有一堆个数为n(n>=2)的石子，游戏双方轮流取石子，规则如下：

1)先手不能在第一次把所有的石子取完，至少取1颗；

2)之后每次可以取的石子数至少为1，至多为对手刚取的石子数的2倍。

约定取走最后一个石子的人为赢家，求必败态。

结论：**当n为Fibonacci数的时候，必败。**

f[i]：1,2,3,5,8,13,21,34,55,89……

# 2516 取石子游戏

**Problem Description**

1堆石子有n个,两人轮流取.先取者第1次可以取任意多个，但不能全部取完.以后每次取的石子数不能超过上次取子数的2倍。取完者胜.先取者负输出"Second win".先取者胜输出"First win".

**Input**

输入有多组.每组第1行是2<=n<2^31. n=0退出.

**Output**

先取者负输出"Second win". 先取者胜输出"First win".   
参看Sample Output.

**Sample Input**

2

13

10000

0

**Sample Output**

Second win

Second win

First win

#include <stdio.h>

#include <string.h>

#include <algorithm>

using namespace std;

\_\_int64 a[50],len;

const \_\_int64 inf = 2147483648+10;

int main()

{

int i,j;

\_\_int64 n;

a[1] = 1;

a[2] = 1;

for(i = 3; i<=1000000; i++)

{

a[i] = a[i-1]+a[i-2];

if(a[i]>=inf)

break;

}

len = i;

while(~scanf("%I64d",&n),n)

{

int flag = 0;

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

{

if(a[i] == n)

{

flag = 1;

break;

}

if(a[i]>n)

break;

}

if(flag)

printf("Second win\n");

else

printf("First win\n");

}

return 0;

}

# a simple stone game

**Problem Description**

After he has learned how to play Nim game,Mike begins to try another stone game which seems much easier.  
The game goes like this:Two players start the game with a pile of n stones.They take stones from the pile in turn and every time they take at least one stone.The one who goes first can take at most n-1 stones for his first move .from then on a player can take at most k times as many stones as his opponent has taken last time.For example,if one player take m stones in his turn,then the other player can taken at most k \* m stones next time.The player who takes the last stone wins the game.suppose that those two players always take the best moves and never make mistakes,your job is to find out who will definitely win the game.

**Input**

The first line contains a integer t,indicating that there are t test cases following. (t<=20).Each test is a line consisting of two integer n and k.(2<=n<=10^8,1<=k<=10^5).

**Output**

For each test case,output one line starting with“Case N:”,N is the case number. And then,if the first player can ensure a winning ,print the minimum number of stones he should take in his first turn.Otherwise ,print “lose ”.Please note that there is a blank following the colon.

**Sample Input**

6

16 1

11 1

32 2

34 2

19 3

100000000 100000

**Sample Output**

Case 1: lose

Case 2: 1

Case 3: 3

Case 4：lose

Case 5: 4

Case 6: 912

#include<iostream>

#include<algorithm>

#include<cstdio>

#include<cstring>

#include<cmath>

using namespace std;

/\*

\* 构造数列a[], b[]

\* 其中a[i]表示构造出的数列

\* b[i]表示a[1~i]中若干个两两不相邻的数能表示出的最大数

\* 那么a[i + 1] = b[i] + 1

\* b[i + 1] = b[t] + a[i + 1], t = max{t|a[t]\*k < a[i + 1]}

\*/

//刚开始数组开小了...

#define maxn 2000000

int a[maxn];

int b[maxn];

int main()

{

int t;

int n, k;

scanf("%d", &t);

for(int cas = 1; cas <= t; cas++)

{

scanf("%d %d", &n, &k);

printf("Case %d: ", cas);

int now = 1;

a[1] = 1;

b[1] = 1;

int last = 1;

for(int i = 2; b[now] <= n; i++)

{

a[i] = b[i - 1] + 1;

b[i] = a[i];

while(a[last + 1]\*k < a[i])

last++;

if(a[last]\*k < a[i])

b[i] = b[last] + a[i];

now++;

}

for(int i = 1; i <= now; i++)

if(a[i] == n)//P点, 先手败

{

puts("lose");

goto nex;

}

//否则先手胜, 拿走最小的那一份就是分成不相邻的多个a[i]的和中最小的那个a[i]即可

while(a[now] != n)

{

if(n > a[now]) n -= a[now];

now--;

}

printf("%d\n", a[now]);

nex:;

}

return 0;

}