Contents

1. DataStructure

1.1. 2DBIT.cpp

```cpp


using namespace std;
#define LL long long
#define pii pair<int, int>
#define N 1005
#define F first
#define S second
int bit[N][N];
#define lb(x) (x & -x)
void upd(int i, int j, int v) {
    for(; j < N; j += lb(j))
        for(int k = i; k < N; k += lb(k)) bit[k][j] += v;
}
int qry2(int i, int j) {
    int ans = 0;
    for(; j; j -= lb(j))
        for(int k = i; k; k -= lb(k)) ans += bit[k][j];
    return ans;
}
int qry(int y1, int x1, int y2, int x2) {
    return qry2(y2, x2) - qry2(y2, x1 - 1) - qry2(y1 - 1, x2) +
           qry2(y1 - 1, x1 - 1);
}
int main() {
    int n, q, i = 1, j, y, x;
    for(scanf("%d %d", &n, &q); getchar(), i <= n; ++i)
        for(j = 1; j <= n; ++j)
            if(getchar() == '*') upd(i, j, 1);
    for(; q--;) {
        scanf("%d", &i);
        if(i == 1)
            scanf("%d%d", &i, &j),
                upd(i, j, 1 - 2 * qry(i, j, i, j));
        else
            scanf("%d%d%d%d", &i, &j, &y, &x),
                printf("%d\n", qry(i, j, y, x));
    }
}
```

1.2. DynamicSegmentTree.cpp

```cpp

#define int long long
using namespace std;

int n, q;
struct node {
    int data, lson, rson, tag;
    int rv() { return data + tag; }
};

node tree[20000005];
int a[200005];
int now = 1;
int mx = 1000000005;

void push(int index) {
    if(!tree[index].lson) {
        tree[index].lson = ++now;
    }
    if(!tree[index].rson) {
        tree[index].rson = ++now;
    }
    int lson = tree[index].lson;
    int rson = tree[index].rson;
    tree[lson].tag += tree[index].tag;
    tree[rson].tag += tree[index].tag;
    tree[index].data = tree[index].rv();
    tree[index].tag = 0;
}

void modify(int l, int r, int L, int R, int val, int index) {
    if(l == L && r == R) {
        tree[index].tag += val;
        return;
    }
    int mid = (l + r) >> 1;
    push(index);
    int lson = tree[index].lson;
    int rson = tree[index].rson;
    if(R <= mid) {
        modify(l, mid, L, R, val, lson);
    } else if(L > mid) {
        modify(mid + 1, r, L, R, val, rson);
    } else {
        modify(l, mid, L, mid, val, lson);
        modify(mid + 1, r, mid + 1, R, val, rson);
    }
    tree[index].data = tree[lson].rv() + tree[rson].rv();
}

int query(int l, int r, int L, int R, int index) {
    // cout << L << " " << R << "\n";
    if(l == L && r == R) {
        return tree[index].rv();
    }
    int mid = (l + r) >> 1;
    push(index);
    int lson = tree[index].lson;
    int rson = tree[index].rson;
    if(R <= mid) {
        return query(l, mid, L, R, lson);
    }
    if(L > mid) {
        return query(mid + 1, r, L, R, rson);
    }
    return query(l, mid, L, mid, lson) +
           query(mid + 1, r, mid + 1, R, rson);
}

signed main() {
    ios::sync_with_stdio(0);
    cin.tie(0);
    cout.tie(0);
    cin >> n >> q;
    for(int i = 1; i <= n; i++) {
        cin >> a[i];
        modify(1, mx, a[i], a[i], 1, 1);
    }
    while(q--) {
        char mode;
        int x, y;
        cin >> mode;
        if(mode == '?') {
            cin >> x >> y;
            cout << query(1, mx, x, y, 1) << "\n";
        } else {
            cin >> x >> y;
            modify(1, mx, a[x], a[x], -1, 1);
            a[x] = y;
            modify(1, mx, a[x], a[x], 1, 1);
        }
    }
}
```

1.3. PbdsGpHashTable.cpp

```cpp

using namespace __gnu_pbds;
#define ull unsigned ll
mt19937 mt(hash<string>()("164253_official_beautiful_fruit"));
struct myhash {
    static ull splitmix64(ull x) {
        x += 0x9e3779b97f4a7c15;
        x = (x ^ (x >> 30)) * 0xbf58476d1ce4e5b9;
        x = (x ^ (x >> 27)) * 0x94d049bb133111eb;
        return x ^ (x >> 31);
    }
    ull operator()(ull x) const {
        static const ull FIXED_RANDOM =
            (ull)make_unique<char>().get() ^
            chrono::high_resolution_clock::now()
                .time_since_epoch()
                .count();
        // static const ull FIXED_RANDOM=mt();
        // static const ull
        // FIXED_RANDOM=chrono::steady_clock::now()
        // .time_since_epoch().count();
        return splitmix64(x + FIXED_RANDOM);
    }
};
/*
gp_hash_table<ull,ull,myhash> gp;
gp[x]=y;
if(gp.find(x)!=gp.end())cout<<gp[x];
gp.count(); //CE
*/
```

1.4. PbdsPriorityQueue.cpp

```cpp

__gnu_pbds::priority_queue<int> pq;
/*
push(x); //return iterator
pop() top() join(pq2) erase(iter) modify(iter,x)
*/
```

1.5. PbdsRope.cpp

```cpp

using namespace __gnu_cxx;
/*
rope<int> r;
r.erase(pos,k); //r=r.[0,pos)+r.[pos+k,r.length());
push_back(x) pop_back() insert(pos,x) clear() find(x)
lower_bound(all(r),x) upper_bound //same as vector
r.length(); //same as .length
r.replace(pos,len=r.length(),x); //r.[pos,pos+len)=x;
r.substr(pos,x); //return r.[pos,pos+x);
rope<char> s="official_beautiful_fruit";
cout<<s; //it's legal
*/
```

1.6. PbdsTree.cpp

```cpp

using namespace __gnu_pbds;
/*
tree<int,null_type,less<int>,rb_tree_tag,
    tree_order_statistics_node_update> tr;
//same as rope<int>, except tr.lower_bound(x) and upper_bound
tr.find_by_order(k); //return kth iterator; k=[0,tr.size())
                     //out of this will get tr.end()
tr.order_of_key(val); //return rank(val);
tr.join(tr2); //merge tr
and tr2, tr2.clear() tr.split(const int&r,RBTree&tr2); //<r
will in tr, >=r will in tr2
*/
```

1.7. PersistentSegmentTree.cpp

```cpp
// cses Range Queries and Copies

using namespace std;
#define LL long long
#define pii pair<int, int>
#define N 200005
#define F first
#define S second
int n, ver = 1;
LL a[N];
struct Seg {
    LL v = 0;
    struct Seg *l = NULL, *r = NULL;
#define M (L + R >> 1)
    static const void init(Seg *node, int L = 1, int R = n) {
        if(L == R) {
            node->v = a[L];
            return;
        }
        node->l = new Seg();
        init(node->l, L, M);
        node->r = new Seg();
        init(node->r, M + 1, R);
        node->v = node->l->v + node->r->v;
    }
    static const void upd(Seg *node, int x, LL v, int L = 1,
                          int R = n) {
        if(L == R) {
            node->v = v;
            return;
        }
        if(x <= M)
            node->l = new Seg(*node->l),
            upd(node->l, x, v, L, M);
        else
            node->r = new Seg(*node->r),
            upd(node->r, x, v, M + 1, R);
        node->v = node->l->v + node->r->v;
    }
    static const LL qry(Seg *node, int l, int r, int L = 1,
                        int R = n) {
        if(l <= L && R <= r) return node->v;
        if(r <= M) return qry(node->l, l, r, L, M);
        if(M + 1 <= l) return qry(node->r, l, r, M + 1, R);
        return qry(node->l, l, M, L, M) +
               qry(node->r, M + 1, r, M + 1, R);
    }
} * tree[N];
int main() {
    ios::sync_with_stdio(0);
    cin.tie(0);
    cout.tie(0);
    int q, i = 1, j, k;
    for(cin >> n >> q; i <= n; ++i) cin >> a[i];
    tree[1] = new Seg();
    Seg::init(tree[1]);
    for(; q--;) {
        cin >> i >> k;
        if(i == 1)
            cin >> i >> j, Seg::upd(tree[k], i, j);
        else if(i == 2)
            cin >> i >> j,
                cout << Seg::qry(tree[k], i, j) << "\n";
        else
            tree[++ver] = new Seg(*tree[k]);
    }
}
```

1.8. Treap.cpp

```
#define pii pair<int, int>
struct node {
    int tag = 0;
    int sum = 0;
    int prio = rand();
    int lson = 0;
    int rson = 0;
    int si = 0;
    int val = 0;
};
node treap[400005];
int cnt = 0;
int root = 0;

void update(int index) {
    int lson = treap[index].lson;
    int rson = treap[index].rson;
    treap[index].si = treap[lson].si + treap[rson].si + 1;
    treap[index].sum = treap[lson].sum;
    treap[index].sum += treap[rson].sum;
    treap[index].sum += treap[index].val;
}
void push(int index) {
    if(!treap[index].tag) return;
    swap(treap[index].lson, treap[index].rson);
    int lson = treap[index].lson;
    int rson = treap[index].rson;
    treap[lson].tag ^= 1;
    treap[rson].tag ^= 1;
    treap[index].tag = 0;
}

pii split(int rk, int index) {
    if(!index) return {0, 0};
    push(index);
    int lson = treap[index].lson;
    int rson = treap[index].rson;
    if(rk <= treap[lson].si) {
        pii temp = split(rk, lson);
        treap[index].lson = temp.second;
        update(index);
        return {temp.first, index};
    } else {
        pii temp = split(rk - treap[lson].si - 1, rson);
        treap[index].rson = temp.first;
        update(index);
        return {index, temp.second};
    }
}

int merge(int x, int y) {
    if(!x && !y) return 0;
    if(!x && y) return y;
    if(x && !y) return x;
    push(x);
    push(y);
    if(treap[x].prio < treap[y].prio) {
        treap[x].rson = merge(treap[x].rson, y);
        update(x);
        return x;
    } else {
        treap[y].lson = merge(x, treap[y].lson);
        update(y);
        return y;
    }
}

void insert(int x, int v) {
    pii temp = split(x - 1, root);
    cnt++;
    treap[cnt].val = v;
    update(cnt);
    temp.first = merge(temp.first, cnt);
    root = merge(temp.first, temp.second);
}

int query(int l, int r) {
    pii R = split(r, root);
    pii L = split(l - 1, R.first);
    int ret = treap[L.second].sum;
    R.first = merge(L.first, L.second);
    root = merge(R.first, R.second);
    return ret;
}

void modify(int l, int r) {
    pii R = split(r, root);
    pii L = split(l - 1, R.first);
    treap[L.second].tag ^= 1;
    R.first = merge(L.first, L.second);
    root = merge(R.first, R.second);
}
```

2. Math

2.1. CRT.cpp

```

#define int long long
using namespace std;

int n;
int a[15];
int b[15];
int mul = 1;

void exgcd(int a, int b, int &x, int &y) {
    if(b == 0) {
        x = 1;
        y = 0;
        return;
    }
    exgcd(b, a % b, y, x);
    y -= (a / b) * x;
}

int inv(int a, int p) {
    int x, y;
    exgcd(a, p, x, y);
    return x;
}

int ans = 0;

signed main() {
    cin >> n;
    for(int i = 1; i <= n; i++) {
        cin >> a[i] >> b[i];
```

```cpp
        mul *= a[i];
    }
    for(int i = 1; i <= n; i++) {
        ans += inv(mul / a[i], a[i]) * (mul / a[i]) % mul *
             b[i] % mul;
        ans %= mul;
    }
    ans = (ans + mul) % mul;
    cout << ans;
}
```

2.2. CountPrimes.cpp

```cpp

using namespace std;
using i64 = long long;
i64 count_pi(i64 N) {
    if(N <= 1) return 0;
    int v = sqrt(N + 0.5);
    int n_4 = sqrt(v + 0.5);
    int T = min((int)sqrt(n_4) * 2, n_4);
    int K = pow(N, 0.625) / log(N) * 2;
    K = max(K, v);
    K = min<i64>(K, N);
    int B = N / K;
    B = N / (N / B);
    B = min<i64>(N / (N / B), K);

    vector<i64> l(v + 1);
    vector<int> s(K + 1);
    vector<bool> e(K + 1);
    vector<int> w(K + 1);
    for(int i = 1; i <= v; ++i) l[i] = N / i - 1;
    for(int i = 1; i <= v; ++i) s[i] = i - 1;

    const auto div = [](i64 n, int d) -> int {
        return double(n) / d;
    };
    int p;
    for(p = 2; p <= T; ++p)
        if(s[p] != s[p - 1]) {
            i64 M = N / p;
            int t = v / p, t0 = s[p - 1];
            for(int i = 1; i <= t; ++i) l[i] -= l[i * p] - t0;
            for(int i = t + 1; i <= v; ++i)
                l[i] -= s[div(M, i)] - t0;
            for(int i = v, j = t; j >= p; --j)
                for(int l = j * p; i >= l; --i)
                    s[i] -= s[j] - t0;
            for(int i = p * p; i <= K; i += p) e[i] = 1;
        }
    e[1] = 1;
    int cnt = 1;
    vector<int> roughs(B + 1);
    for(int i = 1; i <= B; ++i)
        if(!e[i]) roughs[cnt++] = i;
    roughs[cnt] = 0x7fffffff;
    for(int i = 1; i <= K; ++i) w[i] = e[i] + w[i - 1];
    for(int i = 1; i <= K; ++i) s[i] = w[i] - w[i - (i & -i)];

    const auto query = [&](int x) -> int {
        int sum = x;
        while(x) sum -= s[x], x ^= x & -x;
        return sum;
    };
    const auto add = [&](int x) -> void {
        e[x] = 1;
        while(x <= K) ++s[x], x += x & -x;
    };
    cnt = 1;
    for(; p <= n_4; ++p)
        if(!e[p]) {
            i64 q = i64(p) * p, M = N / p;
            while(cnt < q) w[cnt] = query(cnt), cnt++;
            int t1 = B / p, t2 = min<i64>(B, M / q),
                t0 = query(p - 1);
            int id = 1, i = 1;
            for(; i <= t1; i = roughs[++id])
                l[i] -= l[i * p] - t0;
            for(; i <= t2; i = roughs[++id])
                l[i] -= query(div(M, i)) - t0;
            for(; i <= B; i = roughs[++id])
                l[i] -= w[div(M, i)] - t0;
            for(int i = q; i <= K; i += p)
                if(!e[i]) add(i);
        }
    while(cnt <= v) w[cnt] = query(cnt), cnt++;

    vector<int> primes;
    primes.push_back(1);
    for(int i = 2; i <= v; ++i)
        if(!e[i]) primes.push_back(i);
    l[1] += i64(w[v] + w[n_4] - 1) * (w[v] - w[n_4]) / 2;
    for(int i = w[n_4] + 1; i <= w[B]; ++i)
        l[1] -= l[primes[i]];
    for(int i = w[B] + 1; i <= w[v]; ++i)
        l[1] -= query(N / primes[i]);
    for(int i = w[n_4] + 1; i <= w[v]; ++i) {
        int q = primes[i];
        i64 M = N / q;
        int e = w[M / q];
        if(e <= i) break;
        l[1] += e - i;
        i64 t = 0;
        int m = w[sqrt(M + 0.5)];
        for(int k = i + 1; k <= m; ++k)
            t += w[div(M, primes[k])];
        l[1] += 2 * t - (i + m) * (m - i);
    }
    return l[1];
}
```

