

# BSGS HASHMAP

// B^L == N (mod P)

// Code by KSkun, 2018/4

#include <cstdio>

#include <cmath>

#include <cstring>

#include <algorithm>

typedef long long LL;

inline LL fpow(LL n, LL k, LL p) {

LL res = 1; n %= p;

while(k) {

if(k & 1) res = res \* n % p;

n = n \* n % p;

k >>= 1;

}

return res;

}

const int MO = 611977, MAXN = 1000005;

struct HashMap {

int head[MO + 5], key[MAXN], value[MAXN], nxt[MAXN], tot;

inline void clear() {

tot = 0;

memset(head, -1, sizeof(head));

}

HashMap() {

clear();

}

inline void insert(int k, int v) {

int idx = k % MO;

for(int i = head[idx]; ~i; i = nxt[i]) {

if(key[i] == k) {

value[i] = v;

return;

}

}

key[tot] = k; value[tot] = v; nxt[tot] = head[idx]; head[idx] = tot++;

}

inline int operator[](const int &k) const {

int idx = k % MO;

for(int i = head[idx]; ~i; i = nxt[i]) {

if(key[i] == k) return value[i];

}

return -1;

}

} x;

inline LL bsgs(LL a, LL b, LL p) {

a %= p; b %= p;

if(a == 0) return b == 0 ? 1 : -1;

if(b == 1) return 0;

LL m = ceil(sqrt(p - 1)), inv = fpow(a, p - m - 1, p);

x.clear();

x.insert(1, 0);

for(LL i = 1, e = 1; i < m; i++) {

e = e \* a % p;

if(x[e] == -1) x.insert(e, i);

}

for(LL i = 0; i < m; i++) {

if(x[b] != -1) {

LL res = x[b];

return i \* m + res;

}

b = b \* inv % p;

}

return -1;

}

LL p, b, n;

int main() {

while(scanf("%lld%lld%lld", &p, &b, &n) != EOF) {

LL res = bsgs(b, n, p);

if(res != -1) printf("%lld\n", res); else puts("no solution");

}

return 0;

}

# BSGS STLMAP

// B^L == N (mod P)

// Code by KSkun, 2018/4

#include <cstdio>

#include <cmath>

#include <map>

typedef long long LL;

inline LL fpow(LL n, LL k, LL p) {

LL res = 1; n %= p;

while(k) {

if(k & 1) res = res \* n % p;

n = n \* n % p;

k >>= 1;

}

return res;

}

std::map<LL, LL> x;

inline LL bsgs(LL a, LL b, LL p) {

a %= p; b %= p;

if(a == 0) return b == 0 ? 1 : -1;

if(b == 1) return 0;

LL m = ceil(sqrt(p - 1)), inv = fpow(a, p - m - 1, p);

x.clear();

x[1] = m; // use m instead of 0

for(LL i = 1, e = 1; i < m; i++) {

e = e \* a % p;

if(!x[e]) x[e] = i;

}

for(LL i = 0; i < m; i++) {

if(x[b]) {

LL res = x[b];

return i \* m + (res == m ? 0 : res);

}

b = b \* inv % p;

}

return -1;

}

LL p, b, n;

int main() {

while(scanf("%lld%lld%lld", &p, &b, &n) != EOF) {

LL res = bsgs(b, n, p);

if(res != -1) printf("%lld\n", res); else puts("no solution");

}

return 0;

}

# exBSGS HASHMAP

// B^L == N (mod P)

// Code by KSkun, 2018/4

#include <cstdio>

#include <cmath>

#include <cstring>

#include <algorithm>

typedef long long LL;

inline char fgc() {

static char buf[100000], \*p1 = buf, \*p2 = buf;

return p1 == p2 && (p2 = (p1 = buf) + fread(buf, 1, 100000, stdin), p1 == p2) ? EOF

: \*p1++;

}

inline LL readint() {

register LL res = 0, neg = 1;

char c = fgc();

while(c < '0' || c > '9') {

if(c == '-') neg = -1;

c = fgc();

}

while(c >= '0' && c <= '9') {

res = res \* 10 + c - '0';

c = fgc();

}

return res \* neg;

}

inline LL fpow(LL n, LL k, LL p) {

LL res = 1; n %= p;

while(k) {

if(k & 1) res = res \* n % p;

n = n \* n % p;

k >>= 1;

}

return res;

}

inline LL exgcd(LL a, LL b, LL &x, LL &y) {

if(!b) {

x = 1; y = 0;

return a;

}

LL res = exgcd(b, a % b, x, y);

LL t = x; x = y; y = t - a / b \* y;

return res;

}

const int MO = 611977, MAXN = 1000005;

struct HashMap {

LL head[MO + 5], key[MAXN], value[MAXN], nxt[MAXN], tot;

inline void clear() {

tot = 0;

memset(head, -1, sizeof(head));

}

HashMap() {

clear();

}

inline void insert(LL k, LL v) {

int idx = k % MO;

for(int i = head[idx]; ~i; i = nxt[i]) {

if(key[i] == k) {

value[i] = std::min(value[i], v);

return;

}

}

key[tot] = k; value[tot] = v; nxt[tot] = head[idx]; head[idx] = tot++;

