

Contents

1 DataStructure	1	8 Misc	14
1.1 2DBIT.cpp	1	8.1 BigNum(luoguP1005).cpp	14
1.2 DynamicSegmentTree.cpp	2	8.2 Tri-search.cpp	15
1.3 PbdsGpHashTable.cpp	2	8.3 對拍.md	15
1.4 PbdsPriorityQueue.cpp	2	9 AnotherVersionDataStructure	16
1.5 PbdsRope.cpp	2	9.1 BIT.cpp	16
1.6 PbdsTree.cpp	2	9.2 DSU.cpp	16
1.7 PersistentSegmentTree.cpp	3	9.3 Treap.cpp	16
1.8 Treap.cpp	3	9.4 Treap 但可以多個數縮點 (疑似爛的).cpp	17
2 Math	3	9.5 區間插線段單點查詢李超 (是爛的).cpp	18
2.1 CRT.cpp	3	9.6 單點修改動態開點線段樹.cpp	19
2.2 CountPrimes.cpp	4	9.7 單點修改無懶標線段樹.cpp	19
2.3 FFT.cpp	4	9.8 懶標線段樹.cpp	19
2.4 FWT.cpp	5	9.9 純直線單點查詢李超.cpp	20
2.5 Formula.tex	5	10 AnotherVersionMath	20
2.5.1 Dirichlet Convolution	5	10.1 CRT(luoguVersion).cpp	20
2.5.2 Burnside's Lemma	5	10.2 PollardRho.cpp	20
2.5.3 Pick Theorem	5	10.3 快速幂.cpp	20
2.5.4 Fermat's Little Theorem	5	10.4 數論.cpp	21
2.5.5 Wilson's Theorem	5	10.5 篩法.cpp	21
2.5.6 Legendre Theorem	5	11 AnotherVersionString	22
2.5.7 Kummer Theorem	5	11.1 KMP (2).cpp	22
2.5.8 ext-Kummer Theorem	5	11.2 KMP.cpp	22
2.5.9 Factorial with mod	5	11.3 Manacher (2).cpp	22
2.5.10 Properties of nCr with mod	5	11.4 Manacher.cpp	22
2.5.11 ext-Lucas' Theorem	5	11.5 Z.cpp	22
2.5.12 Catalan Number	5	12 AnotherVersionGraph	22
2.5.13 modinv table	5	12.1 Dijkstra.cpp	22
2.5.14 LTE	5	12.2 SCC.cpp	23
2.6 Gaussian-Jordan.cpp	5	12.3 cses 有向圖基環樹森林.cpp	23
2.7 Generator.cpp	5	13 AnotherVersionGeometry	23
2.8 Inv.cpp	6	13.1 DynamicHull.cpp	23
2.9 Lucas.cpp	6	14 AnotherVersionTree	23
2.10 MillerRabin.cpp	6	14.1 LCA.cpp	23
2.11 Mu.cpp	6		
2.12 NTT.cpp	7		
2.13 PollardRho.cpp	7		
2.14 XorBasis.cpp	7		
2.15 mtt.cpp	8		
3 String	8	1. DataStructure	
3.1 Booth.cpp	8	1.1. 2DBIT.cpp	
3.2 KMP.cpp	8		
3.3 LongestPalindrome.cpp	8		
3.4 Z.cpp	8		
4 Graph	9		
4.1 2-SAT(CSES Planets Cycles).cpp	9		
4.2 Dijkstra.cpp	10		
4.3 Dinic.cpp	10		
4.4 MaximumFlow.cpp	10		
4.5 SCC.cpp	10		
4.6 VBCC.cpp	11		
4.7 one-degree-cycle(CSES Planets Cycles).cpp	11		
5 DP	11		
5.1 CHO.cpp	11		
5.2 Li-Chao-SegmentTree.cpp	11		
5.3 SOSDP.cpp	12		
6 Geometry	12		
6.1 164253Version.cpp	12		
6.2 ConvexHull.cpp	13		
6.3 Inside.cpp	13		
6.4 Intersect.cpp	13		
6.5 MinimumEuclideanDistance.cpp	13		
7 Tree	13		
7.1 HeavyLightDecomposition(modify-and-query-on-path).cpp	13		
7.2 LCA.cpp	14		

```

1 // cses Forest Queries II
2 #include <bits/stdc++.h>
3 using namespace std;
4 #define LL long long
5 #define pii pair<int, int>
6 #define N 1005
7 #define F first
8 #define S second
9 int bit[N][N];
10 #define lb(x) (x & -x)
11 void upd(int i, int j, int v) {
12     for(int k = i; k < N; k += lb(k)) bit[k][j] += v;
13 }
14 int qry2(int i, int j) {
15     int ans = 0;
16     for(; j; j -= lb(j))
17         for(int k = i; k < N; k += lb(k)) ans += bit[k][j];
18     return ans;
19 }
20 int qry(int y1, int x1, int y2, int x2) {
21     return qry2(y2, x2) - qry2(y2, x1 - 1) - qry2(y1 - 1, x2) +
22         qry2(y1 - 1, x1 - 1);
23 }
24 int main() {
25     int n, q, i = 1, j, y, x;
26     for(scanf("%d %d", &n, &q); getchar(), i <= n; ++i)
27         for(j = 1; j <= n; ++j)
28             if(getchar() == '*') upd(i, j, 1);
29     for(; q--;) {
30         scanf("%d", &i);
31         if(i == 1)
32             scanf("%d %d", &i, &j),
33             upd(i, j, 1 - 2 * qry(i, j, i, j));
34         else
35