2.3. FFT.cpp

```cpp

using namespace std;
inline int read() {
    int ans = 0;
    char c = getchar();
    while(!isdigit(c)) c = getchar();
    while(isdigit(c)) {
        ans = ans * 10 + c - '0';
        c = getchar();
    }
    return ans;
}
typedef complex<double> comp;
const int MAXN = 1000005;
const comp I(0, 1);
const double PI = acos(-1);
comp A[MAXN * 3], B[MAXN * 3], tmp[MAXN * 3], ans[MAXN * 3];
void fft(comp F[], int N, int sgn = 1) {
    if(N == 1) return;
    memcpy(tmp, F, sizeof(comp) * N);
    for(int i = 0; i < N; i++)
        *(i % 2 ? F + i / 2 + N / 2 : F + i / 2) = tmp[i];
    fft(F, N / 2, sgn), fft(F + N / 2, N / 2, sgn);
    comp *G = F, *H = F + N / 2;
    comp cur = 1, step = exp(2 * PI / N * sgn * I);
    for(int k = 0; k < N / 2; k++) {
        tmp[k] = G[k] + cur * H[k];
        tmp[k + N / 2] = G[k] - cur * H[k];
        cur *= step;
    }
    memcpy(F, tmp, sizeof(comp) * N);
}
int main() {
    int n = read(), m = read(), N = 1 << __lg(n + m + 1) + 1;
    for(int i = 0; i <= n; ++i) A[i] = read();
    for(int i = 0; i <= m; ++i) B[i] = read();
    fft(A, N), fft(B, N);
    for(int i = 0; i < N; ++i) ans[i] = A[i] * B[i];
    fft(ans, N, -1);
    for(int i = 0; i <= n + m; ++i)
        printf("%d ", int(ans[i].real() / N + 0.1));
    return 0;
}
```

2.4. FWT.cpp

```cpp
#define LOGN 21
#define N (1 << LOGN)
void fwt(ll f[], int rev) {
    for(int k = 1; k < LOGN; ++k) {
        for(int i = 0, m = 1 << k - 1; i + m < N; i += 1 << k) {
            for(int j = 0; j < m; ++j) {
                ll u = f[i + j], v = f[i + j + m];
                f[i + j] = u + v;
                f[i + j + m] = u - v;
                if(rev) f[i + j] >>= 1, f[i + j + m] >>= 1;
            }
        }
    }
}
```

2.5. Formula.tex

2.5.1. Dirichlet Convolution

$\varepsilon = \mu * 1$
$\varphi = \mu * \mathrm{Id}$

2.5.2. Burnside's Lemma

Let X be a set and G be a group that acts on X. For $g \in G$, denote by X^g the elements fixed by g:

$$X^g = \{x \in X \mid gx \in X\}$$

Then

$$|X/G| = \frac{1}{|G|}\sum_{g\in G}|X^g|.$$

2.5.3. Pick Theorem

$A = i + \frac{b}{2} - 1$

2.5.4. Fermat's Little Theorem

$(a+b)^p \equiv a + b \equiv a^p + b^p \pmod p$

2.5.5. Wilson's Theorem

$(p-1)! \equiv -1 \pmod p$

2.5.6. Legendre Theorem

v(n):=power of p in n
$(n)_p := \frac{n}{p^{(v(n))}}$
s(n):=sum of all digits of n in base p
$v(n!) = \sum_{i=1}^{\infty}\lfloor\frac{n}{p^i}\rfloor = \frac{n-s(n)}{p-1}$

2.5.7. Kummer Theorem

$v(\binom{n}{m}) = \frac{s(n)+s(m-n)-s(m)}{p-1}$

2.5.8. ext-Kummer Theorem

$v(\binom{n}{m1,m2,...mk}) = \frac{\sum_{i=1}^{k} s(mi)-s(n)}{p-1}$

2.5.9. Factorial with mod

$(n!)_p \equiv -1^{\lfloor\frac{n}{p}\rfloor}((\lfloor\frac{n}{p}\rfloor)!)_p((n\%p)!) \pmod p$ $O(p+\log_p(n))$ with factorial table.

2.5.10. Properties of nCr with mod

If any i in base p satisfies $n_i < m_i$, then $\binom{n_i}{m_i}\%p = 0$. Therefore $\binom{n}{m} = \prod_{i=0}^{\max(\log_p(a),\log_p(b))}\binom{n_i}{m_i}\%p$ so $\binom{n}{m}\%p = 0$. If $p = 2$, then $\binom{n}{m}$ is odd <=> any bit in $n < m$. Lucas' theorem can be derived from this generating function method without relying on Fermat's Little Theorem. It is also true for polynomials.

2.5.11. ext-Lucas' Theorem

For any $k \in$ positive number, calculate $\binom{n}{m}\%k$ can decompose k by Fundamental Theorem of Arithmetic. And then use crt.

2.5.12. Catalan Number

$C_0 = C_1 = 1, \text{if } n > 1 \text{then } C_n = \sum_{k=0}^{n-1} C_k C_{n-1-k} = \frac{\binom{2n}{n}}{n+1}$ Also the number of legal placements of n pairs of brackets is C_n. If there are any k kinds of brackets available, then $k^n C_n$.

2.5.13. modinv table

$p = i*(p/i) + p\%i, -p\%i = i*(p/i), inv(i) = -(p/i)*inv(p\%i)$

2.6. Gaussian-Jordan.cpp

```cpp

#define int long long
using namespace std;

int n;
double a[105][105];

// n <= m
void gaussian(double a[105][105], int n, int m) {
    int curi = 0;
    for(int j = 0; j < m; j++) {
        int i;
        for(i = curi; i < n; i++) {
            if(a[i][j]) {
                break;
            }
        }
        if(a[i][j] == 0) continue;
        for(int k = 0; k < m; k++) {
            swap(a[i][k], a[curi][k]);
        }
        for(int k = m - 1; k >= j; k--) {
            a[curi][k] /= a[curi][j];
        }
        for(int i = 0; i < n; ++i) {
            if(i != curi) {
                for(int k = m - 1; k >= j; k--) {
                    a[i][k] -= a[curi][k] * a[i][j];
                }
            }
        }
        curi++;
    }
}
```

2.7. Generator.cpp

```cpp

#define int long long
using namespace std;

int t;
int n, d;
bitset<1000005> exist;
bitset<1000005> vis;
vector<int> prime;
int phi[1000005];

void init() {
    phi[1] = 1;
    for(int i = 2; i <= 1000000; i++) {
        if(!vis[i]) {
            prime.push_back(i);
            phi[i] = i - 1;
        }
        for(int j : prime) {
            if(i * j > 1000000) break;
            vis[i * j] = 1;
            if(i % j == 0) {
                phi[i * j] = phi[i] * j;
                break;
            } else {
                phi[i * j] = phi[i] * phi[j];
            }
        }
    }
    exist[2] = exist[4] = 1;
    for(int i : prime) {
        if(i == 2) continue;
        for(int j = i; j <= 1000000; j *= i) {
            exist[j] = 1;
            if(j * 2 <= 1000000) {
                exist[j << 1] = 1;
            }
        }
    }
}

vector<int> factors(int x) {
    vector<int> v;
    for(int i = 1; i * i <= x; i++) {
        if(x % i == 0) {
            v.push_back(i);
            if(i * i != x) {
                v.push_back(x / i);
            }
        }
    }
    return v;
}

int f(int x, int y, int mod) {
    int ret = 1;
    while(y) {
        if(y & 1) {
            ret *= x;
            ret %= mod;
        }
        x *= x;
        x %= mod;
        y >>= 1;
    }
    return (ret % mod + mod) % mod;
}

vector<int> findroot(int x) {
    vector<int> ret;
    if(!exist[x]) return ret;
    int phix = phi[x];
    vector<int> fact = factors(phix);
    int fst;
    for(int i = 1;; i++) {
        if(__gcd(i, x) != 1) continue;
        bool ok = 1;
        for(int j : fact) {
            if(j != phix && f(i, j, x) == 1) {
                ok = 0;
                break;
            }
        }
        if(ok) {
            fst = i;
            break;
        }
    }
    int now = fst;
    // cout << fst <<"\n";
    for(int i = 1; i <= phix; i++) {
        if(__gcd(i, phix) == 1) {
            ret.push_back(now);
        }
        now *= fst;
        now %= x;
    }
    return ret;
}

signed main() {
    ios::sync_with_stdio(0);
    cin.tie(0);
    cout.tie(0);
    init();
    cin >> t;
    while(t--) {
        cin >> n >> d;
        vector<int> v = findroot(n);
        sort(v.begin(), v.end());
        cout << v.size() << "\n";
        for(int i = 0; i < v.size(); i++) {
            if(i % d == d - 1) {
                cout << v[i] << " ";
            }
        }
        cout << "\n";
    }
}
```

2.8. Inv.cpp

```cpp
int exgcd(int a, int b, int &x, int &y) {
    if(b == 0) {
        x = 1;
        y = 0;
        return a;
    }
    int d = exgcd(b, a % b, y, x);
    y -= x * (a / b);
    return d;
}

int inv(int a, int p) {
    int x, y;
    exgcd(a, p, x, y);
    return (x % p + p) % p;
}
```

2.9. Lucas.cpp

```cpp
int fact[100005];
int p;

void init() {
    fact[0] = 1;
    for(int i = 1; i <= p; i++) {
        fact[i] = fact[i - 1] * i % p;
    }
}

int inv(int x, int p) {
    if(x == 1) return 1;
    return (p - p / x) * inv(p % x, p) % p;
}

int c(int x, int y, int p) {
    if(x < y) return 0;
    int k = fact[x] * inv(fact[y], p) % p;
    return k * inv(fact[x - y], p) % p;
}

int lucas(int x, int y, int p) {
    if(x == 0) return 1;
    return lucas(x / p, y / p, p) % p * c(x % p, y % p, p) % p;
}
```

2.10. MillerRabin.cpp

```cpp
#define uLL __uint128_t
template <class T, class POW>
void fastpow(T x, POW n, POW p, T &ans) {
    for(; n; n >>= 1) {
        if(n & 1) {
            ans *= x;
            ans %= p;
        }
        x *= x;
        x %= p;
    }
}
/* 輸入 x,n,p,ans 會將 ans 修改為 x^n%p
對整數/矩陣/不要求精度的浮點 皆有效
模板第一個型別是 x,ans 第二個是 n,p(應該放 LL 或 __int128)*/
uLL pri[7] = {2,      325,     9375,      28178,
              450775, 9780504, 1795265022}; /*2^64*/
// int p[3]={2,7,61};/*2^32*/
bool check(const uLL x, const uLL p) {
    uLL d = x - 1, ans = 1;
    fastpow(p, d, x, ans);
    if(ans != 1) return 1;
    for(; !(d & 1);) {
        d >>= 1;
        ans = 1;
        fastpow(p, d, x, ans);
        if(ans == x - 1)
            return 0;
        else if(ans != 1)
            return 1;
    }
    return 0;
}
bool miller_rabin(const uLL x) {
    if(x == 1) return 0;
    for(auto e : pri) {
        if(e >= x) return 1;
        if(check(x, e)) return 0;
    }
    return 1;
}
```

2.11. Mu.cpp

```cpp
vector<int> prime;
bitset<1000005> vis;
int n;
int mu[1000005];

void init() {
    for(int i = 2; i <= n; i++) {
        if(!vis[i]) {
            prime.push_back(i);
            mu[i] = -1;
        }
        for(int p : prime) {
            if(i * p > n) break;
            vis[i * p] = 1;
            if(i % p == 0) {
                mu[i * p] = 0;
                break;
            } else {
                mu[i * p] = mu[i] * mu[p];
            }
        }
    }
}
```

2.12. NTT.cpp

```cpp

#define ll long long
using namespace std;

const int MAXN = 1000005;
const int MOD = 998244353, G = 3;
int rev[MAXN * 3];

int qpow(int x, int y) {
    int ret = 1;
    while(y) {
        if(y & 1) {
            ret *= x;
            ret %= MOD;
        }
        x *= x;
        x %= MOD;
        y >>= 1;
    }
    return ret;
}
```

```cpp
void ntt(int F[], int N, int sgn) {
    int bit = __lg(N);
    for(int i = 0; i < N; ++i) {
        rev[i] = (rev[i >> 1] >> 1) | ((i & 1) << (bit - 1));
        if(i < rev[i]) swap(F[i], F[rev[i]]);
    }
    for(int l = 1, t = 1; l < N; l <<= 1, t++) {
        int step = qpow(G, ((MOD - 1) >> t) * sgn + MOD - 1);
        for(int i = 0; i < N; i += l << 1)
            for(int k = i, cur = 1; k < i + l; ++k) {
                int g = F[k], h = (ll)F[k + l] * cur % MOD;
                F[k] = (g + h) % MOD;
                F[k + l] = ((g - h) % MOD + MOD) % MOD;
                cur = (ll)cur * step % MOD;
            }
    }
    if(sgn == -1) {
        int invN = qpow(N, MOD - 2);
        for(int i = 0; i < N; ++i) F[i] = (ll)F[i] * invN % MOD;
    }
}
```

2.13. PollardRho.cpp

```cpp

using namespace std;
#define LL long long
#define uLL __uint128_t
#define sub(a, b) ((a) < (b) ? (b) - (a) : (a) - (b))
template <class T, class POW>
void fastpow(T x, POW n, POW p, T &ans) {
    for(; n; n >>= 1) {
        if(n & 1) {
            ans *= x;
            ans %= p;
        }
        x *= x;
        x %= p;
    }
}
/*input x, n, p, ans, will modify ans to x ^ n % p
the first is x, ans and the second is n, p (LL or __int128)
*/
uLL pri[7] = {2,      325,     9375,      28178,
              450775, 9780504, 1795265022}; /*2^64*/
// int p[3]={2,7,61};/*2^32*/
bool check(const uLL x, const uLL p) {
    uLL d = x - 1, ans = 1;
    fastpow(p, d, x, ans);
    if(ans != 1) return 1;
    for(; !(d & 1);) {
        d >>= 1;
        ans = 1;
        fastpow(p, d, x, ans);
        if(ans == x - 1)
            return 0;
        else if(ans != 1)
            return 1;
    }
    return 0;
}
bool miller_rabin(const uLL x) {
    if(x == 1) return 0;
    for(auto e : pri) {
        if(e >= x) return 1;
        if(check(x, e)) return 0;
    }
    return 1;
}
template <class T> T gcd(T a, T b) {
    if(!a) return b;
    if(!b) return a;
    if(a & b & 1) return gcd(sub(a, b), min(a, b));
    if(a & 1) return gcd(a, b >> 1);
    if(b & 1) return gcd(a >> 1, b);
    return gcd(a >> 1, b >> 1) << 1;
}
/*gcd(a,b) denote gcd(a, 0) = a*/
mt19937 rnd(time(0));
template <class T> T f(T x, T c, T mod) {
    return (((uLL)x) * x % mod + c) % mod;
}
template <class T> T rho(T n) {
    T mod = n, x = rnd() % mod, c = rnd() % (mod - 1) + 1,
      p = 1;
    for(T i = 2, j = 2, d = x;; ++i) {
        x = f(x, c, mod);
        p = ((uLL)p) * sub(x, d) % mod;
        if(i % 127 == 0 && gcd(p, n) != 1) return gcd(p, n);
        if(i == j) {
            j <<= 1, d = x;
            if(gcd(p, n) != 1) return gcd(p, n);
        }
    }
}
template <class T> T pollard_rho(T n) {
    if(miller_rabin(n)) return n;
    T p = n;
    while(p == n) p = rho(n);
    return max(pollard_rho(p), pollard_rho(n / p));
}
int main() {
    LL t, n, ans;
    for(cin >> t; t--;) {
        cin >> n;
        ans = pollard_rho(n);
        if(ans == n)
            puts("Prime");
        else
            printf("%lld\n", ans);
    }
}
```

