model_gauss

```cpp
#include <bits/stdc++.h>
#define int long long
#define endl '\n'
#define LL __int128
using namespace std;
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; }
int qpow(int a,int b) {int ret = 1; for(; b; b >>= 1, a *= a) if(b & 1) ret *= a; return ret; }
int gcd(int x,int y) {return y ? gcd(y, x % y) : x; }
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 }; }
namespace gauss {
    const int N = 5 + 6e2;
    const double eps = 1e-6;
    int gauss(double a[][N], int n, int m) {    //n = row, m = col
        int now_r = 0;
        for (int i = 0; i < m - 1 && now_r < n; i++, now_r++) {
            int mx = now_r;
            for (int j = now_r; j < n; j++) {
                if (fabs(a[mx][i]) < fabs(a[j][i])) {
                    mx = j;
                }
            }
            if (mx != now_r) {
                for (int j = i; j < m; j++) {
                    swap(a[now_r][j], a[mx][j]);
                }
            }
            if (fabs(a[now_r][i]) < eps) {
                now_r--;
                continue;
            }
            for (int j = now_r + 1; j < n; j++) {
                if (fabs(a[j][i]) < eps) {
                    continue;
                }
                double v = a[j][i] / a[now_r][i];
                for (int k = i; k < m; k++) {
                    a[j][k] -= a[now_r][k] * v;
                }
            }
        }
        for (int i = now_r; i < n; i++) {
            if (fabs(a[i][m - 1]) > eps) {
                return -1;
            }
        }
        if (now_r < n) {
            return n - now_r;
        }
        for (int i = now_r - 1; i >= 0; i--) {
            a[i][m - 1] /= a[i][i];
            for (int k = 0; k < i; k++) {
                a[k][m - 1] -= a[k][i] * a[i][m - 1];
            }
        }
        return 0;
    }
    int gauss(int a[][N], int n, int m) {
        int now_r = 0;
        for (int i = 0; i < m - 1 && now_r < n; i++, now_r++) {
            int mx = now_r;
            for (int j = now_r; j < n; j++) {
                if (a[mx][i] < a[j][i]) {
                    mx = j;
                }
            }
            if (mx != now_r) {
                for (int j = i; j < m; j++) {
                    swap(a[now_r][j], a[mx][j]);
                }
            }
            if (!a[now_r][i]) {
                now_r--;
                continue;
            }
            for (int j = now_r + 1; j < n; j++) {
                if (!a[j][i]) {
                    continue;
                }
                int _lcm = a[j][i] / __gcd(a[j][i], a[now_r][i]) * a[now_r][i];
                int p = _lcm / a[j][i], q = _lcm / a[now_r][i];
                for (int k = i; k < m; k++) {
                    a[j][k] = a[j][k] * p - a[now_r][k] * q;
                }
            }
        }
        for (int i = now_r; i < n; i++) {
            if (a[i][m - 1]) return -1;
        }
        if (now_r < n) {
            return n - now_r;
        }
        for (int i = now_r - 1; i >= 0; i--) {
            if (a[i][m - 1] % a[i][i]) return -2;   //is float ans
            a[i][m - 1] /= a[i][i];
            for (int k = 0; k < i; k++) {
                a[k][m - 1] -= a[k][i] * a[i][m - 1];
            }
        }
        return 0;
    }
    int mod = 998244353;
    int det(int a[][N], int n, int m) {
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < m; j++) {
                a[i][j] = (a[i][j] % mod + mod) % mod;
            }
        }
        int now_r = 0;
        int sum = 1;
        for (int i = 0; i < m && now_r < n; i++, now_r++) {
            int mx = now_r;
            for (int j = now_r + 1; j < n; j++) {
                if (a[mx][i] < a[j][i]) {
                    mx = j;
                }
            }
            if (mx != now_r) {
                //swap(a[mx], a[now_r]);
                for (int j = i; j < m; j++) {
                    swap(a[mx][j], a[now_r][j]);
                }
                sum *= -1;
            }
            if (!a[now_r][i]) {
                now_r--;
                continue;
            }
            for (int j = now_r + 1; j < n; j++) {
                if (a[j][i] > a[now_r][i]) {
                    //swap(a[j], a[now_r]);
                    for (int k = i; k < m; k++) {
                        swap(a[now_r][k], a[j][k]);
                    }
                    sum *= -1;
                }
                while (a[j][i]) {
                    int t = a[now_r][i] / a[j][i];
                    for (int k = i; k < m; k++) {
                        a[now_r][k] -= a[j][k] * t % mod;
                        a[now_r][k] = (a[now_r][k] + mod) % mod;
                    }
                    //swap(a[now_r], a[j]);
                    for (int k = i; k < m; k++) {
                        swap(a[now_r][k], a[j][k]);
                    }
                    sum *= -1;
                }
            }
        }
        for (int i = 0; i < n; i++) {
            sum = sum * a[i][i] % mod;
        }
        return (sum % mod + mod) % mod;
    }
}
const int mod = 998244353;

const int N = 5 + 6e2;
int a[N][N];

signed main() {
    ios :: sync_with_stdio(false), cin.tie(0), cout.tie(0);
    int n, mod; cin >> n >> mod;
    for (int i = 0; i < n; i++) {
```

```cpp
        for (int j = 0; j < n; j++) {
            cin >> a[i][j];
        }
    }
    gauss :: mod = mod;
    cout << gauss :: det(a, n, n) << endl;
}
```