```

```
scanf("%d%d%d", &i, &j, &y, &x),
printf("%d\n", qry(i, j, y, x));
}
```

1.2. DynamicSegmentTree.cpp

```
1 #include <bits/stdc++.h>
2 #define int long long
3 using namespace std;
4
5 int n, q;
6 struct node {
7     int data, lson, rson, tag;
8     int rv() { return data + tag; }
9 };
10
11 node tree[20000005];
12 int a[200005];
13 int now = 1;
14 int mx = 1000000005;
15
16 void push(int index) {
17     if(!tree[index].lson) {
18         tree[index].lson = ++now;
19     }
20     if(!tree[index].rson) {
21         tree[index].rson = ++now;
22     }
23     int lson = tree[index].lson;
24     int rson = tree[index].rson;
25     tree[lson].tag += tree[index].tag;
26     tree[rson].tag += tree[index].tag;
27     tree[index].data = tree[index].rv();
28     tree[index].tag = 0;
29 }
30
31 void modify(int l, int r, int L, int R, int val, int index) {
32     if(l == L && r == R) {
33         tree[index].tag += val;
34         return;
35     }
36     int mid = (l + r) >> 1;
37     push(index);
38     int lson = tree[index].lson;
39     int rson = tree[index].rson;
40     if(R <= mid) {
41         modify(l, mid, L, R, val, lson);
42     } else if(L > mid) {
43         modify(mid + 1, r, L, R, val, rson);
44     } else {
45         modify(l, mid, L, mid, val, lson);
46         modify(mid + 1, r, mid + 1, R, val, rson);
47     }
48     tree[index].data = tree[lson].rv() + tree[rson].rv();
49 }
50
51 int query(int l, int r, int L, int R, int index) {
52     // cout << L << " " << R << "\n";
53     if(l == L && r == R) {
54         return tree[index].rv();
55     }
56     int mid = (l + r) >> 1;
57     push(index);
58     int lson = tree[index].lson;
59     int rson = tree[index].rson;
60     if(R <= mid) {
61         return query(l, mid, L, R, lson);
62     }
63     if(L > mid) {
64         return query(mid + 1, r, L, R, rson);
65     }
66     return query(l, mid, L, mid, lson) +
67         query(mid + 1, r, mid + 1, R, rson);
68 }
69
70 signed main() {
71     ios::sync_with_stdio(0);
72     cin.tie(0);
73     cout.tie(0);
74     cin >> n >> q;
75     for(int i = 1; i <= n; i++) {
76         cin >> a[i];
77         modify(1, mx, a[i], a[i], 1, 1);
78     }
79     while(q--) {
80         char mode;
81         int x, y;
82         cin >> mode;
83         if(mode == '?') {
84             cin >> x >> y;
85             cout << query(1, mx, x, y, 1) << "\n";
86         } else {
87             cin >> x >> y;
88             modify(1, mx, a[x], a[x], -1, 1);
89             a[x] = y;
90             modify(1, mx, a[x], a[x], 1, 1);
91         }
92     }
93 }
```

1.3. PbdsGpHashTable.cpp

```
1 #include <bits/extc++.h>
2 using namespace __gnu_pbds;
3 #define ull unsigned ll
4 mt19937 mt(hash<string>{}("164253_official_beautiful_fruit"));
5 struct myhash {
6     static ull splitmix64(ull x) {
7         x += 0x9e3779b97f4a7c15;
8         x = (x ^ (x >> 30)) * 0xbf58476d1ce4e5b9;
9         x = (x ^ (x >> 27)) * 0x94d049bb133111eb;
10        return x ^ (x >> 31);
11    }
12    ull operator()(ull x) const {
13        static const ull FIXED_RANDOM =
14            (ull)make_unique<char>().get() ^
15            chrono::high_resolution_clock::now()
16                .time_since_epoch()
17                .count();
18        // static const ull FIXED_RANDOM=mt();
19        // static const ull
20        // FIXED_RANDOM=chrono::steady_clock::now()
21        // .time_since_epoch().count();
22        return splitmix64(x + FIXED_RANDOM);
23    }
24 };
25 /*
26 gp_hash_table<ull,ull,myhash> gp;
27 gp[x]=y;
28 if(gp.find(x)!=gp.end())cout<<gp[x];
29 gp.count(); //CE
30 */
```

1.4. PbdsPriorityQueue.cpp

```
1 #include <bits/extc++.h>
2 __gnu_pbds::priority_queue<int> pq;
3 /*
4 push(x); //return iterator
5 __gnu_pbds::priority_queue<T>::point_iterator
6 pop() top() join(pq2) erase(iter) modify(iter,x)
7 */
```