2.14. XorBasis.cpp

```cpp
#pragma GCC optimize(                                   \
    "Ofast,fast-math,unroll-loops,no-stack-protector")

using namespace std;
#define ll long long
#define V vector
#define pb push_back
#define all(x) x.begin(), x.end()
V<ll> v;
ll f(ll k, ll now = 0, ll p = v.size() - 1, ll ans = 0) {
    if(k >= 1 << p) {
        k -= 1 << p;
        ans = max(ans, ans ^ v[now]);
    } else
        ans = min(ans, ans ^ v[now]);
    if(!p) return ans;
    return f(k, now + 1, p - 1, ans);
}
int main() {
    ios::sync_with_stdio(0);
    cin.tie(0);
    cout.tie(0);
    ll n, k;
    cin >> n >> k;
    for(ll x, i = 0; i < n; ++i) {
        cin >> x;
        for(ll &e : v) x = min(x, x ^ e);
        if(x) v.pb(x);
    }
    sort(all(v), greater<ll>());
    ll t = n - v.size(), a = k >> t,
       b = k & ((1 << min(t, 20LL)) - 1), i = 0;
    for(; a--; ++i)
        for(ll j = 1 << t, p = f(i); j--;) cout << p << " ";
    for(i = f(i); b--;) cout << i << " ";
}
```

2.15. mtt.cpp

```cpp

using namespace std;
// https://www.luogu.com.cn/article/08nmgxd1
namespace poly {
long double const pi = acos(-1);
struct comp {
    long double r, i;
    comp() { r = i = 0; }
    comp(long double x, long double y) { r = x, i = y; }
    comp conj() { return comp(r, -i); }
    friend comp operator+(comp x, comp y) {
        return comp(x.r + y.r, x.i + y.i);
    }
    friend comp operator-(comp x, comp y) {
        return comp(x.r - y.r, x.i - y.i);
    }
    friend comp operator*(comp x, comp y) {
        return comp(x.r * y.r - x.i * y.i,
                    x.i * y.r + x.r * y.i);
    }
};
typedef long long ll;
int r[400005];
comp a[400005], b[400005], c[400005], d[400005];
void fft(comp *f, int n, int op) {
    for(int i = 1; i < n; i++)
        r[i] = (r[i >> 1] >> 1) + ((i & 1) ? (n >> 1) : 0);
    for(int i = 1; i < n; i++)
        if(i < r[i]) swap(f[i], f[r[i]]);
    for(int len = 2; len <= n; len <<= 1) {
        int q = len >> 1;
        comp wn = comp(cos(pi / q), op * sin(pi / q));
        for(int i = 0; i < n; i += len) {
            comp w = comp(1, 0);
            for(int j = i; j < i + q; j++, w = w * wn) {
                comp d = f[j + q] * w;
                f[j + q] = f[j] - d;
                f[j] = f[j] + d;
            }
        }
    }
}
void mtt(int *f, int *g, int *h, int n, int p) {
    for(int i = 0; i < n; i++) {
        a[i].r = (f[i] >> 15);
        a[i].i = (f[i] & 32767);
        c[i].r = (g[i] >> 15);
        c[i].i = (g[i] & 32767);
    }
    fft(a, n, 1), fft(c, n, 1);
    for(int i = 1; i < n; i++) b[i] = a[n - i].conj();
    b[0] = a[0].conj();
    for(int i = 1; i < n; i++) d[i] = c[n - i].conj();
    d[0] = c[0].conj();
    for(int i = 0; i < n; i++) {
        comp aa = (a[i] + b[i]) * comp(0.5, 0);
        comp bb = (a[i] - b[i]) * comp(0, -0.5);
        comp cc = (c[i] + d[i]) * comp(0.5, 0);
        comp dd = (c[i] - d[i]) * comp(0, -0.5);
        a[i] = aa * cc + comp(0, 1) * (aa * dd + bb * cc);
        b[i] = bb * dd;
    }
    fft(a, n, -1), fft(b, n, -1);
    for(int i = 0; i < n; i++) {
        int aa = (ll)(a[i].r / n + 0.5) % p,
            bb = (ll)(a[i].i / n + 0.5) % p,
            cc = (ll)(b[i].r / n + 0.5) % p;
        h[i] = ((1ll * aa * (1 << 30) + 1ll * bb * (1 << 15) +
                 cc) %
                    p +
                p) %
               p;
    }
}
} // namespace poly
using namespace poly;
int f[400005], g[400005], h[400005];
// 400005 is 2 * (n + m)
int main() {
    int n, m, p;
    scanf("%d%d%d", &n, &m, &p);
    for(int i = 0; i <= n; i++) scanf("%d", &f[i]);
    for(int i = 0; i <= m; i++) scanf("%d", &g[i]);
    int lim = 1;
    while(lim <= (n + m)) lim <<= 1;
    mtt(f, g, h, lim, p);
    for(int i = 0; i <= n + m; i++) printf("%d ", h[i]);
    return 0;
}
```

3. String

3.1. Booth.cpp

```
#define V vector
string booth(string s) {
    s += s;
    int n = s.size(), k = 0;
    V<int> f(n, -1);
    for(int i = 1; i < n; ++i) {
        int j = f[i - k - 1];
        for(; j >= 0 && s[j + k + 1] != s[i]; j = f[j])
            if(s[i] < s[j + k + 1]) k = i - j - 1;
        if(s[i] != s[j + k + 1]) {
            if(s[i] < s[k]) k = i;
            f[i - k] = -1;
        } else
            f[i - k] = j + 1;
    }
    return s.substr(k, s.size() >> 1);
}
// 給出循環排列後最小字典序的解
```

3.2. KMP.cpp

```
string s, t;
int pmt[1000005];

void init() {
    for(int i = 1, j = 0; i < t.size(); i++) {
        while(j && t[j] ^ t[i]) {
            j = pmt[j - 1];
        }
        if(t[j] == t[i]) j++;
        pmt[i] = j;
    }
}

int kmp(string s) {
    int ret = 0;
    for(int i = 0, j = 0; i < s.size(); i++) {
        while(j && s[i] ^ t[j]) {
            j = pmt[j - 1];
        }
        if(s[i] == t[j]) {
            j++;
        }
        if(j == t.size()) {
            ret++;
            j = pmt[j - 1];
        }
    }
    return ret;
}
```

3.3. LongestPalindrome.cpp

```

#define int long long
using namespace std;

string s;
string t;
int n;
int d[2000005];
int ans = 0;

signed main() {
    cin >> t;
    n = t.size();
    for(int i = 0; i < 2 * n + 1; i++) {
        if(i & 1 ^ 1) {
            s += '0';
        } else {
            s += t[i / 2];
        }
    }
    n = s.size();
    d[0] = 1;
    for(int i = 0, l = 0, r = 0; i < n; i++) {
        if(i > r) {
            d[i] = 1;
            bool a = i + d[i] < n;
            bool b = i - d[i] >= 0;
      bool c = (s[i + d[i]] == s[i - d[i]]);
      while (a && b && c) {
                d[i]++;
                a = i + d[i] < n;
                b = i - d[i] >= 0;
                c = ([i + d[i]] == s[i - d[i]]);
      }
      l = i - d[i] + 1;
      r = i + d[i] - 1;
        } else {
            int j = l + r - i;
            if(j - d[j] + 1 > l) {
                d[i] = d[j];
            } else {
                d[i] = r - i + 1;
                a = i + d[i] < n;
                b = i - d[i] >= 0;
                c = (s[i + d[i]] == s[i - d[i]]);
                while(a && b && c) {
                    d[i]++;
                    a = i + d[i] < n;
                    b = i - d[i] >= 0;
                    c = (s[i + d[i]] == s[i - d[i]]);
                }
                l = i - d[i] + 1;
                r = i + d[i] - 1;
            }
        }
        // cout << d[i] << " ";
        if(d[i] > d[ans]) {
            ans = i;
        }
    }
    for(int i = ans - d[ans] + 1; i < ans + d[ans]; i++) {
        if(s[i] ^ '0') {
            cout << s[i];
        }
    }
}
```

3.4. Z.cpp

```cpp

#define int long long
using namespace std;

string s, t;
int ans = 0;

int z[2000005];

signed main() {
    ios::sync_with_stdio(0);
    cin.tie(0);
    cout.tie(0);
    cin >> s >> t;
    s = t + '0' + s;
    int n, m;
    n = s.size();
    m = t.size();
    for(int i = 0, l = 0, r = 0; i < n; i++) {
        if(z[i - l] < r - i + 1) {
            z[i] = z[i - l];
        } else {
            z[i] = max(r - i + 1, (int)0);
            while(i + z[i] < n && s[i + z[i]] == s[z[i]]) {
                z[i]++;
            }
            l = i;
            r = i + z[i] - 1;
            if(z[i] == m) {
                ans++;
            }
        }
    }
    cout << ans;
}
```

4. Graph

4.1. 2-SAT(CSES Planets Cycles).cpp

```cpp

#define int long long
using namespace std;

int n, m;
vector<int> v[200005];
int d[200005];
int low[200005];
int cnt = 0;
int now = 0;
int scc[200005];
stack<int> s;
int op[200005];
vector<int> v2[200005];
int ind[200005];
queue<int> q;
int ans[200005];

int no(int x) {
    if(x > m) return x - m;
    return x + m;
}

void dfs(int x) {
    d[x] = low[x] = ++cnt;
    s.push(x);
    for(int i : v[x]) {
        if(scc[i]) continue;
        if(d[i]) {
            low[x] = min(low[x], d[i]);
        } else {
            dfs(i);
            low[x] = min(low[x], low[i]);
        }
    }
    if(d[x] == low[x]) {
        now++;
        while(!s.empty()) {
            int k = s.top();
            s.pop();
            scc[k] = now;
            if(k == x) break;
        }
    }
}

signed main() {
    ios::sync_with_stdio(0);
    cin.tie(0);
    cout.tie(0);
    cin >> n >> m;
    while(n--) {
        char a, b;
        int x, y;
        cin >> a >> x >> b >> y;
        if(a == '-') x = no(x);
        if(b == '-') y = no(y);
        v[no(x)].push_back(y);
        v[no(y)].push_back(x);
    }
    for(int i = 1; i <= 2 * m; i++) {
        if(!d[i]) {
            dfs(i);
        }
    }
    for(int i = 1; i <= m; i++) {
        if(scc[i] ^ scc[i + m]) {
            op[scc[i]] = scc[i + m];
            op[scc[i + m]] = scc[i];
        } else {
            cout << "IMPOSSIBLE";
            exit(0);
        }
    }
    for(int i = 1; i <= 2 * m; i++) {
        for(int j : v[i]) {
            if(scc[i] ^ scc[j]) {
                v2[scc[j]].push_back(scc[i]);
                ind[scc[i]]++;
            }
        }
    }
    for(int i = 1; i <= now; i++) {
        if(!ind[i]) {
            q.push(i);
        }
    }
    while(!q.empty()) {
        int k = q.front();
        q.pop();
        if(!ans[k]) {
            ans[k] = 1;
            ans[op[k]] = 2;
        }
        for(int i : v2[k]) {
            ind[i]--;
            if(!ind[i]) {
                q.push(i);
            }
        }
    }
    for(int i = 1; i <= m; i++) {
        if(ans[scc[i]] == 1) {
            cout << "+ ";
        } else {
            cout << "- ";
        }
    }
}
```

4.2. Dijkstra.cpp

```cpp
vector<pair<int, int>> v[100005], v2[100005];
vector<edge> es;
int dis1[100005];
int dis2[100005];
bitset<100005> vis1, vis2;

void dijkstra(int x, int *dis, vector<pair<int, int>> *v,
              bitset<100005> &vis) {
    priority_queue<pair<int, int>, vector<pair<int, int>>,
                   greater<pair<int, int>>>
        pq;
    memset(dis, 0x3f, sizeof(dis1));
    vis.reset();
    dis[x] = 0;
    pq.push({0, x});
    while(!pq.empty()) {
        pair<int, int> now = pq.top();
        pq.pop();
        if(vis[now.second]) continue;
        vis[now.second] = 1;
        for(auto [i, w] : v[now.second]) {
            if(vis[i]) continue;
            if(dis[now.second] + w < dis[i]) {
                dis[i] = dis[now.second] + w;
                pq.push({dis[i], i});
            }
        }
    }
}
```

4.3. Dinic.cpp

```cpp

using namespace std;
#define ll long long
const ll inf = 8e18;
#define N 505
#define pb push_back
struct pp {
    int from, to;
    ll flow;
};
int t, lvl[N], p[N];
vector<int> g[N];
vector<pp> edge;
int bfs(int s) {
    queue<int> q;
    for(q.push(s), lvl[s] = 1; !q.empty(); q.pop()) {
        int u = q.front();
        for(int e : g[u]) {
            int v = edge[e].to;
            if(lvl[v] || !edge[e].flow) continue;
            lvl[v] = lvl[u] + 1;
            q.push(v);
        }
    }
    return lvl[t];
}
ll dfs(int u, ll f = inf) {
    if(u == t || !f) return f;
    ll ans = 0;
    for(int &i = p[u]; i < g[u].size(); ++i) {
        pp &e = edge[g[u][i]], &b = edge[g[u][i] ^ 1];
        if(lvl[e.to] == lvl[u] + 1) {
            ll c = dfs(e.to, min(e.flow, f));
            e.flow -= c;
            b.flow += c;
            f -= c;
            ans += c;
        }
    }
    return ans;
}
ll dinic(int s) {
    ll ans = 0;
    for(; bfs(s); memset(lvl, 0, sizeof lvl))
        for(ll k; k = (memset(p, 0, sizeof(p)), dfs(s));)
            ans += k;
    return ans;
}
int main() {
    ios::sync_with_stdio(0);
    cin.tie(0);
    cout.tie(0);
    int n, m, cnt = 0;
    for(cin >> n >> m; m--;) {
        int u, v;
        ll f;
        cin >> u >> v >> f;
        g[u].pb(cnt++);
        g[v].pb(cnt++);
        edge.pb({u, v, f});
        edge.pb({v, u, 0});
    }
    t = n;
    cout << dinic(1);
}
```

4.4. MaximumFlow.cpp

```cpp

#define int long long
using namespace std;

int n, m;
vector<int> v[1005];
int head[1005];
int c[1005][1005];
int lv[1005];
int ans = 0;

bool bfs() {
    memset(head, 0, sizeof(head));
    memset(lv, 0, sizeof(lv));
    queue<int> q;
    q.push(1);
    while(!q.empty()) {
        int now = q.front();
        q.pop();
        if(now == n) continue;
        for(int i : v[now]) {
            if(i != 1 && c[now][i] && !lv[i]) {
                lv[i] = lv[now] + 1;
                q.push(i);
            }
        }
    }
    return lv[n];
}

int dfs(int x, int flow) {
    int ret = 0;
    if(x == n) return flow;
    for(int i = head[x]; i < v[x].size(); i++) {
        int y = v[x][i];
        head[x] = y;
        if(c[x][y] && lv[y] == lv[x] + 1) {
            int d = dfs(y, min(flow, c[x][y]));
            flow -= d;
            c[x][y] -= d;
            c[y][x] += d;
            ret += d;
        }
    }
    return ret;
}

signed main() {
    cin >> n >> m;
    while(m--) {
        int x, y, z;
        cin >> x >> y >> z;
        if(c[x][y] || c[y][x]) {
            c[x][y] += z;
            continue;
        }
        v[x].push_back(y);
        v[y].push_back(x);
        c[x][y] = z;
    }
    while(bfs()) {
        ans += dfs(1, INT_MAX);
    }
    cout << ans;
}
```

