// 矩阵快速幂2

int maxn,n,mod;

struct maxtrix {

int m[100][100];

}I, A, B, C, T;

void init(maxtrix A) { // 初始化为单位矩阵

memset(A.m, 0, sizeof(A.m));

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

A.m[i][i] = 1;

}

maxtrix add(maxtrix A, maxtrix B) { /// 矩阵加法

maxtrix ans;

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

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

ans.m[i][j] = A.m[i][j] + B.m[i][j],

ans.m[i][j] %= mod;

return ans;

}

maxtrix mul(maxtrix A, maxtrix B) { /// 矩阵乘法

maxtrix ans;

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

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

{

ans.m[i][j] = 0;

for(int k = 0; k < maxn; ++k)

ans.m[i][j] += A.m[i][k] \* B.m[k][j];

ans.m[i][j] %= mod;

}

return ans;

}

maxtrix pow(maxtrix A, int n) { /// 矩阵快速幂 A 的 n 次方

maxtrix ans;

init(ans);

while( n ) {

if( n&1 )

ans = mul(ans, A);

n >>= 1;

A = mul (A, A);

}

return ans;

}

maxtrix sum(maxtrix A, int n) {/// 矩阵幂和 A 自己加 n 次

int m;

maxtrix ans, pre;

if( n == 1)

return A;

m = n/2;

pre = sum(A, m); // [1,n/2]

ans = add(pre, mul(pre, pow(A, n))); // ans=[1,n/2]+a^(n/2)\*[1,n/2]

if( n&1 )

ans = add( ans, pow(A, n) );

return ans;

}

void output(maxtrix A) {

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

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

printf("%d%c", A.m[i][j], j == maxn - 1 ? '\n' : ' ');

}

int main() {

maxtrix ans;

scanf("%d %d %d",&maxn, &n, &mod);

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

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

scanf("%d", &A.m[i][j] ), A.m[i][j] %= mod;

ans = sum(A, n);

output(ans);

return 0;

}