1.5. PbdsRope.cpp

```
1 #include <bits/extc++.h>
2 using namespace __gnu_cxx;
3 /*
4 rope<int> r;
5 r.erase(pos,k); //r=r.[0,pos)+r.[pos+k,r.length());
6 push_back(x) pop_back() insert(pos,x) clear() find(x)
7 lower_bound(all(r),x) upper_bound //same as vector
8 r.length(); //same as .length
9 r.replace(pos,len=r.length(),x); //r.[pos,pos+len]=x;
10 r.substr(pos,x); //return r.[pos,pos+x);
11 rope<char> s="official_beautiful_fruit";
12 cout<<s; //it's legal
13 */
```

1.6. PbdsTree.cpp

```
1 #include <bits/extc++.h>
2 using namespace __gnu_pbds;
3 using BST = tree<int, null_type, less<int>, splay_tree_tag,
4     tree_order_statistics_node_update>;
5 // rb_tree_tag with log^2(n) split
6 using BST_Itr = BST::iterator;
7 BST tr;
8 // overload std::distance for BST for efficiently split
9 namespace std {
10     template <>
11     iterator_traits<BST_Itr>::difference_type
12     distance(BST_Itr begin, BST_Itr end) {
13         if(begin == end) return 0;
14         auto it = begin.m_p_nd;
15         // jump until root
16         while(it->m_p_parent->m_p_parent != it) it = it->m_p_parent;
17         // returns the size for the whole tree (only for split)
18         return it->get_metadata();
19     }
20 } // namespace std
21 void splayAfterSplit(BST &bst) {
22     if(bst.empty()) return;
23     bst.find(*bst.begin());
```

```

}
/*
25 除了 tr.lower_bound(x) upper_bound insert same as rope<int>
27 tr.find_by_order(k); //return kth iterator; k=[0,tr.size())
//out of this will get tr.end()
29 tr.order_of_key(val); //return rank(val);
tr.join(tr2); //merge tr
31 and tr2, tr2.clear() tr.split(const int&r, RBTREE&tr2); //<r
will in tr, >=r will in tr2
33 */

```

1.7. PersistentSegmentTree.cpp

```

1 // cses Range Queries and Copies
#include <bits/stdc++.h>
3 using namespace std;
#define LL long long
5 #define pii pair<int, int>
#define N 200005
7 #define F first
#define S second
9 int n, ver = 1;
LL a[N];
11 struct Seg {
    LL v = 0;
13     struct Seg *l = NULL, *r = NULL;
#define M (L + R >> 1)
15     static const void init(Seg *node, int L = 1, int R = n) {
        if(L == R) {
17             node->v = a[L];
            return;
19         }
        node->l = new Seg();
21         init(node->l, L, M);
        node->r = new Seg();
23         init(node->r, M + 1, R);
        node->v = node->l->v + node->r->v;
25     }
    static const void upd(Seg *node, int x, LL v, int L = 1,
27                         int R = n) {
        if(L == R) {
29             node->v = v;
            return;
31         }
        if(x <= M)
33             node->l = new Seg(*node->l),
            upd(node->l, x, v, L, M);
35         else
            node->r = new Seg(*node->r),
37             upd(node->r, x, v, M + 1, R);
        node->v = node->l->v + node->r->v;
39     }
    static const LL qry(Seg *node, int l, int r, int L = 1,
41                        int R = n) {
        if(l <= L && R <= r) return node->v;
43         if(r <= M) return qry(node->l, l, r, L, M);
        if(M + 1 <= l) return qry(node->r, l, r, M + 1, R);
45         return qry(node->l, l, M, L, M) +
            qry(node->r, M + 1, r, M + 1, R);
47     }
} * tree[N];
49 int main() {
    ios::sync_with_stdio(0);
51     cin.tie(0);
    cout.tie(0);
53     int q, i = 1, j, k;
    for(cin >> n >> q; i <= n; ++i) cin >> a[i];
55     tree[1] = new Seg();
    Seg::init(tree[1]);
57     for(; q--;) {
        cin >> i >> k;
59         if(i == 1)
            cin >> i >> j, Seg::upd(tree[k], i, j);
61         else if(i == 2)
            cin >> i >> j,
63             cout << Seg::qry(tree[k], i, j) << "\n";
        else
65             tree[++ver] = new Seg(*tree[k]);
    }
67 }

```

1.8. Treap.cpp

```

1 #define pii pair<int, int>
struct node {
3     int tag = 0;
    int sum = 0;
5     int prio = rand();
    int lson = 0;
7     int rson = 0;
    int si = 0;
9     int val = 0;