4.5. SCC.cpp

```cpp
int n, m;
vector<int> v[100005];
int d[100005];
int low[100005];
int cnt = 0;
stack<int> s;
int scc[100005];
int now = 0;

void dfs(int x) {
    d[x] = low[x] = ++cnt;
    s.push(x);
    for(int i : v[x]) {
        if(scc[i]) continue;
        if(d[i]) {
            low[x] = min(low[x], d[i]);
        } else {
            dfs(i);
            low[x] = min(low[x], low[i]);
        }
    }
    if(d[x] == low[x]) {
        now++;
        while(!s.empty()) {
            int k = s.top();
            s.pop();
            scc[k] = now;
            if(k == x) break;
        }
    }
}
```

4.6. VBCC.cpp

```cpp

using namespace std;
#define pb push_back
#define pii pair<int, int>
#define N 100005
vector<int> adj[N], bcc[N];
stack<int> st;
int dfn[N], low[N], tag, bc, root;
bitset<N> ap;
void dfs(int now, int par = -1) {
    st.push(now);
    low[now] = dfn[now] = ++tag;
    int f = 0;
    for(int e : adj[now] | views::reverse) {
        if(e == par) continue;
        if(!dfn[e]) {
            dfs(e, now), low[now] = min(low[now], low[e]);
            if(low[e] >= dfn[now]) {
                if(++f > 1 || now != root) ap[now] = 1;
                ++bc;
                for(; st.top() != now; st.pop())
                    bcc[bc].pb(st.top());
                bcc[bc].pb(now);
            }
        } else
            low[now] = min(low[now], dfn[e]);
    }
}
int main() {
    int n, m, u, v;
    cin >> n >> m;
    vector<pii> g(m);
    for(auto &[u, v] : g)
        cin >> u >> v, adj[u].pb(v), adj[v].pb(u);
    for(root = 1; root <= n; ++root)
        if(!dfn[u]) dfs(root);
    int ans = 0;
    for(int i : views::iota(1) | views::take(n))
        if(ap[i]) ++ans;
    cout << ans << "\n";
    for(int i : views::iota(1) | views::take(n))
        if(ap[i]) cout << i << " ";
}
```

4.7. one-degree-cycle(CSES Planets Cycles).cpp

```cpp

#define int long long
using namespace std;

int n, q;
int a[200005];
int r[200005];
int d[200005];
int cycle[200005];
int len[200005];
int cnt = 0;
vector<int> v[200005];
bitset<200005> vis1;
bitset<200005> vis2;

void findcycle(int x) {
    while(!vis1[x]) {
        vis1[x] = 1;
        x = a[x];
    }
    cnt++;
    cycle[x] = cnt;
    r[x] = 0;
    len[cnt] = 1;
    int temp = a[x];
    while(temp ^ x) {
        r[temp] = len[cnt];
        len[cnt]++;
        cycle[temp] = cnt;
        temp = a[temp];
    }
}

void dfs(int x) {
    if(vis2[x]) return;
    vis2[x] = 1;
    for(int i : v[x]) {
        dfs(i);
    }
}

void dfs2(int x) {
    if(cycle[x] || d[x]) return;
    dfs2(a[x]);
    d[x] = d[a[x]] + 1;
    r[x] = r[a[x]];
    cycle[x] = cycle[a[x]];
}

signed main() {
    ios::sync_with_stdio(0);
    cin.tie(0);
    cout.tie(0);
    cin >> n;
    for(int i = 1; i <= n; i++) {
        cin >> a[i];
        v[i].push_back(a[i]);
        v[a[i]].push_back(i);
    }
    for(int i = 1; i <= n; i++) {
        if(!vis2[i]) {
            findcycle(i);
            dfs(i);
        }
    }
    for(int i = 1; i <= n; i++) {
        if(!cycle[i] && !r[i]) {
            dfs2(i);
        }
    }
    for(int i = 1; i <= n; i++) {
        cout << d[i] + len[cycle[i]] << " ";
    }
}
```

5. DP

5.1. CHO.cpp

```cpp
struct line {
    int a, b;
    int y(int x) { return a * x + b; }
};

struct CHO {
    deque<line> dq;
    int intersect(line x, line y) {
        int d1 = x.b - y.b;
        int d2 = y.a - x.a;
        return d1 / d2;
    }
    bool check(line x, line y, line z) {
        int I12 = intersect(x, y);
        int I23 = intersect(y, z);
        return I12 < I23;
    }
    void insert(int a, int b) {
        if(!dq.empty() && a == dq.back().a) return;
        while(dq.size() >= 2 &&
              !check(dq[dq.size() - 2], dq[dq.size() - 1],
                     {a, b})) {
            dq.pop_back();
        }
        dq.push_back({a, b});
    }
    void update(int x) {
        while(dq.size() >= 2 && dq[0].y(x) >= dq[1].y(x)) {
            dq.pop_front();
        }
    }
    int query(int x) {
        update(x);
        return dq.front().y(x);
    }
};
```

5.2. Li-Chao-SegmentTree.cpp

```cpp
struct line {
    int a, b = 1000000000000000;
    int y(int x) { return a * x + b; }
};

line tree[4000005];
int n, x;
int s[200005];
int f[200005];
int dp[200005];

void update(line ins, int l = 1, int r = 1e6, int index = 1) {
    if(l == r) {
        if(ins.y(l) < tree[index].y(l)) {
            tree[index] = ins;
        }
        return;
    }
    int mid = (l + r) >> 1;
    if(tree[index].a < ins.a) swap(tree[index], ins);
    if(tree[index].y(mid) > ins.y(mid)) {
        swap(tree[index], ins);
        update(ins, l, mid, index << 1);
    } else {
        update(ins, mid + 1, r, index << 1 | 1);
    }
}

int query(int x, int l = 1, int r = 1000000, int index = 1) {
    int cur = tree[index].y(x);
    if(l == r) {
        return cur;
    }
    int mid = (l + r) >> 1;
    if(x <= mid) {
        return min(cur, query(x, l, mid, index << 1));
    } else {
        return min(cur, query(x, mid + 1, r, index << 1 | 1));
    }
}
```

5.3. SOSDP.cpp

```cpp
for(int i = 0; i < 20; ++i)
    for(int j = i; j < N; ++j)
        if(j >> i & 1) dp[j] += dp[j ^ (1 << i)]; // subset
for(int i = 0; i < 20; ++i)
    for(int j = 0; j < N; ++j)
        if(!(j >> i & 1))
            dp2[j] += dp2[j | (1 << i)]; // superset
```

6. Geometry

6.1. 164253Version.cpp

```cpp

using namespace std;
#define ll long long
#define pb push_back
#define pll pair<int, int>
#define pdd pair<double, double>
#define pll pair<ll, ll>
#define F first
#define S second
#define eps 1e-6
int sign(double x) {
    return fabs(x) < eps ? 0 : x > 0 ? 1 : -1;
}
int sign(ll x) { return !x ? 0 : x > 0 ? 1 : -1; }
template <typename T1, typename T2>
istream &operator>>(istream &s, pair<T1, T2> &p) {
    auto &[a, b] = p;
    s >> a >> b;
    return s;
}
template <typename T1, typename T2>
ostream &operator<<(ostream &s, const pair<T1, T2> p) {
    auto &[a, b] = p;
    s << a << " " << b;
    return s;
}
pll operator+(const pll a, const pll b) {
    return {a.F + b.F, a.S + b.S};
}
pll operator-(const pll a, const pll b) {
    return {a.F - b.F, a.S - b.S};
}
pll operator-(const pll a) { return {-a.F, -a.S}; }
pll operator*(const pll a, const pll b) {
    return {(ll)a.F * b.F, (ll)a.S * b.S};
}
pdd operator/(const pll a, const double x) {
    return {a.F / x, a.S / x};
}
pdd operator*(const pll a, const double x) {
    return {a.F * x, a.S * x};
}
pdd operator*(const double x, const pll a) {
    return {a.F * x, a.S * x};
}
// 沒有標示幾個 vector 的都是對三個點做事，以第一個點為參考點
ll len2(pll p) {
    return (ll)p.F * p.F + (ll)p.S * p.S;
} // 1 vector
double len(pll p) { return sqrt((double)len2(p)); }
ll cross(pll a, pll b) {
    return (ll)a.F * b.S - (ll)a.S * b.F;
} // 2 vector
ll cross(pll p1, pll p2, pll p3) {
    return cross(p2 - p1, p3 - p1);
} //(b-a) cross (c-a)
ll dot(pll a, pll b, pll c) {
    return (ll)(b.F - a.F) * (c.F - a.F) +
           (ll)(b.S - a.S) * (c.S - a.S);
} //(b-a) dot (c-a)
ll ori(pll p1, pll p2, pll p3) {
    return sign(cross(p1, p2, p3));
} // normalize to {-1,0,1} (b-a) cross (c-a)
bool btw(pll p1, pll p2, pll p3) {
    return ori(p3, p1, p2) == 0 && dot(p3, p1, p2) <= 0;
} // p3 bwteen p1,p2
bool banana(pll p1, pll p2, pll p3,
            pll p4) { // 問兩線段是否香蕉
    if(btw(p1, p2, p3) || btw(p1, p2, p4) || btw(p3, p4, p1) ||
       btw(p3, p4, p2))
        return true;
    return ori(p1, p2, p3) * ori(p1, p2, p4) < 0 &&
           ori(p3, p4, p1) * ori(p3, p4, p2) < 0;
}
pdd banana_point(pll p1, pll p2, pll p3,
                 pll p4) { // 分點，算的是無限延伸直線的交點
                           // 平行的時候 undefined
    return cross(p2 - p1, p4 - p1) /
               (double)cross(p2 - p1, p4 - p3) * p3 -
           cross(p2 - p1, p3 - p1) /
               (double)cross(p2 - p1, p4 - p3) * p4;
}
pdd proj(pll p1, pll p2, pll p3) {
    return dot(p1, p2, p3) / (double)len2(p2 - p1) * (p2 - p1);
}
double min_dis(pll p1, pll p2,
               pll p3) { // min distance of p3 to segment p1,p2
    if(dot(p1, p2, p3) < 0 || dot(p2, p1, p3) < 0)
        return min(len(p3 - p1), len(p3 - p2));
    return abs(cross(p1, p2, p3)) / len(p2 - p1);
}
ll area2(vector<pll> &v) { // 傳入一個多邊形照順序的點集
                           // 起點要出現兩次，回傳兩倍面積
                           // 注意是兩倍才可以 ll 避免浮點數
    int n = v.size() - 1;
    ll ans = 0;
    for(int i = 0; i < n; ++i) ans += cross(v[i], v[i + 1]);
    return abs(ans);
}
int in_polygon(vector<pll> &v,
               pll p) { // 傳入多邊形，起點要出現兩次，回傳
                        //{-1:in, 0:on, 1:out}
    int n = v.size() - 1, ans = 1;
    for(int i = 0; i < n; ++i)
        if(btw(v[i], v[i + 1], p)) return 0;
    for(int i = 0; i < n; ++i)
        if(banana(v[i], v[i + 1], p, {(ll)2e9 + 7, p.S + 1LL}))
            ans *= -1;
    // 對於任意 p 到 {W, p.S+1}
    // 的向量中不會有整數點存在，其中需要滿足 {W, p.S+1}
    // 必須很遠，保證在多邊形外
    return ans;
}
void solve() {
    int n;
    cin >> n;
    vector<pll> v(n);
    for(pll &e : v) cin >> e;
    v.pb(v[0]);
    ll ans = area2(v) + 2, ans2 = 0;
    for(int i = 0; i < n; ++i) {
        if(v[i].F == v[i + 1].F)
            ans2 += abs(v[i].S - v[i + 1].S);
        else if(v[i].S == v[i + 1].S)
            ans2 += abs(v[i].F - v[i + 1].F);
        else
            ans2 += gcd(abs(v[i].F - v[i + 1].F),
                        abs(v[i].S - v[i + 1].S));
    }
    cout << (ans - ans2) / 2 << " " << ans2;
}
int main() {
    int t = 1;
    // cin>>t;
    for(; t--;) {
        solve();
    }
}
```

6.2. ConvexHull.cpp

```cpp

#define int long long
#define fastio                                                   \
    ios_base::sync_with_stdio(0);                                \
    cin.tie(0);                                                  \
    cout.tie(0);

using namespace std;

template <typename T>
pair<T, T> operator-(pair<T, T> a, pair<T, T> b) {
    return make_pair(a.first - b.first, a.second - b.second);
}

template <typename T> T cross(pair<T, T> a, pair<T, T> b) {
    return a.first * b.second - a.second * b.first;
}

template <typename T>
vector<pair<T, T>> getCH(vector<pair<T, T>> v) {
    int n = v.size();
    sort(v.begin(), v.end());
    vector<pair<T, T>> hull;
    for(int i = 0; i < 2; i++) {
        int t = hull.size();
        for(auto x : v) {
            while(hull.size() - t >= 2 &&
                  cross(hull[hull.size() - 1] -
                            hull[hull.size() - 2],
                        x - hull[hull.size() - 2]) <= 0)
                hull.pop_back();
            hull.push_back(x);
        }
        hull.pop_back();
        reverse(v.begin(), v.end());
    }
    return hull;
}
```

6.3. Inside.cpp

```cpp
int inside(point p) {
    int ans = 0;
    for(int i = 1; i <= n; i++) {
        if(onseg(a[i], a[i + 1], {p.x, p.y})) {
            return -1;
        }
        if(intersect({p.x, p.y}, {INF, p.y}, a[i], a[i + 1])) {
            ans ^= 1;
        }
        point temp = a[i].y > a[i + 1].y ? a[i] : a[i + 1];
        if(temp.y == p.y && temp.x > p.x) {
            ans ^= 1;
        }
    }
    return ans;
}
```

6.4. Intersect.cpp

```cpp
struct point {
    int x, y;
    point operator+(point b) { return {x + b.x, y + b.y}; }
    point operator-(point b) { return {x - b.x, y - b.y}; }
    int operator*(point b) { return x * b.x + y * b.y; }
    int operator^(point b) { return x * b.y - y * b.x; }
};

bool onseg(point x, point y, point z) {
    return ((x - z) ^ (y - z)) == 0 && (x - z) * (y - z) <= 0;
}

int dir(point x, point y) {
    int k = x ^ y;
    if(k == 0) return 0;
    if(k > 0) return 1;
    return -1;
}

bool intersect(point x, point y, point z, point w) {
    if(onseg(x, y, z) || onseg(x, y, w)) return 1;
    if(onseg(z, w, x) || onseg(z, w, y)) return 1;
    if(dir(y - x, z - x) * dir(y - x, w - x) == -1 &&
       dir(z - w, x - w) * dir(z - w, y - w) == -1) {
        return 1;
    }
    return 0;
}
```

6.5. MinimumEuclideanDistance.cpp

```cpp

#define int long long
#define pii pair<int, int>
using namespace std;

int n;
vector<pair<int, int>> v;
set<pair<int, int>> s;
int dd = LONG_LONG_MAX;

int dis(pii x, pii y) {
    return (x.first - y.first) * (x.first - y.first) +
           (x.second - y.second) * (x.second - y.second);
}

signed main() {
    ios::sync_with_stdio(0);
    cin.tie(0);
    cout.tie(0);
    cin >> n;
    for(int i = 0; i < n; i++) {
        int x, y;
        cin >> x >> y;
        x += 1000000000;
        v.push_back({x, y});
    }
    sort(v.begin(), v.end());
    int l = 0;
    for(int i = 0; i < n; i++) {
        int d = ceil(sqrt(dd));
        while(l < i && v[i].first - v[l].first > d) {
            s.erase({v[l].second, v[l].first});
            l++;
        }
        auto x = s.lower_bound({v[i].second - d, 0});
        auto y = s.upper_bound({v[i].second + d, 0});
        for(auto it = x; it != y; it++) {
            dd = min(dd, dis({it->second, it->first}, v[i]));
        }
        s.insert({v[i].second, v[i].first});
    }
    cout << dd;
}
```

