

model_gauss

```
1  #include <bits/stdc++.h>
2  #define int long long
3  #define endl '\n'
4  #define LL __int128
5  using namespace std;
6  int qpow(int a, int b, int p) {int ret = 1; for(a %= p; b; b >>= 1, a = a *
a % p) if(b & 1) ret = ret * a % p; return ret; }
7  int qpow(int a,int b) {int ret = 1; for(; b; b >>= 1, a *= a) if(b & 1) ret
*= a; return ret; }
8  int gcd(int x,int y) {return y ? gcd(y, x % y) : x; }
9  pair<int,int> exgcd(int a,int b) { if(!b) return {1, 0}; pair<int,int> ret
= exgcd(b, a % b); return {ret.second, ret.first - a / b * ret.second }; }
10 namespace gauss {
11     const int N = 5 + 6e2;
12     const double eps = 1e-6;
13     int gauss(double a[][N], int n, int m) { //n = row, m = col
14         int now_r = 0;
15         for (int i = 0; i < m - 1 && now_r < n; i++, now_r++) {
16             int mx = now_r;
17             for (int j = now_r; j < n; j++) {
18                 if (fabs(a[mx][i]) < fabs(a[j][i])) {
19                     mx = j;
20                 }
21             }
22             if (mx != now_r) {
23                 for (int j = i; j < m; j++) {
24                     swap(a[now_r][j], a[mx][j]);
25                 }
26             }
27             if (fabs(a[now_r][i]) < eps) {
28                 now_r--;
29                 continue;
30             }
31             for (int j = now_r + 1; j < n; j++) {
32                 if (fabs(a[j][i]) < eps) {
33                     continue;
34                 }
35                 double v = a[j][i] / a[now_r][i];
36                 for (int k = i; k < m; k++) {
37                     a[j][k] -= a[now_r][k] * v;
38                 }
39             }
40         }
41         for (int i = now_r; i < n; i++) {
42             if (fabs(a[i][m - 1]) > eps) {
43                 return -1;
44             }
45         }
46         if (now_r < n) {
47             return n - now_r;
48         }
49         for (int i = now_r - 1; i >= 0; i--) {
```

```

50         a[i][m - 1] /= a[i][i];
51         for (int k = 0; k < i; k++) {
52             a[k][m - 1] -= a[k][i] * a[i][m - 1];
53         }
54     }
55     return 0;
56 }
57 int gauss(int a[][N], int n, int m) {
58     int now_r = 0;
59     for (int i = 0; i < m - 1 && now_r < n; i++, now_r++) {
60         int mx = now_r;
61         for (int j = now_r; j < n; j++) {
62             if (a[mx][i] < a[j][i]) {
63                 mx = j;
64             }
65         }
66         if (mx != now_r) {
67             for (int j = i; j < m; j++) {
68                 swap(a[now_r][j], a[mx][j]);
69             }
70         }
71         if (!a[now_r][i]) {
72             now_r--;
73             continue;
74         }
75         for (int j = now_r + 1; j < n; j++) {
76             if (!a[j][i]) {
77                 continue;
78             }
79             int _lcm = a[j][i] / __gcd(a[j][i], a[now_r][i]) * a[now_r]
[i];
80             int p = _lcm / a[j][i], q = _lcm / a[now_r][i];
81             for (int k = i; k < m; k++) {
82                 a[j][k] = a[j][k] * p - a[now_r][k] * q;
83             }
84         }
85     }
86     for (int i = now_r; i < n; i++) {
87         if (a[i][m - 1]) return -1;
88     }
89     if (now_r < n) {
90         return n - now_r;
91     }
92     for (int i = now_r - 1; i >= 0; i--) {
93         if (a[i][m - 1] % a[i][i]) return -2; //is float ans
94         a[i][m - 1] /= a[i][i];
95         for (int k = 0; k < i; k++) {
96             a[k][m - 1] -= a[k][i] * a[i][m - 1];
97         }
98     }
99     return 0;
100 }
101 int mod = 998244353;
102 int det(int a[][N], int n, int m) {
103     for (int i = 0; i < n; i++) {
104         for (int j = 0; j < m; j++) {
105             a[i][j] = (a[i][j] % mod + mod) % mod;
106         }

```

```

107     }
108     int now_r = 0;
109     int sum = 1;
110     for (int i = 0; i < m && now_r < n; i++, now_r++) {
111         int mx = now_r;
112         for (int j = now_r + 1; j < n; j++) {
113             if (a[mx][i] < a[j][i]) {
114                 mx = j;
115             }
116         }
117         if (mx != now_r) {
118             //swap(a[mx], a[now_r]);
119             for (int j = i; j < m; j++) {
120                 swap(a[mx][j], a[now_r][j]);
121             }
122             sum *= -1;
123         }
124         if (!a[now_r][i]) {
125             now_r--;
126             continue;
127         }
128         for (int j = now_r + 1; j < n; j++) {
129             if (a[j][i] > a[now_r][i]) {
130                 //swap(a[j], a[now_r]);
131                 for (int k = i; k < m; k++) {
132                     swap(a[now_r][k], a[j][k]);
133                 }
134                 sum *= -1;
135             }
136             while (a[j][i]) {
137                 int t = a[now_r][i] / a[j][i];
138                 for (int k = i; k < m; k++) {
139                     a[now_r][k] -= a[j][k] * t % mod;
140                     a[now_r][k] = (a[now_r][k] + mod) % mod;
141                 }
142                 //swap(a[now_r], a[j]);
143                 for (int k = i; k < m; k++) {
144                     swap(a[now_r][k], a[j][k]);
145                 }
146                 sum *= -1;
147             }
148         }
149     }
150     for (int i = 0; i < n; i++) {
151         sum = sum * a[i][i] % mod;
152     }
153     return (sum % mod + mod) % mod;
154 }
155
156 const int mod = 998244353;
157
158 const int N = 5 + 6e2;
159 int a[N][N];
160
161 signed main() {
162     ios :: sync_with_stdio(false), cin.tie(0), cout.tie(0);
163     int n, mod; cin >> n >> mod;
164     for (int i = 0; i < n; i++) {

```

```
165         for (int j = 0; j < n; j++) {
166             cin >> a[i][j];
167         }
168     }
169     gauss :: mod = mod;
170     cout << gauss :: det(a, n, n) << endl;
171 }
```