```

```

};
11 node treap[400005];
int cnt = 0;
13 int root = 0;

15 void update(int index) {
    int lson = treap[index].lson;
17     int rson = treap[index].rson;
    treap[index].si = treap[lson].si + treap[rson].si + 1;
19     treap[index].sum = treap[lson].sum;
    treap[index].sum += treap[rson].sum;
21     treap[index].sum += treap[index].val;
}

23 void push(int index) {
    if(!treap[index].tag) return;
25     swap(treap[index].lson, treap[index].rson);
    int lson = treap[index].lson;
27     int rson = treap[index].rson;
    treap[lson].tag ^= 1;
29     treap[rson].tag ^= 1;
    treap[index].tag = 0;
31 }

33 pii split(int rk, int index) {
    if(!index) return {0, 0};
35     push(index);
    int lson = treap[index].lson;
37     int rson = treap[index].rson;
    if(rk <= treap[lson].si) {
39         pii temp = split(rk, lson);
        treap[index].lson = temp.second;
41         update(index);
        return {temp.first, index};
43     } else {
        pii temp = split(rk - treap[lson].si - 1, rson);
45         treap[index].rson = temp.first;
        update(index);
47         return {index, temp.second};
    }
49 }

51 int merge(int x, int y) {
    if(!x && !y) return 0;
53     if(!x && y) return y;
    if(x && !y) return x;
55     push(x);
    push(y);
    if(treap[x].prio < treap[y].prio) {
57         treap[x].rson = merge(treap[x].rson, y);
        update(x);
59         return x;
    } else {
        treap[y].lson = merge(x, treap[y].lson);
63         update(y);
        return y;
65     }
}

67 void insert(int x, int v) {
    pii temp = split(x - 1, root);
69     cnt++;
    treap[cnt].val = v;
71     update(cnt);
    temp.first = merge(temp.first, cnt);
73     root = merge(temp.first, temp.second);
75 }

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

85 void modify(int l, int r) {
    pii R = split(r, root);
87     pii L = split(l - 1, R.first);
    treap[L.second].tag ^= 1;
89     R.first = merge(L.first, L.second);
    root = merge(R.first, R.second);
91 }

```

2. Math

2.1. CRT.cpp

```

1 #include <bits/stdc++.h>
#define int long long
3 using namespace std;

```

```

5 int n;
6 int a[15];
7 int b[15];
8 int mul = 1;
9
10 void exgcd(int a, int b, int &x, int &y) {
11     if(b == 0) {
12         x = 1;
13         y = 0;
14         return;
15     }
16     exgcd(b, a % b, y, x);
17     y -= (a / b) * x;
18 }
19
20 int inv(int a, int p) {
21     int x, y;
22     exgcd(a, p, x, y);
23     return x;
24 }
25
26 int ans = 0;
27
28 signed main() {
29     cin >> n;
30     for(int i = 1; i <= n; i++) {
31         cin >> a[i] >> b[i];
32         mul *= a[i];
33     }
34     for(int i = 1; i <= n; i++) {
35         ans += inv(mul / a[i], a[i]) * (mul / a[i]) % mul *
36             b[i] % mul;
37         ans %= mul;
38     }
39     ans = (ans + mul) % mul;
40     cout << ans;
41 }

```

2.2. CountPrimes.cpp

```

1 #include <bits/stdc++.h>
2 using namespace std;
3 using i64 = long long;
4 i64 count_pi(i64 N) {
5     if(N <= 1) return 0;
6     int v = sqrt(N + 0.5);
7     int n_4 = sqrt(v + 0.5);
8     int T = min((int)sqrt(n_4) * 2, n_4);
9     int K = pow(N, 0.625) / log(N) * 2;
10    K = max(K, v);
11    K = min<i64>(K, N);
12    int B = N / K;
13    B = N / (N / B);
14    B = min<i64>(N / (N / B), K);
15
16    vector<i64> l(v + 1);
17    vector<int> s(K + 1);
18    vector<bool> e(K + 1);
19    vector<int> w(K + 1);
20    for(int i = 1; i <= v; ++i) l[i] = N / i - 1;
21    for(int i = 1; i <= v; ++i) s[i] = i - 1;
22
23    const auto div = [](i64 n, int d) -> int {
24        return double(n) / d;
25    };
26    int p;
27    for(p = 2; p <= T; ++p)
28        if(s[p] != s[p - 1]) {
29            i64 M = N / p;
30            int t = v / p, t0 = s[p - 1];
31            for(int i = 1; i <= t; ++i) l[i] -= l[i * p] - t0;
32            for(int i = t + 1; i <= v; ++i)
33                l[i] -= s[div(M, i)] - t0;
34            for(int i = v, j = t; j >= p; --j)
35                for(int l = j * p; i >= l; --i)
36                    s[i] -= s[j] - t0;
37            for(int i = p * p; i <= K; i += p) e[i] = 1;
38        }
39    e[1] = 1;
40    int cnt = 1;
41    vector<int> roughs(B + 1);
42    for(int i = 1; i <= B; ++i)
43        if(!e[i]) roughs[cnt++] = i;
44    roughs[cnt] = 0x7fffffff;
45    for(int i = 1; i <= K; ++i) w[i] = e[i] + w[i - 1];
46    for(int i = 1; i <= K; ++i) s[i] = w[i] - w[i - (i & -i)];
47
48    const auto query = [&](int x) -> int {
49        int sum = x;
50        while(x) sum -= s[x], x ^= x & -x;
51        return sum;
52    };
53    const auto add = [&](int x) -> void {