7. Tree

7.1. HeavyLightDecomposition(modify-and-query-on-path).cpp

```cpp

#define int long long
using namespace std;

int tree[800005];

int n, q;
int a[200005];
int st[200005];
int tp[200005];
int p[200005];
int cnt = 0;
int d[200005];
int si[200005];
vector<int> v[200005];
int b[200005];

void build(int l = 1, int r = n, int index = 1) {
    if(l == r) {
        tree[index] = b[l];
        return;
    }
    int mid = (l + r) >> 1;
    build(l, mid, index << 1);
    build(mid + 1, r, index << 1 | 1);
    tree[index] = max(tree[index << 1], tree[index << 1 | 1]);
}

int query(int L, int R, int l = 1, int r = n, int index = 1) {
    if(L == l && r == R) {
        return tree[index];
    }
    int mid = (l + r) >> 1;
    if(R <= mid) {
        return query(L, R, l, mid, index << 1);
    }
    if(L > mid) {
        return query(L, R, mid + 1, r, index << 1 | 1);
    }
    return max(query(L, mid, l, mid, index << 1),
               query(mid + 1, R, mid + 1, r, index << 1 | 1));
}

void modify(int x, int val, int l = 1, int r = n,
            int index = 1) {
    if(l == r) {
        tree[index] = val;
        return;
    }
    int mid = (l + r) >> 1;
    if(x <= mid) {
        modify(x, val, l, mid, index << 1);
    } else {
        modify(x, val, mid + 1, r, index << 1 | 1);
    }
    tree[index] = max(tree[index << 1], tree[index << 1 | 1]);
}

void dfs(int x, int pre) {
    si[x] = 1;
    for(int i : v[x]) {
        if(i == pre) continue;
        p[i] = x;
        d[i] = d[x] + 1;
        dfs(i, x);
        si[x] += si[i];
    }
}

void dfs2(int x, int pre, int t) {
    tp[x] = t;
    st[x] = ++cnt;
    int ma = 0;
    for(int i : v[x]) {
        if(i == pre) continue;
        if(si[i] > si[ma]) {
            ma = i;
        }
    }
    if(!ma) return;
    dfs2(ma, x, t);
    for(int i : v[x]) {
        if(i == pre || i == ma) {
            continue;
        }
        dfs2(i, x, i);
    }
}

int f(int x, int y) {
    int ret = 0;
    while(tp[x] ^ tp[y]) {
        if(d[tp[x]] < d[tp[y]]) {
            swap(x, y);
        }
        ret = max(ret, query(st[tp[x]], st[x]));
        x = p[tp[x]];
    }
    if(d[x] > d[y]) swap(x, y);
    ret = max(ret, query(st[x], st[y]));
    return ret;
}

signed main() {
    ios::sync_with_stdio(0);
    cin.tie(0);
    cout.tie(0);
    cin >> n >> q;
    for(int i = 1; i <= n; i++) {
        cin >> a[i];
    }
    for(int i = 1; i < n; i++) {
        int x, y;
        cin >> x >> y;
        v[x].push_back(y);
        v[y].push_back(x);
    }
    dfs(1, 0);
    dfs2(1, 0, 1);
    for(int i = 1; i <= n; i++) {
        b[st[i]] = a[i];
    }
    build();
    while(q--) {
        int mode, x, y;
        cin >> mode >> x >> y;
        if(mode == 1) {
            modify(st[x], y);
        } else {
            cout << f(x, y) << " ";
        }
    }
}
```

7.2. LCA.cpp

```cpp

#define int long long
using namespace std;

int n, q;
int a[200005][21];
int d[200005];
vector<int> v[200005];

void init() {
    for(int j = 1; j < 21; j++) {
        for(int i = 1; i <= n; i++) {
            a[i][j] = a[a[i][j - 1]][j - 1];
        }
    }
}

void dfs(int x, int pre) {
    for(int i : v[x]) {
        if(i == pre) {
            continue;
        }
        a[i][0] = x;
        d[i] = d[x] + 1;
        dfs(i, x);
    }
}

int lca(int x, int y) {
    while(d[x] ^ d[y]) {
        if(d[x] < d[y]) {
            swap(x, y);
        }
        int k = __lg(d[x] - d[y]);
        x = a[x][k];
    }
    if(x == y) {
        return x;
    }
    for(int i = 20; i >= 0; i--) {
        if(a[x][i] != a[y][i]) {
            x = a[x][i];
            y = a[y][i];
        }
    }
    return a[x][0];
}

signed main() {
    ios::sync_with_stdio(0);
    cin.tie(0);
    cout.tie(0);
    cin >> n >> q;
    for(int i = 1; i < n; i++) {
        int x, y;
        cin >> x >> y;
        v[x].push_back(y);
        v[y].push_back(x);
    }
    dfs(1, 0);
    init();
    while(q--) {
        int x, y;
        cin >> x >> y;
        int k = lca(x, y);
        cout << (d[x] + d[y] - 2 * d[k]) << "\n";
    }
}
```

8. Misc

8.1. BigNum(luoguP1005).cpp

```cpp
// 洛谷 P1005

using namespace std;
#define N 85
#define LL long long
#define pii pair<int, int>
#define F first
#define S second
struct num {
    const static LL base = 1000000000LL; // base 1e9
    LL p[505], len;
    num() {
        memset(p, 0, sizeof(p));
        len = 0;
    }
    num(LL x) {
        memset(p, 0, sizeof(p));
        len = 0;
        for(p[len++] = x; p[len - 1] >= base; ++len)
            p[len] = p[len - 1] / base, p[len - 1] %= base;
    }
    num operator=(LL x) {
        memset(p, 0, sizeof(p));
        len = 0;
        for(p[len++] = x; p[len - 1] >= base; ++len)
            p[len] = p[len - 1] / base, p[len - 1] %= base;
        return *this;
    }
    num max(const num &b) {
        if(len != b.len) return len > b.len ? *this : b;
        for(int i = len; i--;)
            if(p[i] != b.p[i]) return p[i] > b.p[i] ? *this : b;
        return *this;
    }
    num operator+(const num &b) {
        num c;
        LL x = 0;
        for(LL &i = c.len; i < len || i < b.len; ++i) {
            c.p[i] = p[i] + b.p[i] + x;
            x = c.p[i] / base;
            c.p[i] %= base;
        }
        if(x) c.p[c.len++] = x;
        return c;
    }
    num operator*(LL b) {
        num c;
        c.len = len;
        LL x = 0;
        for(LL i = 0; i < len; ++i) {
            c.p[i] = p[i] * b + x;
            x = c.p[i] / base;
            c.p[i] %= base;
        }
        for(; x; x /= base) c.p[c.len++] = x % base;
        return c;
    }
} dp[N][N], ans;
ostream &operator<<(ostream &s, num a) {
    if(!a.len) return s << "0";
    s << a.p[a.len - 1];
    for(int i = a.len - 1; i--;) {
        if(!a.p[i])
            s << "000000000";
        else {
            for(int k = 10; k * a.p[i] < (LL)1e9; k *= 10)
                s << "0";
            s << a.p[i];
        }
    }
    return s;
}
LL a[N];
int main() {
    ios::sync_with_stdio(0);
    cin.tie(0);
    cout.tie(0);
    int n, m, i, j;
    for(cin >> n >> m; n--;) {
        for(i = 0; i < m; ++i) cin >> a[i];
        for(i = 0; i < m; ++i)
            for(j = 0; j < m; ++j) dp[i][j] = 0;
        for(i = 0; i < m; ++i) dp[i][i] = a[i] << 1;
        for(j = 1; j < m; ++j)
            for(i = 0; i + j < m; ++i)
                dp[i][i + j] =
                    (dp[i][i + j - 1] + a[i + j])
                        .max(dp[i + 1][i + j] + a[i]) *
                    2;
        ans = ans + dp[0][m - 1];
    }
    cout << ans;
}
```

8.2. Tri-search.cpp

```cpp

using namespace std;
int n;
double a[15], x, y;

double get(double x) {
    double ret = 0;
    double k = 1;
    for(int i = 0; i <= n; i++) {
        ret += k * a[i];
        k *= x;
    }
    return -ret;
}

template <class T> T bi_search(T l, T r, T end) {
    if(!check(r - end)) return r - end;
    for(; r - l > end;) {
        T mid = (l + r) / 2;
        if(check(mid))
            r = mid;
        else
            l = mid;
    }
    return l;
}
/*check gives 000000001111 find the last 0*/

template <class T> T tri_search(T l, T r, T end) {
    T midl, midr;
    for(;;) {
        midl = (l + r) / 2;
        midr = (midl + r) / 2;
        if(midr - midl < end) break;
        if(get(midr) > get(midl))
            r = midr;
        else
            l = midl;
    }
    for(; r - l > end;) {
        midl = (l + r) / 2;
        if(get(r) > get(l))
            r = midl;
        else
            l = midl;
    }
    return l;
}
/*get gives the value, find the minimum*/

int main() {
    cin >> n >> x >> y;
    for(int i = n; i >= 0; i--) {
        cin >> a[i];
    }
    cout << fixed << setprecision(7)
         << tri_search<double>(x, y, 1e-7);
}
```

9. AnotherVersionDataStructure

9.1. BIT.cpp

```cpp
template <class T> class BIT {
#define lb(x) ((x) & -(x))
#define N (int)2e5 + 5
  public:
    T bit[N] = {0};
    void update(T x, T v) {
        for(; x < N; x += lb(x)) bit[x] += v;
    }
    T qry(T x) {
        T ans = 0;
        for(; x; x -= lb(x)) ans += bit[x];
        return ans;
    }
#undef lb
#undef N
};
/*1based bit update 預設是加值 */
```

9.2. DSU.cpp

```cpp
template <class T> class Dsu {
#define N 2000005
  public:
    T dsu[N], size[N];
    Dsu(T n) {
        for(; n; --n) dsu[n] = n, size[n] = 1;
    }
    T qry(T x) {
        if(dsu[x] == x) return x;
        return dsu[x] = qry(dsu[x]);
    }
    void merge(T a, T b) {
        a = qry(a);
        b = qry(b);
        if(a == b) return;
        if(size[a] < size[b])
            dsu[a] = b, size[b] += size[a];
        else
            dsu[b] = a, size[a] += size[b];
    }
#undef N
};
/*1based 初始化為 dsu[x]=x 路徑壓縮 + 啟發式合併 */
```

9.3. Treap.cpp

```cpp
// treap 模板 洛谷 P3369 【模板】普通平衡树

using namespace std;
#define pnn pair<node *, node *>
#define F first
#define S second
mt19937 mt(hash<string>()("official_beautiful_fruit"));
struct node {
    node *l, *r;
    int val, sz;
    int mx, mn, sum;
    int rev_tag, add_tag;
    node(int x)
        : val(x), l(0), r(0), sz(1), rev_tag(0), add_tag(0),
          mx(x), mn(x), sum(x) {}
    node(node *tr)
        : val(tr->val), l(tr->l), r(tr->r), sz(tr->sz),
          rev_tag(tr->rev_tag), add_tag(tr->add_tag),
          mx(tr->mx), mn(tr->mn) {}
    void pull() {
        sz = 1;
        mx = mn = sum = val;
        if(l)
            sz += l->sz, mx = max(mx, l->mx),
                         mn = min(mn, l->mn), sum += l->sum;
        if(r)
            sz += r->sz, mx = max(mx, r->mx),
                         mn = min(mn, r->mn), sum += r->sum;
    }
    void push() {
        if(rev_tag) swap(l, r);
        if(l) l->add_tag += add_tag, l->rev_tag ^= rev_tag;
        if(r) r->add_tag += add_tag, r->rev_tag ^= rev_tag;
        mx += add_tag;
        mn += add_tag;
        sum += add_tag;
        add_tag = 0;
        rev_tag = 0;
    }
};
void debug(node *tr) {
    if(!tr) return;
    tr->push();
    tr->pull();
    debug(tr->l);
    cout << tr->val << " ";
    debug(tr->r);
}
void debug2(node *tr) {
    if(!tr) return;
    tr->push();
    tr->pull();
    cout << tr->val << " ";
    debug2(tr->l);
    debug2(tr->r);
}
int sz(node *tr) { return tr ? tr->sz : 0; }
node *merge(node *a, node *b) {
    if(!a || !b) return a ?: b;
    a->push();
    b->push();
    if(mt() % (sz(a) + sz(b)) < sz(a)) {
        a->r = merge(a->r, b);
        a->pull();
        return a;
    }
    b->l = merge(a, b->l);
    b->pull();
    return b;
}
pnn split(node *tr, int v) { //(-inf,v],(v,inf)
    if(!tr) return {0, 0};
    tr->push();
    if(tr->val <= v) {
        auto [l, r] = split(tr->r, v);
        tr->r = l;
        tr->pull();
        return {tr, r};
    }
    auto [l, r] = split(tr->l, v);
    tr->l = r;
    tr->pull();
    return {l, tr};
}
pnn splitsz(node *tr, int k) { //[rk.1,rk.k],(rk.k,rk.n]
    if(!tr || sz(tr) <= k) return {tr, 0};
    tr->push();
    if(k <= sz(tr->l)) {
        auto [l, r] = splitsz(tr->l, k);
        tr->l = r;
        tr->pull();
        return {l, tr};
    } else if(k <= sz(tr->l) + 1) {
        auto r = tr->r;
        tr->r = 0;
        tr->pull();
        return {tr, r};
    } else {
        auto [l, r] = splitsz(tr->r, k - (sz(tr->l) + 1));
        tr->r = l;
        tr->pull();
        return {tr, r};
    }
}
node *insert(node *tr, int v) {
    auto [l, r] = split(tr, v);
    return merge(merge(l, new node(v)), r);
}
node *insertkth(node *tr, int k) {
    auto [l, r] = splitsz(tr, k - 1);
    return merge(merge(l, new node(0)),
                 r); // new node 拿來區間操作初始化
}
node *eraseall(node *tr, int v) {
    auto [l, r] = split(tr, v - 1);
    return merge(l, split(r, v).S);
}
node *eraseone(node *tr, int v) {
    auto [l, r] = split(tr, v - 1);
    return merge(l, splitsz(r, 1).S);
}
node *erasekth(node *tr, int k) {
    auto [l, r] = splitsz(tr, k - 1);
    return merge(l, splitsz(r, k).S);
}
int rnk(node *tr, int v) {
    if(!tr) return 0;
    if(tr->val <= v) return sz(tr->l) + 1 + rnk(tr->r, v);
    return rnk(tr->l, v);
}
int kth(node *&tr, int k) {
    auto [l, x] = splitsz(tr, k - 1);
    auto [m, r] = splitsz(x, 1);
    if(!m) return 0;
    int ans = m->val;
    tr = merge(merge(l, m), r);
    return ans;
}
int count(node *&tr, int L, int R) { // count[L,R]
    auto [l, x] = split(tr, L - 1);
    auto [m, r] = split(x, R);
    int ans = m->sz; // 看要改啥
    tr = merge(merge(l, m), r);
    return ans;
}
int countkth(node *&tr, int L, int R) { // count[rk.L,rk.R]
    auto [l, x] = splitsz(tr, L - 1);
    auto [m, r] = splitsz(x, R - L);
    int ans = m->sum; // 看要改啥
    tr = merge(merge(l, m), r);
    return ans;
}
int prev(node *&tr, int v) {
    auto [x, r] = split(tr, v - 1);
    auto [l, m] = splitsz(x, sz(x) - 1);
    int ans = m->val;
    tr = merge(merge(l, m), r);
    return ans;
}
int next(node *&tr, int v) {
    auto [l, x] = split(tr, v);
    auto [m, r] = splitsz(x, 1);
    int ans = m->val;
    tr = merge(merge(l, m), r);
    return ans;
}
int qry(node *&tr, int L, int R) { // qry[L,R]
    auto [x, r] = splitsz(tr, R);
    auto [l, m] = splitsz(x, L - 1);
    int ans = m->sum; // 看要改啥
    tr = merge(merge(l, m), r);
    return ans;
}
void modify(node *&tr, int L, int R, int v) { // modify[L,R]
    auto [x, r] = splitsz(tr, R);
    auto [l, m] = splitsz(x, L - 1);
    m->val += v;
    m->add_tag += v;
    m->rev_tag = 1; // 看要改啥
    tr = merge(merge(l, m), r);
}
int main() {
    int t;
    node *tr = 0;
    for(cin >> t; t--;) {
        int op, x;
        cin >> op >> x;
        switch(op) {
        case 1:
            tr = insert(tr, x);
            break;
        case 2:
            tr = eraseone(tr, x);
            break;
        case 3:
            cout << rnk(tr, x - 1) + 1 << "\n";
            break;
        case 4:
            cout << kth(tr, x) << "\n";
            break;
        case 5:
            cout << prev(tr, x) << "\n";
            break;
        case 6:
            cout << next(tr, x) << "\n";
            break;
        }
    }
}
```

