题意：有n个海盗，分m块金子，从第n个海盗开始，依次提出自己的分配方案，如果50%以上的人赞成，则方案通过，开始分金子，如果不通过，则把提出方案的扔到海里，下一个人继续。问编号为p的人最多分得多少金子？

思路：以下是cxlove的思路：如果只有两个人的话那么2号开始提出方案，这时候2号知道不管提什么，他自己肯定赞成，过半数，方案通过，那么2号肯定把所有的金子都给了自己。如果只有三个人的话：那么3号知道，如果自己死了，那么2号肯定能把所有金子拿下，对于1号来说没有半点好处。那么他就拿出1个金子贿赂1号，1号拿到1个金子，总比没有好，肯定赞成3号，剩下的金子3号拿下。如果只有四个人的话：那么4号知道，如果自己死了，那么1号拿到1个金子，2号什么都没有，3号拿下剩下的金子。那他就可以拿出1个金子贿赂2号，2号知道如果4号死了，自己将什么都没有，他肯定赞成4号，4号拿下剩下的全部金子。如此类推下去，貌似就是第一个决策的时候，与他奇偶性相同的人会被贿赂拿到1个金子，剩下的全归提出方案的人所有。但是会有一个问题便是，如果金子不够贿赂怎么办？

（1）如果n<=2\*m时候，前面与n相同奇偶性的得到1个金子，剩下的第n个人拿下。

（2）如果n==2\*m+1，第n个人拿出m个金子贿赂前面的m个人。自己不拿金子，这样刚好保证自己不死；

（3）如果n>2\*m+1，我们可以得到,编号为m\*2+2^i的人可以不死且拿不到一个金子。假设有500个海盗，只有100个金子，那么前面201个已经分析过了。对于202号来说，自己不能拿金币，而贿赂上一轮没有拿到金币的101人中的100人就够了。对于203号来说，需要102个人的支持，显然加上他自己，还需要101票，而金子不够贿赂，别人会反对，而达到杀人的目的。对于204号来说，他知道一旦自己死了，203号是必死，抓住这点，203必然支持他，因为203号宁可不要金币，也要保住性命，所以204号把100个金币分给之前的100个人，然后203和他自己的两票保证自己不死。对于205号来说，203和204是不会支持他的，因为一旦205死了，他们不仅可以保住性命，而且还可以看着205死掉。所以205是必死。那么206呢，虽然205必死，会支持他，但是还是缺一票，所以必死。对于207呢，205和206之前是必死，会支持他，但是加上自己以及100个贿赂名额，还是必死。对于208号，205，206，207是必死的，肯定会支持208成功，那么208刚好能凑齐104票，得以保命。

#include<bits/stdc++.h>

using namespace std;

typedef long long ll;

int fac[15]={2,4,8,16,32,64,128,256,512,1024,2048,4096,8192,16384,32768};

int main()

{

int T;

int n,m,p;

cin>>T;

while(T--)

{

cin>>n>>m>>p;

//金币够贿赂的情况

if(n<=2\*m)

{

if(p==n)

cout<<m-((n-1)/2)<<endl;

else

if((p&1)==(n&1))//n与m奇偶性相同

puts("1");

else puts("0");

continue;

}

//这时候的不同在于决策者不能拿金币

if(n==2\*m+1)

{

if((p&1)==1 && p<2\*m)

puts("1");

else puts("0");

continue;

}

int t=n-2\*m;

//这是刚好保命的情况，对于决策者来说，肯定没有金币

//对于其它人来说，要么肯定没有金币，要么可能没有金币，不确定性

for(int i=0;i<14;i++)

if(t==fac[i])

{

puts("0");

continue;

}

for(int i=1;i<14;i++)

//决策者必死

if(t<fac[i])

{

if(p>2\*m+fac[i-1] && p<2\*m+fac[i])

puts("Thrown");

else puts("0");

break;

}

}

return 0;

}