gaussian elimination

```cpp
// ADD
for (int i = 0; i < n; i += 1) {
    for (int j = 0; j <= n; j += 1) {
        std::cin >> f[i][j];
    }
}
constexpr double eps = 1e-9;
int r = 0;
std::vector<int>loc(n, -1);
for (int c = 0; c < n && r < n; c += 1) {
    int sel = r;
    for (int j = r; j < n; j += 1) {
        if (std::abs(f[j][c]) > std::abs(f[sel][c])) {
            sel = j;
        }
    }
    if (std::abs(f[sel][c]) < eps) {
        continue;
    }
    std::swap(f[sel], f[r]);
    loc[c] = r;
    for (int i = 0; i < n; i += 1) {
        if (i != r) {
            double d = f[i][c] / f[r][c];
            for (int j = c; j <= n; j += 1) {
                f[i][j] -= d * f[r][j];
            }
        }
    }
    r += 1;
}
for (int i = 0; i < n; i += 1) {
    double s = 0;
    for (int j = 0; j < n; j += 1) {
        if (loc[j] == -1) {
            continue;
        }
        s += f[loc[j]].back() / f[loc[j]][j] * f[i][j];
    }
    if (std::abs(s - f[i].back()) > eps) {
        std::cout << "-1\n";
        std::exit(0);
    }
}
if (std::count(loc.begin(), loc.end(), -1)) {
    std::cout << "0\n";
    std::exit(0);
}
```

```cpp
for (int i = 0; i < n; i += 1) {
    std::cout << std::fixed << std::setprecision(2) << 'x' << i + 1 << '='
<< (f[loc[i]].back() / f[loc[i]][i]) << '\n';
}
// XOR
for (int i = 0; i < n; i += 1) {
    for (int j = 0; j < m; j += 1) {
        int x;
        std::cin >> x;
        if (x) {
            f[i].set(j);
        }
    }
}
for (int i = 0; i < n; i += 1) {
    for (int j = 0; j < p; j += 1) {
        int x;
        std::cin >> x;
        if (x) {
            f[i].set(m + j);
        }
    }
}
std::vector<int>loc(m, -1);
int r = 0;
for (int c = 0; c < m && r < n; c += 1) {
    int sel = r;
    while (sel < n && !f[sel].test(c)) {
        sel += 1;
    }
    if (sel == n) {
        // 这一行没有主元，有无穷多解
        continue;
    }
    std::swap(f[sel], f[r]);
    loc[c] = r;
    for (int i = 0; i < n; i += 1) {
        if (i != r && f[i].test(c)) {
            f[i] ^= f[r];
        }
    }
    r += 1;
}
for (int i = r; i < n; i += 1) {
    bool find = false;
    for (int j = 0; j < m; j += 1) {
        if (f[i].test(j)) {
            find = true;
        }
    }
    if (!find) {
        for (int j = m; j < m + p; j += 1) {
                if (f[i].test(j)) {
                    std::exit(0);
                }
            }
        }
    }
    for (int i = 0; i < m; i += 1) {
        for (int j = 0; j < p; j += 1) {
            std::cout << (loc[i] != -1 ? f[loc[i]].test(m + j) : 0) << ' ';
        }
        std::cout << '\n';
    }
    vector<long long> gauss_mod(vector<vector<long long>>& a, int n, int p) {
        int m = n + 1;
        vector<int> where(n, -1);
        for (int col = 0, row = 0; col < n && row < n; ++col) {
            int sel = row;
            while (sel < n && a[sel][col] % p == 0) sel++;
            if (sel == n) continue;
            swap(a[sel], a[row]);
            where[col] = row;
            long long inv = modinv(a[row][col], p);
            for (int j = col; j < m; ++j)
                a[row][j] = a[row][j] * inv % p;
            for (int i = 0; i < n; ++i) if (i != row) {
                long long factor = a[i][col];
                for (int j = col; j < m; ++j)
                    a[i][j] = (a[i][j] - factor * a[row][j]) % p;
            }
            ++row;
        }
        vector<long long> x(n, 0);
        for (int i = 0; i < n; ++i)
            if (where[i] != -1)
                x[i] = (a[where[i]][n] % p + p) % p;
        return x;
    }
```