```

```

54        e[x] = 1;
55        while(x <= K) ++s[x], x += x & -x;
56    };
57    cnt = 1;
58    for(; p <= n_4; ++p)
59        if(!e[p]) {
60            i64 q = i64(p) * p, M = N / p;
61            while(cnt < q) w[cnt] = query(cnt), cnt++;
62            int t1 = B / p, t2 = min<i64>(B, M / q),
63                t0 = query(p - 1);
64            int id = 1, i = 1;
65            for(; i <= t1; i = roughs[++id])
66                l[i] -= l[i * p] - t0;
67            for(; i <= t2; i = roughs[++id])
68                l[i] -= query(div(M, i)) - t0;
69            for(; i <= B; i = roughs[++id])
70                l[i] -= w[div(M, i)] - t0;
71            for(int i = q; i <= K; i += p)
72                if(!e[i]) add(i);
73        }
74    while(cnt <= v) w[cnt] = query(cnt), cnt++;
75
76    vector<int> primes;
77    primes.push_back(1);
78    for(int i = 2; i <= v; ++i)
79        if(!e[i]) primes.push_back(i);
80    l[1] += i64(w[v] + w[n_4] - 1) * (w[v] - w[n_4]) / 2;
81    for(int i = w[n_4] + 1; i <= w[B]; ++i)
82        l[1] -= l[primes[i]];
83    for(int i = w[B] + 1; i <= w[v]; ++i)
84        l[1] -= query(N / primes[i]);
85    for(int i = w[n_4] + 1; i <= w[v]; ++i) {
86        int q = primes[i];
87        i64 M = N / q;
88        int e = w[M / q];
89        if(e <= i) break;
90        l[1] += e - i;
91        i64 t = 0;
92        int m = w[sqrt(M + 0.5)];
93        for(int k = i + 1; k <= m; ++k)
94            t += w[div(M, primes[k])];
95        l[1] += 2 * t - (i + m) * (m - i);
96    }
97    return l[1];
98 }

```

2.3. FFT.cpp

```

1 #include <bits/stdc++.h>
2 using namespace std;
3 inline int read() {
4     int ans = 0;
5     char c = getchar();
6     while(!isdigit(c)) c = getchar();
7     while(isdigit(c)) {
8         ans = ans * 10 + c - '0';
9         c = getchar();
10    }
11    return ans;
12 }
13 typedef complex<double> comp;
14 const int MAXN = 1000005;
15 const comp I(0, 1);
16 const double PI = acos(-1);
17 comp A[MAXN * 3], B[MAXN * 3], tmp[MAXN * 3], ans[MAXN * 3];
18 void fft(comp F[], int N, int sgn = 1) {
19     if(N == 1) return;
20     memcpy(tmp, F, sizeof(comp) * N);
21     for(int i = 0; i < N; i++)
22         *(i % 2 ? F + i / 2 + N / 2 : F + i / 2) = tmp[i];
23     fft(F, N / 2, sgn), fft(F + N / 2, N / 2, sgn);
24     comp *G = F, *H = F + N / 2;
25     comp cur = 1, step = exp(2 * PI / N * sgn * I);
26     for(int k = 0; k < N / 2; k++) {
27         tmp[k] = G[k] + cur * H[k];
28         tmp[k + N / 2] = G[k] - cur * H[k];
29         cur *= step;
30     }
31     memcpy(F, tmp, sizeof(comp) * N);
32 }
33 int main() {
34     int n = read(), m = read(), N = 1 << __lg(n + m + 1) + 1;
35     for(int i = 0; i <= n; ++i) A[i] = read();
36     for(int i = 0; i <= m; ++i) B[i] = read();
37     fft(A, N), fft(B, N);
38     for(int i = 0; i < N; ++i) ans[i] = A[i] * B[i];
39     fft(ans, N, -1);
40     for(int i = 0; i <= n + m; ++i)
41         printf("%d ", int(ans[i].real() / N + 0.1));
42     return 0;
43 }

```

2.4. FWT.cpp

```

1 #define LOGN 21
2 #define N (1 << LOGN)
3 void fwt(ll f[], int rev) {
4     for(int k = 1; k < LOGN; ++k) {
5         for(int i = 0, m = 1 << k - 1; i + m < N; i += 1 << k) {
6             for(int j = 0; j < m; ++j) {
7                 ll u = f[i + j], v = f[i + j + m];
8                 f[i + j] = u + v;
9                 f[i + j + m] = u - v;
10                if(rev) f[i + j] >>= 1, f[i + j + m] >>= 1;
11            }
12        }
13    }
}

```

2.5. Formula.tex

2.5.1. Dirichlet Convolution

$$\varepsilon = \mu * 1$$

$$\varphi = \mu * \text{Id}$$

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

$$\text{Area} = \text{inner lattice point} + \frac{\text{lattice point on border}}{2} - 1$$