9.4. Treap 但可以多個數縮點 (疑似爛的).cpp

```
// treap 模板 洛谷 P3369 【模板】普通平衡树

using namespace std;
#define pnn pair<node *, node *>
#define F first
#define S second
#define int long long
mt19937 mt(hash<string>()("official_beautiful_fruit"));
struct node {
    node *l, *r;
    int val, sz;
    int mx, mn, sum, num;
    int rev_tag, add_tag;
    node(int _val = 0, int _num = 1)
        : val(_val), l(0), r(0), sz(1), sum(_num), num(_num),
          mx(_val), mn(_val), rev_tag(0), add_tag(0) {}
    node(node *tr)
        : val(tr->val), l(tr->l), r(tr->r), sz(tr->sz) {}
    void pull() {
        sz = 1;
        mx = mn = sum = num;
        if(l)
            sz += l->sz, mx = max(mx, l->mx),
                         mn = min(mn, l->mn), sum += l->sum;
        if(r)
            sz += r->sz, mx = max(mx, r->mx),
                         mn = min(mn, r->mn), sum += r->sum;
    }
    void push() {
        if(rev_tag) swap(l, r);
        if(l) l->add_tag += add_tag, l->rev_tag ^= rev_tag;
        if(r) r->add_tag += add_tag, r->rev_tag ^= rev_tag;
        mx += add_tag;
        mn += add_tag;
        sum += add_tag;
        add_tag = 0;
        rev_tag = 0;
    }
};
void debug(node *tr) {
    if(!tr) return;
    debug(tr->l);
    cout << tr->val << " ";
    debug(tr->r);
}
void debug2(node *tr) {
    if(!tr) return;
    cout << tr->val << " ";
    debug2(tr->l);
    debug2(tr->r);
}
int sz(node *tr) { return tr ? tr->sz : 0; }
node *merge(node *a, node *b) {
    if(!a || !b) return a ?: b;
    if(mt() % (sz(a) + sz(b)) < sz(a)) {
        a->r = merge(a->r, b);
        a->pull();
        return a;
    }
    b->l = merge(a, b->l);
    b->pull();
    return b;
}
pnn split(node *tr, int v) { //(-inf,v],(v,inf)
    if(!tr) return {0, 0};
    tr->push();
    if(tr->val <= v) {
        auto [l, r] = split(tr->r, v);
        tr->r = l;
        tr->pull();
        return {tr, r};
    }
    auto [l, r] = split(tr->l, v);
    tr->l = r;
    tr->pull();
    return {l, tr};
}
pnn splitsz(node *tr, int k) { //[rk.1,rk.k],(rk.k,rk.n]
    if(!tr || sz(tr) <= k) return {tr, 0};
    tr->push();
    if(k <= sz(tr->l)) {
        auto [l, r] = splitsz(tr->l, k);
        tr->l = r;
        tr->pull();
        return {l, tr};
    } else if(k <= sz(tr->l) + 1) {
        auto r = tr->r;
        tr->r = 0;
        tr->pull();
        return {tr, r};
    } else {
        auto [l, r] = splitsz(tr->r, k - (sz(tr->l) + 1));
        tr->r = l;
        tr->pull();
        return {tr, r};
    }
}
node *insert(node *tr, int val = 0, int num = 1) {
    auto [l, r] = split(tr, val);
    return merge(merge(l, new node(val, num)), r);
}
node *insertkth(node *tr, int k) {
    auto [l, r] = splitsz(tr, k - 1);
    return merge(merge(l, new node()),
                 r); // new node 拿來區間操作初始化
}
node *eraseall(node *tr, int v) {
    auto [l, r] = split(tr, v - 1);
    return merge(l, split(r, v).S);
}
node *eraseone(node *tr, int v) {
    auto [l, r] = split(tr, v - 1);
    return merge(l, splitsz(r, 1).S);
}
node *erasekth(node *tr, int k) {
    auto [l, r] = splitsz(tr, k - 1);
    return merge(l, splitsz(r, k).S);
}
int rnk(node *tr, int v) {
    if(!tr) return 0;
    if(tr->val <= v) return sz(tr->l) + 1 + rnk(tr->r, v);
    return rnk(tr->l, v);
}
int kth(node *&tr, int k) {
    auto [l, x] = splitsz(tr, k - 1);
    auto [m, r] = splitsz(x, 1);
    if(!m) return 0;
    int ans = m->val;
    tr = merge(merge(l, m), r);
    return ans;
}
int count(node *&tr, int L, int R) { // count[L,R]
    auto [l, x] = split(tr, L - 1);
    auto [m, r] = split(x, R);
    int ans = m->sum; // 看要改啥
    tr = merge(merge(l, m), r);
    return ans;
}
int countkth(node *&tr, int L, int R) { // count[rk.L,rk.R]
    auto [l, x] = splitsz(tr, L - 1);
    auto [m, r] = splitsz(x, R - L);
    int ans = m->sum; // 看要改啥
    tr = merge(merge(l, m), r);
    return ans;
}
int prev(node *&tr, int v) {
    auto [x, r] = split(tr, v - 1);
    auto [l, m] = splitsz(x, sz(x) - 1);
    int ans = m->val;
    tr = merge(merge(l, m), r);
    return ans;
}
int next(node *&tr, int v) {
    auto [l, x] = split(tr, v);
    auto [m, r] = splitsz(x, 1);
    int ans = m->val;
    tr = merge(merge(l, m), r);
    return ans;
}
int qry(node *&tr, int L, int R) { // qry[L,R]
    auto [l, x] = splitsz(tr, L - 1);
    auto [m, r] = splitsz(x, R);
    int ans = m->sum; // 看要改啥
    tr = merge(merge(l, m), r);
    return ans;
}
void modify(node *&tr, int L, int R, int v) { // modify[L,R]
    auto [l, x] = splitsz(tr, L - 1);
    auto [m, r] = splitsz(x, R);
    m->val += v;
    m->add_tag += v; // 看要改啥
    tr = merge(merge(l, m), r);
}
signed main() {
    vector<node *> tr(2);
    int n, m;
    scanf("%lld%lld", &n, &m);
    for(int i = 1, x; i <= n; ++i)
        scanf("%lld", &x), (x) && (tr[1] = insert(tr[1], i, x));
    for(; m--;) {
```

```cpp
        int op = -1, p = -1, x = -1, y = -1;
        scanf("%lld", &op);
        if(!op) {
            scanf("%lld%lld%lld", &p, &x, &y);
            auto [l, tmp] = split(tr[p], x - 1);
            auto [m, r] = split(tmp, y);
            tr[p] = merge(l, r);
            tr.push_back(m);
        } else if(op == 1) {
            scanf("%lld%lld", &p, &x);
            // cout<<kth(tr[x],1)<<"\n";//break;
            auto [l, r] = split(tr[p], kth(tr[x], 1));
            tr[p] = merge(merge(l, tr[x]), r);
        } else
            switch(op) {
            case 2:
                scanf("%lld%lld%lld", &p, &x, &y);
                tr[p] = insert(tr[p], y, x);
                break;
            case 3:
                scanf("%lld%lld%lld", &p, &x, &y);
                printf("%lld\n", count(tr[p], x, y));
                break;
            case 4:
                scanf("%lld%lld", &p, &x);
                printf("%lld\n", kth(tr[p], x));
                break;
            }
    }
}
```

9.5. 區間插線段單點查詢李超 (是爛的).cpp

```cpp
// luogu P4097 區間插線段李超

using namespace std;
#define N 50005
struct Line {
    double a, b;
    int l, r, id; // ax+b{l<=x<=r}
    Line(double _a = -1e6, double _b = -1, int _l = 1,
         int _r = N, int _id = 0)
        : a(_a), b(_b), l(_l), r(_r), id(_id) {}
    double operator()(int x) { return a * x + b; }
} line[N];
int seg[N << 2];
#define lid (id << 1)
#define rid (id << 1 | 1)
#define M (L + R >> 1)
#define eps 1e-6
void ins(int l, int L = 1, int R = N, int id = 1) {
    // cout<<"ins{"<<line[l].a<<","<<line[l].b<<","<<line[l].l<<","<<line[l].r<<"}"<<L<<"
    // "<<R<<"\n";
    if(line[l].r < L || R < line[l].l) return;
    if(L == R) {
        if(line[l](M) - line[seg[id]](M) > eps) seg[id] = l;
        return;
    }
    if(line[l].l <= M && M <= line[l].r &&
       line[l](M) - line[seg[id]](M) > eps)
        swap(l, seg[id]);
    if(line[l].l <= L && R <= line[l].r) {
        if(line[l].a - line[seg[id]].a > eps)
            ins(l, M + 1, R, rid);
        else
            ins(l, L, M, lid);
    }
    /*if(line[l].a>line[seg[id]].a)*/ ins(l, M + 1, R, rid);
    /*else */ ins(l, L, M, lid);
}
int qry(int x, int L = 1, int R = N, int id = 1) {
    // cout<<"qry"<<x<<"{"<<line[seg[id]].a<<","<<line[seg[id]].b<<"}"<<L<<"
    // "<<R<<" "<<id<<"\n";
    if(L == R) return seg[id];
    int k = (x <= M ? qry(x, L, M, lid)
                    : qry(x, M + 1, R, rid)),
        not_k = 0, not_seg = 0;
    if(line[k].r < x || x < line[k].l) not_k = 1;
    if(line[seg[id]].r < x || x < line[seg[id]].l) not_seg = 1;
    if(not_k && not_seg) return 0;
    if(not_k) return seg[id];
    if(not_seg) return k;
    return line[k](x) - line[seg[id]](x) > eps ? k : seg[id];
}
int main() {
    int n, ans = 0, p = 1;
    for(cin >> n; n--;) {
        int op;
        cin >> op;
        if(op) {
            int x0, y0, x1, y1;
            cin >> x0 >> y0 >> x1 >> y1;
            x0 = (x0 + ans - 1) % 39989 + 1;
            y0 = (y0 + ans - 1) % 1000000000 + 1;
            x1 = (x1 + ans - 1) % 39989 + 1;
            y1 = (y1 + ans - 1) % 1000000000 + 1;
            if(x0 > x1) swap(x0, x1), swap(y0, y1);
            // cout<<"?"<<((double)y1-y0)/(x1-x0)<<"
            // "<<y0-x0*((double)y1-y0)/(x1-x0)<<"\n";
            if(x0 != x1)
                line[p] = Line(((double)y1 - y0) / (x1 - x0),
                               y0 - x0 * ((double)y1 - y0) /
                                        (x1 - x0),
                               x0, x1, p);
            else
                line[p] = Line(0, max(y0, y1), x0, x1, p);
            ins(p);
            ++p;
        } else {
            int k;
            cin >> k;
            k = (k + ans - 1) % 39989 + 1;
            cout << (ans = qry(k)) << "\n";
        }
    }
    // cout<<qry(9)<<"\n";
}
```

9.6. 單點修改動態開點線段樹.cpp

```cpp

using namespace std;
#define N 200005
#define M int m = l + r >> 1
#define MAX 1000000000
int a[N];
typedef struct node {
    struct node *l, *r;
    int val;
};
void check(node *tree, int flag) {
    if(flag && !tree->r)
        tree->r = (node *)malloc(sizeof(struct node)),
        tree->r->val = 0;
    else if(!flag && !tree->l)
        tree->l = (node *)malloc(sizeof(struct node)),
        tree->l->val = 0;
}
void upd(int pos, int val, int l, int r, node *tree) {
    tree->val += val;
    if(l == r) return;
    M;
    if(pos > m)
        check(tree, 1), upd(pos, val, m + 1, r, tree->r);
    else
        check(tree, 0), upd(pos, val, l, m, tree->l);
}
int qry(int a, int b, int l, int r, node *tree) {
    if(!tree) return 0;
    if(a <= l && r <= b) return tree->val;
    M;
    if(a > m) return qry(a, b, m + 1, r, tree->r);
    if(b <= m) return qry(a, b, l, m, tree->l);
    return qry(a, b, m + 1, r, tree->r) +
           qry(a, b, l, m, tree->l);
}
int main() {
    int n, q, i = 1, x;
    node *root = (node *)malloc(sizeof(struct node));
    root->val = 0;
    for(scanf("%d %d", &n, &q); i <= n; ++i)
        getchar(), scanf("%d", a + i),
            upd(a[i], 1, 1, MAX, root);
    // printf("%d %d %d %d
    // %d\n",qry(2,2,1,n,1),qry(3,3,1,n,1),qry(5,5,1,n,1),qry(5,5,1
    for(; q--;) {
        getchar();
        char c = getchar();
        scanf(" %d %d", &x, &i);
        if(c == '!')
            upd(a[x], -1, 1, MAX, root),
                a[x] = i, upd(i, 1, 1, MAX, root);
        else
            printf("%d\n", qry(x, i, 1, MAX, root));
    }
}
```

9.7. 單點修改無懶標線段樹.cpp

```cpp
template <class T> class Seg {
#define lid id << 1
#define rid id << 1 | 1
#define M (L + R >> 1)
#define N 200005
  public:
    T a[N], seg[N << 2];
    Seg() {
        for(int i = 1; i <= n; ++i) cin >> a[i];
        init();
    }
    T update(int pos, int val, int L = 1, int R = n,
             int id = 1) {
        if(L == R) return seg[id] = val;
        if(pos > M)
            return seg[id] = seg[lid] +
                             update(pos, val, M + 1, R, rid);
        return seg[id] = update(pos, val, L, M, lid) + seg[rid];
    }
    T qry(int l, int r, int L = 1, int R = n, int id = 1) {
        if(l <= L && R <= r) return seg[id];
        if(L == R) return seg[id];
        int M = L + R >> 1;
        if(l > M) return qry(l, r, M + 1, R, rid);
        if(r <= M) return qry(l, r, L, M, lid);
        return qry(l, M, L, M, lid) +
               qry(M + 1, r, M + 1, R, rid);
    }

  private:
    T init(int l = 1, int r = n, int id = 1) {
        if(l == r) return seg[id] = a[l];
        int m = l + r >> 1;
        return seg[id] = init(l, m, lid) + init(m + 1, r, rid);
    }
#undef lid
#undef rid
#undef N
};
/*1based 陣列 1based id 單點修改 預設維護區間和 */
```

