博弈论(一)

基本概念

ICG游戏：两个人参与的游戏，轮流做出对中间最有利的决策。

必败态：P-position（无法转移的状态），处于这种状态的人必输。

必胜态：N-position（可以转移到P的局面），处于这种状态的人必胜。

巴什博弈

一堆n个物品，两个人从中轮流取1~m个，最后不能继续取的人输。

同余定理：n=k*(m+1)+r；先取者拿走r个，那么后者无论拿走1~m个先者只要拿的数目之和为m+1那么先手就必赢。反之若n=k*(m+1)，那么先者无论怎么样都会输。

代码

```
if（n%(m+1)） return false;
else return true;
```

威佐夫博弈

有两堆各若干个物品，两个人轮流从任意一堆中取出至少1个或者同时从两堆中取出同样多的物品，规定每次至少取1个，至多不限，最后取光者胜利。

代码

```
double r=(sqrt(5.0)+1)/2;
int d=abs(a-b)*r;
if(d!=min(a,b)) return true;
else return false;
```

简单的思考

用数对(x,y)来表示两堆物品的数量，对于一个必败态(n,m)与(m,n)由于其本质相同。最简单的，显然(0,0)的情况为必败态。

假如局势为(1,2)，那么先手只有4中取法：

（1）先手取走第一堆的1个，那么后手取走第二堆的2个，后手胜。

（2）先手取走第二堆的2个，那么后手取走第一堆的1个，后手胜。

（3）先手取走第二堆的1个，那么后手在第一第二堆各取1个，后手胜。

（4）先手在第一第二堆各取1个，那么后手在第二堆取1个，后手胜。

可见这是先手的必败态。

定义先手必输的局势为奇异局势，前几个奇异局势为(0,0),(1,2),(3,5),(4,7),(6,10)...

性质（不会证）

1、x为前1...k个奇异局势中没有出现过的最小正整数，y=x+k

2、任何一个自然数都包含在一个且仅有一个奇异局势中。

3、任何操作都会将奇异局势变为非奇异局势

4、可以采用适合的方法将非奇异局势变为奇异局势。

根据**Beatty定理**（记得回去补证明），假设两堆石子为(x,y)（其中x<y）

那么先手必败，当且仅当$(y-x)*\frac{\sqrt{5}+1}{2}=x$

（黄金分割数1.618*两堆的差=最小值时为先手必败态）

尼姆博弈（Nim Game）

n堆物品，两个人轮流取，每次取某堆中不少于1个，最后取完者胜。

结论：将n堆物品数量全部异或后结果为0则必败，否则必胜。

尼姆博弈的结论是非常重要的，但证明过程较为繁琐。

代码

```
int res=0;
for(int i=1;i<=n;i++) res=res^arr[i];
if(res) return true;
else return false;
```

对于一般的博弈论问题，我们可以尝试通过SG函数转化为Nim Game

SG（Sprague-Grundy）函数

首先给出一种ICG博弈游戏模型，给定一个有向无环图和一个起始顶点上的一枚棋子，两名选手交替的将这枚棋子沿有向边进行移动，无法移动者判负。

将ICG问题进行转换：任何一个ICG都可以通过把每个局面看成一个顶点，对每个局面和它的子局面连一条有向边来抽象成这个"有向图游戏"。

于是我们可以通过将ICG问题转换为上述这个游戏，再通过寻找这个游戏的一遍解法来解决ICG问题。

首先定义mex(minimal excludant)运算，这是施加于一个集合的运算，表示最小的不属于这个集合的非负整数。例如mex{0,1,2,4}=3,mex{2,3,5}=0,mex{}=0;

对于一个给定的有向无环图，定义关于图的每个顶点的SG函数如下：

sg(x)=mex{sg(y)|y是x的后继}

步骤

一、找出必败态（SG值为0）

二、找到当前所有状态的前驱节点

三、根据定义计算节点SG值

重复上述步骤，直到整棵树建立完成

SG定理

游戏的和的SG函数值是它的所有子游戏的SG函数值的异或。

因此，当我们面对n个不同的游戏组成的游戏时，只需求出每个游戏的SG函数值把这些SG值全部看成Nim的石子堆，然后依照找Nim的必胜策略的方法来找这个游戏的必胜策略。

模板

```
//f[i]:可改变当前状态的方式，需要预处理
//S[i]:为i后继状态的集合
void getSG(int n){
    int i,j;
    memset(SG,0,sizeof(SG));
    for(i=1;i<=n;i++){
        memset(S,0,sizeof(S));
        for(j=0;f[j]<=i&&j<=N;j++) S[SG[i-f[j]]]=1;
        for(j=0;;j++) if(!S[j]){ //mex
            SG[i]=j;
            break;
        }
    }
```

```
}
```

Fibonacci again and again

```
#include<bits/stdc++.h>
#define inf 0x3f3f3f3f
#define int long long
using namespace std;
const int N=1e3+7;
int m,n,p,f[N],SG[N],S[N];

void get_SG(){
    memset(SG,0,sizeof(SG));
    for(int i=1;i<=1000;i++){
        memset(S,0,sizeof(S));
        for(int j=1;f[j]<=i&&j<=N;j++) S[SG[i-f[j]]]=1;
        for(int j=0;;j++) if(!S[j]){
            SG[i]=j;
            break;
        }
    }
}

signed main(){
    f[0]=1;f[1]=1;
    for(int i=2;i<=100;i++) f[i]=f[i-1]+f[i-2];
    get_SG();
    cin>>m>>n>>p;
    while(m!=0||n!=0||p!=0){
        int ans=SG[m]^SG[n]^SG[p];
        if(ans) puts("Fibo");
        else puts("Nacci");
        cin>>m>>n>>p;
    }
    return 0;
}
```

参考资料

https://www.bilibili.com/video/BV15b411W73J

https://www.cnblogs.com/zwfymqz/p/8469863.html

https://blog.csdn.net/qq_41311604/article/details/79980882

https://www.bilibili.com/video/BV1MT4y1L7BX

https://www.cnblogs.com/zwfymqz/p/8469840.html