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

$$v_p(n) := \text{power of } p \text{ in } n$$

$$(n)_p := \frac{n}{p^{(v_p(n))}}$$

$$s_p(n) := \text{sum of all digits of } n \text{ in base } p$$

$$v_p(n!) = \sum_{i=1}^{\infty} \lfloor \frac{n}{p^i} \rfloor = \frac{n - s_p(n)}{p - 1}$$

2.5.7. Kummer Theorem

$$v_p\left(\binom{n}{m}\right) = \frac{s_p(n) + s_p(m - n) - s_p(m)}{p - 1}$$

2.5.8. ext-Kummer Theorem

$$v_p\left(\binom{n}{m_1, m_2, \dots, m_k}\right) = \frac{\sum_{i=1}^k s_p(m_i) - s_p(n)}{p - 1}$$

2.5.9. Factorial with mod

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

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 \Leftrightarrow 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

$C_0 = C_1 = 1$, if $n > 1$ then $C_n = \sum_{k=0}^{n-1} C_k C_{n-1-k} = \frac{\binom{2n}{n+1}}{n+1}$. 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

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

2.5.14. LTE

$$p \text{ is odd prime, } n \in \mathbb{N}, x \in \mathbb{Z}, y \in \mathbb{Z}$$

$$p \mid (x - y), p \nmid x, p \nmid y$$

$$v_p(x^n - y^n) = v_p(x - y) + v_p(n)$$

$$v_2(x^n - y^n) = v_2(x - y) + v_2(n) \quad \text{if } n \text{ is odd}$$

$$v_2(x^n - y^n) = v_2(x - y) + v_2(x + y) + v_2(n) - 1 \quad \text{else}$$

$$4 \mid (x - y) \Rightarrow v_2(x + y) = 1, v_2(x^n - y^n) = v_2(x - y) + v_2(n)$$

2.6. Gaussian-Jordan.cpp

```

1 #include <bits/stdc++.h>
2 #define int long long
3 using namespace std;
4
5 int n;
6 double a[105][105];
7
8 // n <= m
9 void gaussian(double a[105][105], int n, int m) {
10     int curi = 0;
11     for(int j = 0; j < m; j++) {
12         int i;
13         for(i = curi; i < n; i++) {
14             if(a[i][j]) {
15                 break;
16             }
17         }
18         if(a[i][j] == 0) continue;
19         for(int k = 0; k < m; k++) {
20             swap(a[i][k], a[curi][k]);
21         }
22         for(int k = m - 1; k >= j; k--) {
23             a[curi][k] /= a[curi][j];
24         }
25         for(int i = 0; i < n; ++i) {
26             if(i != curi) {
27                 for(int k = m - 1; k >= j; k--) {
28                     a[i][k] -= a[curi][k] * a[i][j];
29                 }
30             }
31         }
32         curi++;
33     }
34 }

```

2.7. Generator.cpp

```

1 #include <bits/stdc++.h>
2 #define int long long
3 using namespace std;
4
5 int t;
6 int n, d;
7 bitset<1000005> exist;
8 bitset<1000005> vis;
9 vector<int> prime;
10 int phi[1000005];
11
12 void init() {
13     phi[1] = 1;
14     for(int i = 2; i <= 1000000; i++) {
15         if(!vis[i]) {
16             prime.push_back(i);
17             phi[i] = i - 1;
18         }
19         for(int j : prime) {
20             if(i * j > 1000000) break;
21             vis[i * j] = 1;
22             if(i % j == 0) {
23                 phi[i * j] = phi[i] * j;
24                 break;
25             } else {
26                 phi[i * j] = phi[i] * phi[j];
27             }
28         }
29     }
30     exist[2] = exist[4] = 1;
31     for(int i : prime) {
32         if(i == 2) continue;
33         for(int j = i; j <= 1000000; j += i) {
34             exist[j] = 1;
35             if(j * 2 <= 1000000) {
36                 exist[j * 2] = 1;
37             }
38         }
39     }
40 }
41
42 vector<int> factors(int x) {

```



```

43     vector<int> v;
44     for(int i = 1; i * i <= x; i++) {
45         if(x % i == 0) {
46             v.push_back(i);
47             if(i * i != x) {
48                 v.push_back(x / i);
49             }
50         }
51     }
52     return v;
53 }

55 int f(int x, int y, int mod) {
56     int ret = 1;
57     while(y) {
58         if(y & 1) {
59             ret *= x;
60             ret %= mod;
61         }
62         x *= x;
63         x %= mod;
64         y >>= 1;
65     }
66     return (ret % mod + mod) % mod;
67 }

69 vector<int> findroot(int x) {
70     vector<int> ret;
71     if(!exist[x]) return ret;
72     int phix = phi[x];
73     vector<int> fact = factors(phix);
74     int fst;
75     for(int i = 1; i <= phix; i++) {
76         if(_gcd(i, x) != 1) continue;
77         bool ok = 1;
78         for(int j : fact) {
79             if(j != phix && f(i, j, x) == 1) {
80                 ok = 0;
81                 break;
82             }
83         }
84         if(ok) {
85             fst = i;
86             break;
87         }
88     }
89     int now = fst;
90     // cout << fst << "\n";
91     for(int i = 1; i <= phix; i++) {
92         if(_gcd(i, phix) == 1) {
93             ret.push_back(now);
94         }
95         now *= fst;
96         now %= x;
97     }
98     return ret;
99 }

101 signed main() {
102     ios::sync_with_stdio(0);
103     cin.tie(0);
104     cout.tie(0);
105     init();
106     cin >> t;
107     while(t--) {
108         cin >> n >> d;
109         vector<int> v = findroot(n);
110         sort(v.begin(), v.end());
111         cout << v.size() << "\n";
112         for(int i = 0; i < v.size(); i++) {
113             if(i % d == d - 1) {
114                 cout << v[i] << " ";
115             }
116         }
117         cout << "\n";
118     }
119 }