9.8. 懶標線段樹.cpp

```
struct Seg {
#define lid (id << 1)
#define rid ((id << 1) | 1)
#define M (L + R >> 1)
#define N 200005
    LL seg[N << 2], tag[N << 2];
    void inline addtag(int id, LL v, int L, int R) {
        seg[id] += v * (R - L + 1);
        tag[id] += v;
    }
    void inline push(int id, int L, int R) {
        addtag(lid, tag[id], L, M);
        addtag(rid, tag[id], M + 1, R);
        tag[id] = 0;
    }
    void inline pull(int id) { seg[id] = seg[lid] + seg[rid]; }
    void init(int L = 1, int R = n, int id = 1) {
        if(L == R) {
            seg[id] = 0;
            tag[id] = 0;
            return;
        }
        init(L, M, lid);
        init(M + 1, R, rid);
        pull(id);
    }
    void upd(int l, int r, LL v, int L = 1, int R = n,
             LL id = 1) {
        if(l <= L && R <= r) {
            addtag(id, v, L, R);
            return;
        }
        push(id, L, R);
        if(r <= M)
            upd(l, r, v, L, M, lid);
        else if(M + 1 <= l)
            upd(l, r, v, M + 1, R, rid);
        else
            upd(l, M, v, L, M, lid),
                upd(M + 1, r, v, M + 1, R, rid);
        pull(id);
    }
    LL qry(int l, int r, int L = 1, int R = n, int id = 1) {
        if(l <= L && R <= r) return seg[id];
        push(id, L, R);
        if(r <= M) return qry(l, r, L, M, lid);
        if(M + 1 <= l) return qry(l, r, M + 1, R, rid);
        return qry(l, M, L, M, lid) +
               qry(M + 1, r, M + 1, R, rid);
    }
} seg;
/*1based 陣列 1based id 區間修改 預設維護區間和 */
```

9.9. 純直線單點查詢李超.cpp

```
// luogu P4254 李超

using namespace std;
#define N 50005
struct Line {
    double a, b; // ax+b
    Line(double _a = -1, double _b = -1e6)
        : a(_a), b(_b - _a) {}
    double operator()(int x) { return a * x + b; }
} seg[N << 2];
#define lid (id << 1)
#define rid (id << 1 | 1)
#define M (L + R >> 1)
void ins(Line l, int L = 1, int R = N, int id = 1) {
    if(L == R) {
        if(seg[id].a < 0 || l(M) > seg[id](M)) seg[id] = l;
        return;
    }
    if(l(M) > seg[id](M)) swap(l, seg[id]);
    if(l.a > seg[id].a)
        ins(l, M + 1, R, rid);
    else
        ins(l, L, M, lid);
}
double qry(int x, int L = 1, int R = N, int id = 1) {
    if(L == R) return seg[id](x);
    if(x <= M) return max(qry(x, L, M, lid), seg[id](x));
    return max(seg[id](x), qry(x, M + 1, R, rid));
}
int main() {
    int n;
    for(cin >> n; n--;) {
        string s;
        cin >> s;
        if(s[0] == 'Q') {
            int x;
            cin >> x;
            cout << max(0, ((int)(qry(x) * 100)) / 10000)
                 << "\n";
        } else {
            double s, p;
            cin >> s >> p;
            ins(Line(p, s));
        }
    }
}
```

10. AnotherVersionMath

10.1. CRT(luoguVersion).cpp

```
long long CRT(long long *W, long long *B,
              long long k /* 方程组数 */) {
    long long x, y, a = 0, m, n = 1;
    for(long long i = 0; i < k; i++) n *= W[i];
    for(long long i = 0; i < k; i++) {
        m = n / W[i];
        ext_gcd(W[i], m, x, y);
        a = (a + y * m * B[i]) % n;
    }
    return a > 0 ? a : a + n;
}
```

10.2. PollardRho.cpp

```

using namespace std;
#define LL long long
#define uLL __uint128_t
#define sub(a, b) ((a) < (b) ? (b) - (a) : (a) - (b))
template <class T, class POW>
void fastpow(T x, POW n, POW p, T &ans) {
    for(; n; n >>= 1) {
        if(n & 1) {
            ans *= x;
            ans %= p;
        }
        x *= x;
        x %= p;
    }
}
/* 輸入 x,n,p,ans 會將 ans 修改為 x^n%p
對整數/矩陣/不要求精度的浮點 皆有效
模板第一個型別是 x,ans 第二個是 n,p(應該放 LL 或 __int128)*/
uLL pri[7] = {2,      325,     9375,      28178,
              450775, 9780504, 1795265022}; /*2^64*/
// int p[3]={2,7,61};/*2^32*/
bool check(const uLL x, const uLL p) {
    uLL d = x - 1, ans = 1;
    fastpow(p, d, x, ans);
    if(ans != 1) return 1;
    for(; !(d & 1);) {
```

```cpp
        d >>= 1;
        ans = 1;
        fastpow(p, d, x, ans);
        if(ans == x - 1)
            return 0;
        else if(ans != 1)
            return 1;
    }
    return 0;
}
bool miller_rabin(const uLL x) {
    if(x == 1) return 0;
    for(auto e : pri) {
        if(e >= x) return 1;
        if(check(x, e)) return 0;
    }
    return 1;
}
template <class T> T gcd(T a, T b) {
    if(!a) return b;
    if(!b) return a;
    if(a & b & 1) return gcd(sub(a, b), min(a, b));
    if(a & 1) return gcd(a, b >> 1);
    if(b & 1) return gcd(a >> 1, b);
    return gcd(a >> 1, b >> 1) << 1;
}
/*gcd(a,b) 默認 gcd(a,0)=a*/
mt19937 rnd(time(0));
template <class T> T f(T x, T c, T mod) {
    return (((uLL)x) * x % mod + c) % mod;
}
template <class T> T rho(T n) {
    T mod = n, x = rnd() % mod, c = rnd() % (mod - 1) + 1,
      p = 1;
    for(T i = 2, j = 2, d = x;; ++i) {
        x = f(x, c, mod), p = ((uLL)p) * sub(x, d) % mod;
        if(i % 127 == 0 && gcd(p, n) != 1) return gcd(p, n);
        if(i == j) {
            j <<= 1, d = x;
            if(gcd(p, n) != 1) return gcd(p, n);
        }
    }
}
template <class T> T pollard_rho(T n) {
    if(miller_rabin(n)) return n;
    T p = n;
    while(p == n) p = rho(n);
    return max(pollard_rho(p), pollard_rho(n / p));
}
int main() {
    LL t, n, ans;
    for(cin >> t; t--;) {
        cin >> n;
        ans = pollard_rho(n);
        if(ans == n)
            puts("Prime");
        else
            printf("%lld\n", ans);
    }
}
```

10.3. 快速冪.cpp

```cpp
template <class T, class POW>
void fastpow(T x, POW n, POW p, T &ans) {
    for(; n; n >>= 1) {
        if(n & 1) {
            ans *= x;
            ans %= p;
        }
        x *= x;
        x %= p;
    }
}
/* 輸入 x,n,p,ans 會將 ans 修改為 x^n%p
對整數/矩陣/不要求精度的浮點 皆有效
模板第一個型別是 x,ans 第二個是 n,p(應該放 LL 或 __int128)*/
```

10.4. 數論.cpp

```cpp
template <class T> T extgcd(T a, T b, T &x, T &y) {
    if(!b) {
        x = 1;
        y = 0;
        return a;
    }
    T ans = extgcd(b, a % b, y, x);
    y -= a / b * x;
    return ans;
}
/*extgcd(a,b,x,y)=ax+by,x 跟 y 是會被修改的參數 */
template <class T> T modeq(T a, T b, T p) {
    T x, y, d = extgcd(a, p, x, y);
    if(b % d) return 0;
    return ((b / d * x) % p + p) % p;
}
/*x=modeq(a,b,n),ax=b(mod n),0<=x<n
modeq(a,1,n) 相當於求 a 在 mod n 下的逆元 */
template <class T> T gcd(T a, T b) {
    if(!a) return b;
    if(!b) return a;
    if(a & b & 1) return gcd(abs(a - b), min(a, b));
    if(a & 1) return gcd(a, b >> 1);
    if(b & 1) return gcd(a >> 1, b);
    return gcd(a >> 1, b >> 1) << 1;
}
/*gcd(a,b) 默認 gcd(a,0)=a*/
ll crt(V<ll> &p, V<ll> &a) {
    ll n = 1, ans = 0, k = a.size();
    for(ll &e : p) n *= e;
    for(int i = 0; i < k; ++i)
        ans = (ans + a[i] * n / p[i] % n *
                         modeq(n / p[i], 1LL, p[i]) % n) %
              n;
    return (ans % n + n) % n;
}
/*$(a+b)^p \equiv a+b \equiv a^p+b^p \pmod p$ (小費馬)
$(p-1)! \equiv -1 \pmod p$ (威爾遜定理)
$v(n) :=$ $n$中$p$的冪次, $(n)_p := \frac{n}{p^{v(n)}}$,
$s(n) :=$ $p$進制下$n$的所有位數和
$v(n!) = \sum_{i=1}^{\infty} \lfloor$
$\frac{n}{p^i} \rfloor = \frac{n-s(n)}{p-1}$ (勒壤得定理)
$v(\binom{n}{m}) = \frac{s(n)+s(m-n)-s(m)}{p-1}$ (庫默爾定理)
$v(\binom{n}{m1,m2,\ldots mk}) =$
$\frac{\sum_{i=1}^{k} s(mi)-s(n)}{p-1}$(庫默爾定理推廣)
\[
    (n!)_p\equiv-1^{\lfloor\frac{n}{p}\rfloor}
\]
\[
    ((\lfloor\frac{n}{p}\rfloor)!)_p((n\%p)!)\pmod p
\]
打階乘表 + 迭代這條式子可以 $O(p+log_p(n))$ (mod 下階乘)
$\binom{n}{m} \equiv \frac{((n+m)!)_p}{(n!)_p(m!)_p}$
$p^{v(n+m)-v(n)-v(m)} \pmod{p^q}$
把 $p$ 從 $C(n,m)$ 裡面隔離掉了 就能用上面的
$(n!)_p$+ 模逆元 (mod 下階乘推廣至二項式)
$((p^q)!)_p \equiv \pm 1 \pmod p^q$ (威爾遜定理推廣)
\[
    \binom{n}{m}\equiv\binom{\lfloor\frac{n}{p}\rfloor}
\]
$\lfloor\frac{m}{p}\rfloor\binom{n\%p}{m\%p} \pmod p$
(lucas 定理) 打階乘表跟模逆元表 + 迭代這條式子可以 $O(p+log_p(n))$
若 $p$ 進制下任何一位 $i$ 滿足 $n_i < m_i$ 則
$\binom{n_i}{m_i}\%p = 0$
則因 $\binom{n}{m} = \prod_{i=0}^{\max(\log_p(a),\log_p(b))}$
$\binom{n_i}{m_i}\%p$ 導致 $\binom{n}{m}\%p = 0$
設 $p = 2$ 則有 $\binom{n}{m}$ 是奇數的充要條件為二進制下每一位
$n < m$ (lucas 定理額外性質) lucas 定理可由此生成函數做法得到
不依賴小費馬 對多項式也成立 根據上述
$\binom{n}{m}\%k$ 可將 $k$ 做唯一質數分解
個別做完再做 crt 得到結果 (exlucas 定理)
\[
    卡特蘭數 C(0)=C(1)=1,n>1 時 C(n)=\sum_{k=0}^{n-1}C(k)C(n-1-k)=
\]
$\frac{\binom{2n}{n}}{n+1}$
同時 $n$ 對括號的合法放置數即是 $C(n)$ 若有任意 k 種括號可選 則
$C(n)k^n$
模逆元表 p=i*(p/i)+p%i,-p%i=i*(p/i),inv(i)=-(p/i)*inv(p%i)*/
LL fracp[N], invp[N];
void fracp_init(LL p) {
    fracp[0] = 1;
    for(int i = 1; i < p; ++i) fracp[i] = fracp[i - 1] * i % p;
}
void invp_init(LL p) {
    invp[0] = invp[1] = 1;
    for(int i = 2; i < p; ++i)
        invp[i] = p - (p / i * invp[p % i]) % p;
}
/* 階乘表跟模逆元表 之後可以考慮改一下長相 */
template <class T> T lucas(T n, T m, T p) {
    if(!m) return 1;
    if(m > n || m % p > n % p) return 0;
    return lucas(n / p, m / p, p) * fracp[n % p] % p *
           invp[fracp[n % p - m % p]] % p * invp[fracp[m % p]] %
           p;
}
/*lucas(n,m,p)=C(n,m)%p 要求要帶階乘表跟模逆元表
 * O(p+log_p(n))*/
/* 米勒拉賓質數 2,325,9375,28178,450775,9780504,1795265022*/
/*crt 質數
(2^16)+1 65537 3
```

```
7*17*(2^23)+1 998244353 3
1255*(2^20)+1 1315962881 3
51*(2^25)+1 1711276033 29
*/
```

10.5. 篩法.cpp

```
// 待加入分塊篩
template <class T> class Prime {
#define N (int)1e8 + 9
  public:
    vector<T> list, factor;
    Prime(T n) {
        eular(n);
        // eratosthenes(n);
        // sqrt_sieve
        // factorize(n);
    }
    void show() {
        for(T e : list) printf("%lld ", e);
        putchar('\n');
    }

  private:
    bitset<N> notprime; // 1e8<2^27=128MB
    void eular(T n) {
        for(T i = 2; i <= n; ++i) {
            if(!notprime[i]) list.emplace_back(i);
            const T k = n / i;
            for(T j : list) {
                if(j > k) break;
                notprime[i * j] = 1;
                if(!(i % j)) break;
            }
        }
    }
    void eratosthenes(T n) {
        for(T i = 2; i <= n; ++i) {
            if(!notprime[i]) list.emplace_back(i);
            const T k = n / i;
            for(T j : list) {
                if(j > k) break;
                notprime[i * j] = 1;
                if(!(i % j)) break;
            }
        }
    }
    void sqrt_sieve(T n) {
        for(T i = 2; i <= n; ++i) {
            bool isprime = 1;
            for(T j : list) {
                if(j > i / j) break;
                if(!(i % j)) {
                    isprime = 0;
                    break;
                }
            }
            if(isprime) list.emplace_back(i);
        }
    }
    void factorize(T n) {
        factor = vector<T>(n);
        if(list.empty()) eular(n);
        for(T j : list) factor[j] = j;
        for(T i = 2; i <= n; ++i) {
            const T k = n / i;
            for(T j : list) {
                if(j > k) break;
                factor[i * j] = j;
                if(!(i % j)) break;
            }
        }
    }
#undef N
};
/*Prime prime(n) 建立打好 1~n 質數表的物件
prime.list(一個 vector) 是質數表
可修改 define N 決定歐篩/埃篩上限
可在建構子選擇篩法 有歐篩/埃篩/根號暴力搜
prime.factorize(n) 用歐篩方式得到 1~n 所有數的最小質因數
可在 factor(一個 vector) 上一路回溯 logn 得到一個數的質因數分解
做 n 個數質因數分解共花 nlogn
show() 會以空格隔開 顯示所有 list 內的元素 有尾空格尾換行
printf 裡面用%lld 視情況換為%d 或 cout*/
```

11. AnotherVersionString

11.1. KMP (2).cpp

```
#define V vector
V<int> kmp(string s) {
    int n = s.size();
    V<int> f(n);
    for(int i = 1; i < n; ++i) {
        int j = f[i - 1];
        for(; j > 0 && s[j] != s[i];) j = f[j - 1];
        f[i] = j + (s[j] == s[i]);
    }
    return f;
}
// kmp(s+"#"+t) 得到的陣列中，f[i]=s.size() 的格子代表 t
// 中匹配到 s 的結尾位置
```

