```cpp
#include <bits/stdc++.h>
using namespace std;

using vi = vector<int>; using vvi = vector<vi>; using vvvi = vector<vvi>;
using ll = long long int;
using vll = vector<ll>; using vvll = vector<vll>; using vvvll = vector<vvll>;
using vd = vector<double>; using vvd = vector<vd>; using vvvd = vector<vvd>;
using P = pair<int, int>;
using Pll = pair<ll, ll>;
using cdouble = complex<double>;

const double eps = 1e-7;
#define Loop(i, n) for(int i = 0; i < int(n); i++)
#define Loopll(i, n) for(ll i = 0; i < ll(n); i++)
#define Loop1(i, n) for(int i = 1; i <= int(n); i++)
#define Loopll1(i, n) for(ll i = 1; i <= ll(n); i++)
#define Loopr(i, n) for(int i = int(n) - 1; i >= 0; i--)
#define Looprll(i, n) for(ll i = ll(n) - 1; i >= 0; i--)
#define Loopr1(i, n) for(int i = int(n); i >= 1; i--)
#define Looprll1(i, n) for(ll i = ll(n); i >= 1; i--)
#define Foreach(buf, container) for(auto buf : container)
#define Loopdiag(i, j, h, w, sum) for(int i = ((sum) >= (h) ? (h) - 1 : (sum)), j = (sum) - i; i >= 0 && j < (w); i--, j++)
#define Loopdiagr(i, j, h, w, sum) for(int j = ((sum) >= (w) ? (w) - 1 : (sum)), i = (sum) - j; j >= 0 && i < (h); j--, i++)
#define Loopdiagsym(i, j, h, w, gap) for (int i = ((gap) >= 0 ? (gap) : 0), j = i - (gap); i < (h) && j < (w); i++, j++)
#define Loopdiagsymr(i, j, h, w, gap) for (int i = ((gap) > (h) - (w) - 1 ? (h) - 1 : (w) - 1 + (gap)), j = i - (gap); i >= 0 && j >= 0; i--, j--)
#define Loopitr(itr, container) for(auto itr = container.begin(); itr != container.end(); itr++)
#define printv(vector) Loop(ex_i, vector.size()) { cout << vector[ex_i] << " "; } cout << endl;
#define printmx(matrix) Loop(ex_i, matrix.size()) { Loop(ex_j, matrix[ex_i].size()) { cout << matrix[ex_i][ex_j] << " "; } cout << endl; }
#define quickio() ios::sync_with_stdio(false); cin.tie(0);
#define bitmanip(m,val) static_cast<bitset<(int)m>>(val)
#define Comp(type_t) bool operator<(const type_t &another) const
#define fst first
#define snd second
#define INF INFINITY
bool feq(double x, double y) { return abs(x - y) <= eps; }
bool inrange(ll x, ll t) { return x >= 0 && x < t; }
bool inrange(vll xs, ll t) { Foreach(x, xs) if (!(x >= 0 && x < t)) return false; return true; }
int ceillog2(ll x) { int ret = 0; x--; while (x > 0) { ret++; x >>= 1; } return ret; }
ll rndf(double x) { return (ll)(x + (x >= 0 ? 0.5 : -0.5)); }
ll floorsqrt(ll x) { ll m = (ll)sqrt((double)x); return m + (m * m <= x ? 0 : -1); }
ll ceilsqrt(ll x) { ll m = (ll)sqrt((double)x); return m + (x <= m * m ? 0 : 1); }
ll rnddiv(ll a, ll b) { return (a / b + (a % b * 2 >= b ? 1 : 0)); }
ll ceildiv(ll a, ll b) { return (a / b + (a % b == 0 ? 0 : 1)); }
ll gcd(ll m, ll n) { if (n == 0) return m; else return gcd(n, m % n); }
ll lcm(ll m, ll n) { return m * n / gcd(m, n); }

/*******************************************************/


```