//Floyed’s Cycle Finding Algorithm

//Tested for Uva 350 Pseudo Random Numbers

#include <bits/stdc++.h>

using namespace std;

typedef long long int ll;

//implementation of floyd cycle finding algorithm

ll Z,I,M;

ll move\_function(ll L){

ll value = ((Z%M \* L%M)%M + I % M)%M;

return value;

}

//----m----- -----k----

// A->B->C->D->E->F->G ||

// | | ||

// I<-H<-P<-Q ||

// matched here||

ll cycle\_finding\_algorithm(ll start){

//tortoise will move single move

//hare will move double move

ll hare\_move=start,tortoise\_move=start;

ll save;

int counter=0;

while(true){

counter++; //(m+k) banai

hare\_move=move\_function(hare\_move);

hare\_move=move\_function(hare\_move); //double move

tortoise\_move=move\_function(tortoise\_move); //single move

if(hare\_move==tortoise\_move){

//matched now at kth node from start

//now need to move m from here and start

save=hare\_move;

break;

}

}

if(save==start){

//printf("yo %d\n",counter);

return counter;

}

//now single move and one node will be matched at mth move

ll second\_tortoise=start;

int m=0;

while(true){

m++;

second\_tortoise =move\_function(second\_tortoise);

tortoise\_move = move\_function(tortoise\_move);

if(second\_tortoise == tortoise\_move) {

break;

}

}

int k=0;

while(true){

if(second\_tortoise == save) {

break;

}

second\_tortoise =move\_function(second\_tortoise);

k++;

}

return (k+1); //counter=m+k k cycle's length, m distance from start

}

int main(int argc,char \*\*argv){

ll L;

int t=1;

while(scanf("%lld %lld %lld %lld",&Z,&I,&M,&L) == 4) {

if(Z == 0 && I == 0 && M == 0 && L == 0) break;

ll answer = cycle\_finding\_algorithm(L);

printf("Case %d: %lld\n",t,answer);

t++;

}

return 0;

}