```

2.8. Inv.cpp

```

1 int exgcd(int a, int b, int &x, int &y) {
2     if(b == 0) {
3         x = 1;
4         y = 0;
5         return a;
6     }
7     int d = exgcd(b, a % b, y, x);
8     y -= x * (a / b);
9     return d;
10 }

11 int inv(int a, int p) {
12     int x, y;

```

```

15     exgcd(a, p, x, y);
16     return (x % p + p) % p;
17 }

```

2.9. Lucas.cpp

```

1 int fact[100005];
2 int p;
3
4 void init() {
5     fact[0] = 1;
6     for(int i = 1; i <= p; i++) {
7         fact[i] = fact[i - 1] * i % p;
8     }
9 }
10
11 int inv(int x, int p) {
12     if(x == 1) return 1;
13     return (p - p / x) * inv(p % x, p) % p;
14 }
15
16 int c(int x, int y, int p) {
17     if(x < y) return 0;
18     int k = fact[x] * inv(fact[y], p) % p;
19     return k * inv(fact[x - y], p) % p;
20 }
21
22 int lucas(int x, int y, int p) {
23     if(x == 0) return 1;
24     return lucas(x / p, y / p, p) % p * c(x % p, y % p, p) % p;
25 }

```

2.10. MillerRabin.cpp

```

1 #define ull __uint128_t
2 template <class T, class POW>
3 void fastpow(T x, POW n, POW p, T &ans) {
4     for(; n; n >>= 1) {
5         if(n & 1) {
6             ans *= x;
7             ans %= p;
8         }
9         x *= x;
10        x %= p;
11    }
12 }
13 /* 輸入 x,n,p,ans 會將 ans 修改為 x^n%p
14    對整數/矩陣/不要求精度的浮點 皆有效
15    模板第一個型別是 x,ans 第二個是 n,p(應該放 LL 或 __int128)*/
16 ull pri[7] = {2, 325, 9375, 28178, 450775, 9780504, 1795265022}; /*2^64*/
17 // int p[3]={2,7,61};/*2^32*/
18 bool check(const ull x, const ull p) {
19     ull d = p - 1, ans = 1;
20     fastpow(p, d, x, ans);
21     if(ans != 1) return 1;
22     for(; !(d & 1);) {
23         d >>= 1;
24         ans = 1;
25         fastpow(p, d, x, ans);
26         if(ans == x - 1)
27             return 0;
28         else if(ans != 1)
29             return 1;
30     }
31     return 0;
32 }
33 bool miller_rabin(const ull x) {
34     if(x == 1) return 0;
35     for(auto e : pri) {
36         if(e >= x) return 1;
37         if(check(x, e)) return 0;
38     }
39     return 1;
40 }
41 }

```

2.11. Mu.cpp

```

1 vector<int> prime;
2 bitset<1000005> vis;
3 int n;
4 int mu[1000005];
5
6 void init() {
7     for(int i = 2; i <= n; i++) {
8         if(!vis[i]) {
9             prime.push_back(i);
10            mu[i] = -1;
11        }
12        for(int p : prime) {
13            if(i * p > n) break;
14            vis[i * p] = 1;

```

```

15         if(i % p == 0) {
16             mu[i * p] = 0;
17             break;
18         } else {
19             mu[i * p] = mu[i] * mu[p];
20         }
21     }
22 }
23 }

```

2.12. NTT.cpp

```

1 #include <bits/stdc++.h>
2 #define ll long long
3 using namespace std;
4
5 const int MAXN = 1000005;
6 const int MOD = 998244353, G = 3;
7 int rev[MAXN * 3];
8
9 int qpow(int x, int y) {
10     int ret = 1;
11     while(y) {
12         if(y & 1) {
13             ret *= x;
14             ret %= MOD;
15         }
16         x *= x;
17         x %= MOD;
18         y >>= 1;
19     }
20     return ret;
21 }
22
23 void ntt(int F[], int N, int sgn) {
24     int bit = __lg(N);
25     for(int i = 0; i < N; ++i) {
26         rev[i] = (rev[i >> 1] >> 1) | ((i & 1) << (bit - 1));
27         if(i < rev[i]) swap(F[i], F[rev[i]]);
28     }
29     for(int l = 1, t = 1; l < N; l <= 1, t++) {
30         int step = qpow(G, ((MOD - 1) >> t) * sgn + MOD - 1);
31         for(int i = 0; i < N; i += l << 1) {
32             for(int k = i, cur = 1; k < i + l; ++k) {
33                 int g = F[k], h = (ll)F[k + l] * cur % MOD;
34                 F[k] = (g + h) % MOD;
35                 F[k + l] = ((g - h) % MOD + MOD) % MOD;
36                 cur = (ll)cur * step % MOD;
37             }
38         }
39     }
40     if(sgn == -1) {
41         int invN = qpow(N, MOD - 2);
42         for(int i = 0; i < N; ++i) F[i] = (ll)F[i] * invN % MOD;
43     }
44 }

