**/\* LCA \*/**

**const** **int** MAXN = 100000, MAXLOG = 18;

**int** n, m, u, v, q,

A[MAXLOG][MAXN], //A[i][v] = ancestor of node v to distance 2 ^ i

P[MAXN], //P[v] = parent of node v

L[MAXN]; //L[v] = level of node v in the tree

**vector**<**int**> g[MAXN];

**void** dfs (**int** u, **int** l, **int** p) {

L[u] = l; P[u] = p;

**for** (**int** i = 0; i < g[u].**size**(); ++i)

if (g[u][i] != p) dfs(g[u][i], l + 1, u);

}

**void** build () {

dfs(0, 0, -1);

memset(A, -1, **sizeof** A);

**for** (**int** i = 0; i < n; ++i) A[0][i] = P[i];

**for** (**int** i = 1; i < MAXLOG; ++i)

**for** (**int** j = 0; j < n; ++j)

**if** (A[i - 1][j] != -1)

A[i][j] = A[i - 1][A[i - 1][j]];

}

**int** query (**int** u, **int** v) {

**if** (L[u] < L[v]) **swap**(u, v);

**int** d = L[u] - L[v];

**for** (**int** i = 0; (1 << i) <= d; ++i)

if (d & (1 << i)) u = A[i][u];

**for** (**int** log = MAXLOG - 1; log >= 0; --log)

**if** (A[log][u] != -1 && A[log][u] != A[log][v]) {

u = A[log][u]; v = A[log][v];

}

**return** u == v ? u : P[u];

}

**int** main() {

**cin** >> n >> m;

**for** (**int** i = 0; i < m; ++i) {

**cin** >> u >> v;

--u, --v;

g[u].**push\_back**(v); g[v].**push\_back**(u);

}

build();

**for** (**int** i = 0; ; ++i) {

**cin** >> u >> v;

--u, --v;

**cout** << query(u, v) + 1 << **endl**;

}

**return** 0;

}