11.2. KMP.cpp

```
class Kmp {
#define N 1000005
  public:
    int fail[N], p[N];
    Kmp(char *t, int n) {
        fail[0] = -1;
        for(int i = 1; i < n; ++i) {
            for(fail[i] = fail[i - 1];
                t[i] != t[fail[i] + 1] && fail[i] != -1;)
                fail[i] = fail[fail[i]];
            if(t[i] == t[fail[i] + 1]) ++fail[i];
        }
    }
    void match(char *s, int n, char *t, int m) {
        p[0] = (s[0] == t[0]) - 1;
        for(int i = 1; i < n; ++i) {
            for(p[i] = p[i - 1];
                s[i] != t[p[i] + 1] && p[i] != -1;)
                p[i] = fail[p[i]];
            if(s[i] == t[p[i] + 1]) ++p[i];
        }
    }
#undef N
};
/*Kmp kmp(t) 會建好 t 的失配函數 fail[]
 * match 會把每格匹配完的失配函數 p[] 建好 */
```

11.3. Manacher (2).cpp

```
#define T(x) ((x)&1 ? s[(x) >> 1] : '.')
int ex(string &s, int l, int r, int n) {
    int i = 0;
    while(l - i >= 0 && r + i < n && T(l - i) == T(r + i)) ++i;
    return i;
}
int manacher(string s, int n) {
    n = 2 * n + 1;
    int mx = 0;
    int center = 0;
    vector<int> r(n);
    int ans = 1;
    r[0] = 1;
    for(int i = 1; i < n; i++) {
        int ii = center - (i - center);
        int len = mx - i + 1;
        if(i > mx) {
            r[i] = ex(s, i, i, n);
            center = i;
            mx = i + r[i] - 1;
        } else if(r[ii] == len) {
            r[i] = len + ex(s, i - len, i + len, n);
            center = i;
            mx = i + r[i] - 1;
        } else {
            r[i] = min(r[ii], len);
        }
        ans = max(ans, r[i]);
    }
    return ans - 1;
}
```

11.4. Manacher.cpp

```
#define V vector
string manacher(string t) {
    int n = t.size() << 1 | 1;
    string s(n, '#');
    for(int i = 0, m = t.size(); i < m; ++i)
        s[i << 1 | 1] = t[i];
    V<int> p(n);
    for(int i = 0, m = 0, r = 0; i < n; ++i) {
        p[i] = r > i ? min(r - i, p[m - (i - m)]) : 1;
        for(; i - p[i] >= 0 && i + p[i] < n &&
              s[i - p[i]] == s[i + p[i]];)
            ++p[i];
        if(i + p[i] > r) r = i + p[i], m = i;
    }
    int k = 0;
```

```cpp
    string ans = "";
    for(int i = 0; i < n; ++i)
        if(p[i] > p[k]) k = i;
    for(int r = k + p[k], l = k - p[k]; ++l < r;)
        if(s[l] != '#') ans += s[l];
    return ans;
}
// manacher(s) 給出 s
// 中的最長回文，若有多個則給字典序最小的，p[i] = 以 i
// 為中心的最大回文半徑，所有字之間和頭尾都加上 '#'
```

11.5. Z.cpp

```cpp
class Z {
  public:
    vector<int> z;
    Z(string s) {
        z = vector<int>(s.size());
        for(int l = 0, i = 1; i < n; ++i) {
            if(l + z[l] >= i)
                z[i] = min(z[l] + l - i, z[i - l]);
            while(i + z[i] < n && s[z[i]] == s[i + z[i]])
                ++z[i];
            if(i + z[i] > l + z[l]) l = i;
        }
    }
};
// Z(s+"#"+t) 得到的陣列中，f[i]=s.size() 的格子代表 t
// 中匹配到 s 的開頭位置
```

12. AnotherVersionGraph

12.1. Dijkstra.cpp

```cpp
// cses Shortest Routes I

using namespace std;
#define N 100005
#define LL long long
#define pii pair<int, int>
#define pil pair<LL, LL>
#define F first
#define S second
#define pb push_back
#define DE if(1)
#define INF (LL)1e16
vector<pil> adj[N];
LL d[N];
bitset<N> vis;
int main() {
    int n, m, u, v;
    LL c;
    priority_queue<pil, vector<pil>, greater<pil>> q;
    for(cin >> n >> m; m--;)
        cin >> u >> v >> c, adj[u].pb({v, c});
    q.push({0, 1});
    d[1] = 0;
    for(u = 2; u <= n; ++u) d[u] = INF;
    for(; !q.empty(); q.pop()) {
        if(vis[q.top().S]) continue;
        vis[q.top().S] = 1;
        for(auto &e : adj[q.top().S]) {
            if(!vis[e.F] && q.top().F + e.S < d[e.F]) {
                d[e.F] = q.top().F + e.S;
                q.push({d[e.F], e.F});
            }
        }
    }
    for(u = 1; u <= n; ++u) printf("%lld ", d[u]);
}
```

12.2. SCC.cpp

```cpp

using namespace std;
#define pb push_back
#define pii pair<int, int>
#define N 100005
vector<int> adj[N];
stack<int> st;
int dfn[N], low[N], tag, scc[N], scchead[N], sc;
bitset<N> in;
void dfs(int now, int par = -1) {
    st.push(now);
    in[now] = 1;
    low[now] = dfn[now] = ++tag;
    for(int e : adj[now]) {
        if(e == par) continue;
        if(!dfn[e])
            dfs(e, now), low[now] = min(low[now], low[e]);
        else if(in[e])
            low[now] = min(low[now], dfn[e]);
    }
    if(dfn[now] == low[now]) {
        ++sc;
        for(; st.top() != now; st.pop())
            scc[st.top()] = sc, in[st.top()] = 0;
        st.pop();
        scc[now] = sc;
        in[now] = 0;
        scchead[sc] = now;
    }
}
int main() {
    int n, m, u, v;
    cin >> n >> m;
    vector<pii> g(m);
    for(auto &[u, v] : g)
        cin >> u >> v, adj[u].pb(v), adj[v].pb(u);
    for(u = 1; u <= n; ++u)
        if(!dfn[u]) dfs(u);
    int ans = 0;
    for(auto &[u, v] : g)
        if(scc[u] != scc[v]) ++ans; //=eBCC
    cout << ans << "\n";
    for(auto &[u, v] : g)
        if(scc[u] != scc[v]) cout << u << " " << v << "\n";
}
```

12.3. cses 有向圖基環樹森林.cpp

```cpp
// cses Planets Queries II 基環樹森林模板

using namespace std;
#define N 200005
#define pb push_back
// int cyc[i]=1~n 代表 i 屬於哪顆樹
// bitset incyc[i]=0/1 代表 i 是否在環上
// int len[k]=1~n 代表第 k 棵樹的環長度
// int num[i]=1~n 如果 incyc[i] 代表的是在環上的編號
// 否則代表的是環上最近的點的編號 int dis[i]=0~n-1
// 代表到環上最近點的距離 若 i 在環上則為 0
int tag = 1, cyc[N], len[N], num[N], dis[N], nxt[N][19];
bitset<N> vis, incyc;
vector<int> path;
void dfs(int now) {
    if(vis[now]) {
        int i = 1;
        for(int k; k = path.back(), path.pop_back(),
                   k != now && !path.empty();
            ++i) {
            cyc[k] = tag;
            incyc[k] = 1;
            num[k] = i;
        }
        cyc[now] = tag;
        incyc[now] = 1;
        num[now] = i;
        len[tag] = i;
        ++tag;
        return;
    }
    vis[now] = 1;
    path.pb(now);
    if(!cyc[nxt[now][0]]) dfs(nxt[now][0]);
    if(cyc[now]) return;
    cyc[now] = cyc[nxt[now][0]];
    num[now] = num[nxt[now][0]];
    dis[now] = dis[nxt[now][0]] + 1;
}
int jmp(int a, int x) {
    for(int k = 19; k--;)
        for(; 1 << k <= x;) x -= 1 << k, a = nxt[a][k];
    return a;
}
int main() {
    ios::sync_with_stdio(0);
    cin.tie(0);
    cout.tie(0);
    int n, q, i = 1, u, v;
    for(cin >> n >> q; i <= n; ++i) cin >> nxt[i][0];
    for(int k = 1; k < 19; ++k)
        for(i = 1; i <= n; ++i)
            nxt[i][k] = nxt[nxt[i][k - 1]][k - 1];
    for(i = 1; i <= n; ++i)
        if(!cyc[i]) path.clear(), dfs(i);
    for(; q--;) {
        cin >> u >> v;
        if(cyc[u] == cyc[v]) {
            if(incyc[v])
                cout << (!incyc[u] ? dis[u] : 0) +
                            (num[u] - num[v] + len[cyc[u]]) %
                                len[cyc[u]]
                     << "\n";
            else if(num[u] == num[v] && dis[u] >= dis[v] &&
                    jmp(u, dis[u] - dis[v]) == v)
                cout << dis[u] - dis[v] << "\n";
            else
                cout << "-1\n";
        } else
            cout << "-1\n";
    }
}
```

13. AnotherVersionGeometry

13.1. DynamicHull.cpp

```cpp
struct Line {
    mutable int a, b, r;
    bool operator<(const Line &o) const { return a < o.a; }
    bool operator<(const int o) const { return r < o; }
};

struct DynamicHull : multiset<Line, less<>> {
    inline int Div(int a, int b) {
        return a / b - ((a ^ b) < 0 && a % b);
    }
    inline bool intersect(iterator x, iterator y) {
        if(y == end()) {
            x->r = inf;
            return false;
        }
        if(x->a == y->a)
            x->r = (x->b) > (y->b) ? inf : -inf;
        else
            x->r = Div((y->b) - (x->b), (x->a) - (y->a));
        return (x->r) >= (y->r);
    }
    void Insert(int a, int b) {
        auto y = insert({a, b, 0}), z = next(y), x = y;
        while(intersect(y, z)) z = erase(z);
        if(x != begin() && intersect(--x, y))
            intersect(x, y = erase(y));
        while((y = x) != begin() && ((--x)->r) >= (y->r))
            intersect(x, erase(y));
    }
    int query(int x) const {
        auto l = *lower_bound(x);
        return (l.a) * x + (l.b);
    }
};
```

14. AnotherVersionTree

14.1. LCA.cpp

```cpp
#define N 100005
#define LG 15
int dep[N], par[N][LG], sub[N];
vector<int> g[N];
void dfs(int now = 1, int pre = 0) {
    dep[now] = dep[pre] + 1;
    par[now][0] = pre;
    sub[now] = 1;
    for(int e : g[now])
        if(e != pre) dfs(e, now), sub[now] += sub[e];
}
int jmp(int x, int k) {
    for(int i = LG; i--;)
        for(; k >= 1 << i; k -= 1 << i) x = par[x][i];
    return x;
}
int lca(int a, int b) {
    if(dep[a] > dep[b]) swap(a, b);
    b = jmp(b, dep[b] - dep[a]);
    if(a == b) return a;
    for(int i = LG; i--;)
        for(; par[a][i] != par[b][i]; b = par[b][i])
            a = par[a][i];
    return par[a][0];
}
int main() {
    int n;
    cin >> n;
    for(int i = n, u, v; --i;)
        cin >> u >> v, g[u].pb(v), g[v].pb(u);
    dfs();
    for(int i = 1; i < LG; ++i)
        for(int j = 1; j <= n; ++j)
            par[j][i] = par[par[j][i - 1]][i - 1];
    int k = lca(1, n);
}
// 點編號 1~n，建的無向圖但改 dfs
// 就能變有向，改有向記得邊要反著建 dep[n] 代表 n 的深度 (1
//  base)，par[i][j] 代表 i 往上 1<<j 步的祖先是誰，不存在則是
//  0，sub[i] 代表 i 的子樹大小 jmp(i,j) 代表 i 往上 j
//  步的祖先是誰

#pragma GCC optimize(                                             \
    "Ofast,fast-math,unroll-loops,no-stack-protector")

using namespace std;
#define ll long long
#define pb push_back
#define N 200005
#define pii pair<int, int>
#define V vector
#define inf 1000000007
#define M 200005
#define LG 18
#define pii pair<int, int>
#define ppp pair<pii, pii>
char buf[1 << 22], *p1, *p2;
int p[12];
#define gc()                                                      \
    (p1 == p2 &&                                                  \
             (p2 = (p1 = buf) + fread(buf, 1, 1 << 22, stdin),    \
              p1 == p2)                                           \
         ? EOF                                                    \
         : *p1++)
inline int gi() {
    int x = 0;
    for(char c; '0' <= (c = gc()) && c <= '9'; x += c - '0')
        x *= 10;
    return x;
}
inline void pi(int x, char c = ' ') {
    if(!x) putchar('0');
    int i = 0;
    for(; x; x /= 10) p[i++] = x % 10;
    for(; i--;) putchar(p[i] + '0');
    putchar(c);
}
int main() {
    cin.tie(0)->sync_with_stdio(0);
    int n, m, q;
    cin >> n >> m >> q;
    vector<ppp> g(m);
    bitset<M> ans;
    vector<vector<pii>> adj(n + 1, vector<pii>());
    for(int i = 0; i < m; ++i) {
        auto &[p1, p2] = g[i];
        auto &[w, idx] = p1;
        auto &[u, v] = p2;
        cin >> u >> v >> w;
        idx = i;
    }
    sort(g.begin(), g.end());
    vector<ll> dsu(n + 1, -1);
    auto qry = [&dsu](auto qry, int x) -> int {
        return dsu[x] < 0 ? x : dsu[x] = qry(qry, dsu[x]);
    };
    auto upd = [&dsu, &qry](int u, int v) -> void {
        if(dsu[u = qry(qry, u)] > dsu[v = qry(qry, v)])
            swap(u, v);
        dsu[u] += dsu[v];
        dsu[v] = u;
    };
    for(auto &[p1, p2] : g) {
        auto &[w, idx] = p1;
        auto &[u, v] = p2;
        if(qry(qry, u) != qry(qry, v))
            upd(u, v), adj[u].pb({v, w}), adj[v].pb({u, w});
    }
    vector<vector<int>> par(n + 1, vector<int>(LG)),
        mx(n + 1, vector<int>(LG));
    vector<int> dep(n + 1);
    auto dfs = [&par, &mx, &dep, &adj](auto dfs, int now,
                                       int p = 0,
                                       int w = 0) -> void {
        par[now][0] = p;
        mx[now][0] = w;
        dep[now] = dep[p] + 1;
        for(auto &[e, w] : adj[now])
            if(e != p) dfs(dfs, e, now, w);
    };
    dfs(dfs, 1);
    for(int i = 1; i < LG; ++i)
        for(int j = 1; j <= n; ++j)
            par[j][i] = par[par[j][i - 1]][i - 1],
            mx[j][i] =
                max(mx[j][i - 1], mx[par[j][i - 1]][i - 1]);
    auto lca = [&par, &dep](int u, int v) -> int {
        if(dep[u] > dep[v]) swap(u, v);
        for(int i = LG; i--;)
            if((1 << i) & (dep[v] - dep[u])) v = par[v][i];
        if(u == v) return u;
        for(int i = LG; i--;)
            if(par[u][i] != par[v][i])
                u = par[u][i], v = par[v][i];
        return par[u][0];
    };
    auto path = [&par, &mx, &dep](int k, int x) -> int {
        int ans = 0;
        for(int i = LG; i--;)
            if((1 << i) & (dep[x] - dep[k]))
                ans = max(ans, mx[x][i]), x = par[x][i];
        return ans;
    };
    for(auto &[p1, p2] : g) {
        auto &[w, idx] = p1;
        auto &[u, v] = p2;
        int k = lca(u, v);
        ans[idx] = max(path(k, u), path(k, v)) >= w;
    }
    for(int i = 0; i < m; ++i)
        cout << i << " "
             << (const char[2][5]){"NO\n", "YES\n"}[ans[i]];
    cout << "\n";
    for(int k; q--;) {
        cin >> k;
        int flag = 1;
        for(int x; k--;) {
            cin >> x;
            if(!ans[x - 1]) flag = 0;
        }
        cout << (const char[2][5]){"NO\n", "YES\n"}[flag];
    }
}
```