n,**int** len) {

**for**(**int** i=0;i<len;i++) {

sg[i]=n;

}

}

**public** **static** **void** getSG(**int** n) {

*memset*(*sg*,0,*sg*.length); //错误1：忘记在 i之前 初始化sg[]

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

*memset*(*vis*,0,*vis*.length);//错误2：忘记在 j之前 初始化vis[]

**for**(**int** j=1;*F*[j]<=i;j++) {

*vis*[*sg*[i-*F*[j]]]=1; //错误3：[]中的是 sg[i-F[j]];

}

**for**(**int** j=0;j<=n;j++)

{

**if**(*vis*[j]==0) {

*sg*[i]=j;

**break**;

}

}

}

}

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

Scanner in=**new** Scanner(System.***in***);

*F*[0]=*F*[1]=1;

**for**(**int** i=2;i<=18;i++) {

*F*[i]=*F*[i-1]+*F*[i-2]; /\*错误4：应当注意把F[0]=F[1]=第一种可行操作

但是 注意循环次数,必须使得F[]有超过n的元素\*/

}

**while**(in.hasNext()) {

**int** m=in.nextInt();

**int** n=in.nextInt();

**int** p=in.nextInt();

*getSG*(1000);

**if**(m==0&&n==0&&p==0)**break**;

**if**((*sg*[m]^*sg*[n]^*sg*[p])==0) {

System.***out***.println("Nacci");

}**else** {

System.***out***.println("Fibo");

}

}

}

}

3.Nimk Game

每次最多可以同时从k堆石子进行操作，这k堆可以取不同数量的石子

一堆石子的SG函数为石子数

对每一个二进制位单独算，求SG函数每一个二进制位1的个数mod(k+1)，如果都为0，则必败，否则必胜

证明：

对于必败态不管怎么走都只能走到必胜态

对于变化的SG的最高位，你至少变化为1，最多变化为k，所以这一位1的个数不可能mod(k+1)还是为0

对于必胜态我们肯定可以找到一种方法走到必败态

我们从高位往低位做，记s为这一位可以随意填值的数字个数（如果把某一位从1变成0，那么更低位就能随便取值了）

假设我们现在做到第k位，记n为除了能随便取值的s位以外这一位1的个数mod(k+1)

如果n+s<=k，那么很简单，我们取出n个第k位为1的让这些数字的第k位变成0，那s个数字这一位也变成0，然后s+=n

如果n+s>k，即n+s>=k+1，那么s>=k+1-n，我们在s中间取k+1-n个变为1，其他变为0就可以满足条件了

4.Anti-Nim Game

不能取石子的一方获胜

必胜：SG不为0且至少有一堆石子数大于0，SG为0且每一堆石子数都为1

必败：其余为必败

5.Staircase Nim

阶梯博弈

每次可以从一个阶梯上拿掉任意数量石子放到下一层阶梯，不能操作的为输

SG函数为奇数阶梯上的石子的亦或和

如果移动偶数层的石子到奇数层，对手一定可以继续移动这些石子到偶数层，使得其SG不变

7.Take & Break

每次可以把一堆石子分成两堆甚至多堆不为0的石子，不能操作的为输

暴力计算SG

8.树上删边游戏

给定根节点，每次可以删掉一条边，不与根节点相连的部分删除

叶子节点SG为0，其他节点的SG函数为子树SG+1的亦或和

证明：

将子树SG+1看做石子数（我们可以定义没有节点的图的SG为-1），然后就变成了取石子游戏

9.无向图删边

规则同树上删边游戏

结论：把奇环缩成一个点加一条新边，把偶环缩成一个点，不影响SG，然后套用树上删边游戏

10.翻硬币游戏

n枚硬币排成一排，有的正面朝上，有的反面朝上。   
游戏者根据某些约束翻硬币（如：每次只能翻一或两枚，或者每次只能翻连续的几枚），但他所翻动的硬币中，最右边的必须是从正面翻到反面。   
谁不能翻谁输。

需要先开动脑筋把游戏转化为其他的取石子游戏之类的,然后用如下定理解决:   
局面的 SG 值等于局面中每个正面朝上的棋子单一存在时的 SG 值的异或和。