}

inline LL operator[](const LL &k) const {

int idx = k % MO;

for(int i = head[idx]; ~i; i = nxt[i]) {

if(key[i] == k) return value[i];

}

return -1;

}

} x;

inline LL bsgs(LL a, LL b, LL p) {

a %= p; b %= p;

if(a == 0) return b == 0 ? 1 : -1;

if(b == 1) return 0;

LL m = ceil(sqrt(p - 1)), inv, y;

exgcd(fpow(a, m, p), p, inv, y); inv = (inv % p + p) % p;

x.clear();

x.insert(1, 0);

for(LL i = 1, e = 1; i < m; i++) {

e = e \* a % p;

if(x[e] == -1) x.insert(e, i);

}

for(LL i = 0; i < m; i++) {

if(x[b] != -1) {

LL res = x[b];

return i \* m + res;

}

b = b \* inv % p;

}

return -1;

}

inline LL gcd(LL a, LL b) {

if(!b) return a;

return gcd(b, a % b);

}

inline LL exbsgs(LL a, LL b, LL p) {

if(b == 1) return 0;

LL tb = b, tmp = 1, k = 0;

for(int g = gcd(a, p); g != 1; g = gcd(a, p)) {

if(tb % g) return -1;

tb /= g; p /= g; tmp = tmp \* a / g % p;

k++;

if(tmp == tb) return k;

}

return bsgs(a, b, p);

}

LL a, b, p;

int main() {

for(;;) {

a = readint(); p = readint(); b = readint();

if(!a && !b && !p) break;

LL res = exbsgs(a, b, p);

if(res != -1) printf("%lld\n", res); else puts("No Solution");

}

return 0;

}

# 高斯消元解异或线性方程组

#include <iostream>

#include <algorithm>

using namespace std;

const int N = 110;

int n;

int a[N][N];

int gauss()

{

int c,r;

for(c=0,r=0;c<n;c++)

{

// 找主元

int t=-1;

for(int i=r;i<n;i++)

if(a[i][c])

{

t=i;

break;

}

if(t==-1) continue;

// 交换主元行

for(int j=c;j<=n;j++) swap(a[r][j],a[t][j]);

// 左下角消

for(int i=r+1;i<n;i++)

if(a[i][c])//漏了

for(int j=n;j>=c;j--)//漏了

a[i][j] ^= a[r][j];

r++;

}

// 判断

if(r<n)

{

for(int i=r;i<n;i++)//i=r

if(a[i][n])

return 2;

return 1;

}

// 消右上角

for(int i=n-1;i>=0;i--)

for(int j=i+1;j<n;j++)

//如果是0 就不用下面的a[j][j] 来^a[i][j]了

//如果不是0 才需要用第j行第j列a[j][j]来^第i行第j列a[i][j]

//进而进行整行row[i]^row[j] 间接导致 a[i][n]^a[j][n]

if(a[i][j])

a[i][n]^=a[j][n];

return 0;

}

int main()

{

cin >> n;

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

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

cin >> a[i][j];

int t = gauss();

if(t==0)

{

for(int i=0;i<n;i++) cout << a[i][n] << endl;

}

else if(t==1) puts("Multiple sets of solutions");

else puts("No solution");

return 0;

}

# 模意义下的高斯消元

#include<cstdio>

#define maxn 110

#define r register

using namespace std;

typedef long long ll;

int n,p,maxi;

ll tmp,ans[maxn],a[maxn][maxn];

int read()

{

r char ch=getchar();r int in=0;

while(ch>'9'||ch<'0') ch=getchar();

while(ch>='0'&&ch<='9') in=(in<<3)+(in<<1)+ch-'0',ch=getchar();

return in;

}

ll ksm(r ll x,r int y)

{

if(!y) return 1;

r ll ret=ksm(x,y>>1);

if(y&1) return ret\*ret%p\*x%p;

return ret\*ret%p;

}

int main()

{

n=read(),p=read();

for(r int i=1;i<=n;i++)

for(r int j=1;j<=n+1;j++)

a[i][j]=read();

for(r int i=1;i<=n;i++)

{

if(!a[i][i])//主元不能为0

{

maxi=0;

for(r int j=i+1;j<=n&&!maxi;j++)

if(a[j][i]) maxi=j;

if(!maxi) continue;//如果一整列都为0，不需要消元

for(r int j=i;j<=n+1;j++)

tmp=a[maxi][j],a[maxi][j]=a[i][j],a[i][j]=tmp;

}

for(r int j=i+1;j<=n;j++)

{

tmp=a[j][i];

if(!tmp) continue;//已经为0，不需要消元

for(r int k=i;k<=n+1;k++)

a[j][k]=((a[j][k]\*a[i][i]-a[i][k]\*tmp)%p+p)%p;

}

}

for(r int i=n;i;i--)

{

for(r int j=i+1;j<=n;j++)

a[i][n+1]=((a[i][n+1]-ans[j]\*a[i][j])%p+p)%p;

ans[i]=a[i][n+1]\*ksm(a[i][i],p-2)%p;

}

for(r int i=1;i<=n;i++) printf("%lld ",ans[i]);

return 0;

}