```

2.13. PollardRho.cpp

```

1 #include <bits/stdc++.h>
2 using namespace std;
3 #define LL long long
4 #define uLL __uint128_t
5 #define sub(a, b) ((a) < (b) ? (b) - (a) : (a) - (b))
6 template <class T, class POW>
7 void fastpow(T x, POW n, POW p, T &ans) {
8     for(; n; n >>= 1) {
9         if(n & 1) {
10             ans *= x;
11             ans %= p;
12         }
13         x *= x;
14         x %= p;
15     }
16 }
17 /*input x, n, p, ans, will modify ans to x ^ n % p
18 the first is x, ans and the second is n, p (LL or __uint128)
19 */
20 uLL pri[7] = {2, 325, 9375, 28178,
21              450775, 9780504, 1795265022}; /*2^64*/
22 // int p[3]={2,7,61};/*2^32*/
23 bool check(const uLL x, const uLL p) {
24     uLL d = x - 1, ans = 1;
25     fastpow(p, d, x, ans);
26     if(ans != 1) return 1;
27     for(; !(d & 1);) {
28         d >>= 1;
29         ans = 1;
30         fastpow(p, d, x, ans);
31         if(ans == x - 1)
32             return 0;
33         else if(ans != 1)
34             return 1;
35     }
36 }

```

```

37     return 0;
38 }
39 bool miller_rabin(const uLL x) {
40     if(x == 1) return 0;
41     for(auto e : pri) {
42         if(e >= x) return 1;
43         if(check(x, e)) return 0;
44     }
45     return 1;
46 }
47 template <class T> T gcd(T a, T b) {
48     if(!a) return b;
49     if(!b) return a;
50     if(a & b & 1) return gcd(sub(a, b), min(a, b));
51     if(a & 1) return gcd(a, b >> 1);
52     if(b & 1) return gcd(a >> 1, b);
53     return gcd(a >> 1, b >> 1) << 1;
54 }
55 /*gcd(a,b) denote gcd(a, 0) = a*/
56 mt19937 rnd(time(0));
57 template <class T> T f(T x, T c, T mod) {
58     return (((uLL)x) * x % mod + c) % mod;
59 }
60 template <class T> T rho(T n) {
61     T mod = n, x = rnd() % mod, c = rnd() % (mod - 1) + 1,
62     p = 1;
63     for(T i = 2, j = 2, d = x; ++i) {
64         x = f(x, c, mod);
65         p = ((uLL)p) * sub(x, d) % mod;
66         if(i % 127 == 0 && gcd(p, n) != 1) return gcd(p, n);
67         if(i == j) {
68             j <= 1, d = x;
69             if(gcd(p, n) != 1) return gcd(p, n);
70         }
71     }
72 }
73 template <class T> T pollard_rho(T n) {
74     if(miller_rabin(n)) return n;
75     T p = n;
76     while(p == n) p = rho(n);
77     return max(pollard_rho(p), pollard_rho(n / p));
78 }
79 int main() {
80     LL t, n, ans;
81     for(cin >> t; t--;) {
82         cin >> n;
83         ans = pollard_rho(n);
84         if(ans == n)
85             puts("Prime");
86         else
87             printf("%lld\n", ans);
88     }
89 }

```

2.14. XorBasis.cpp

```

1 #pragma GCC optimize(
2     "Ofast,fast-math,unroll-loops,no-stack-protector")
3 #include <bits/stdc++.h>
4 using namespace std;
5 #define ll long long
6 #define V vector
7 #define pb push_back
8 #define all(x) x.begin(), x.end()
9 V<ll> v;
10 ll f(ll k, ll now = 0, ll p = v.size() - 1, ll ans = 0) {
11     if(k >= 1 << p) {
12         k -= 1 << p;
13         ans = max(ans, ans ^ v[now]);
14     } else {
15         ans = min(ans, ans ^ v[now]);
16     }
17     if(!p) return ans;
18     return f(k, now + 1, p - 1, ans);
19 }
20 int main() {
21     ios::sync_with_stdio(0);
22     cin.tie(0);
23     cout.tie(0);
24     ll n, k;
25     cin >> n >> k;
26     for(ll x, i = 0; i < n; ++i) {
27         cin >> x;
28         for(ll &e : v) x = min(x, x ^ e);
29         if(x) v.pb(x);
30     }
31     sort(all(v), greater<ll>());
32     ll t = n - v.size(), a = k >> t,
33     b = k & ((1 << min(t, 20LL)) - 1), i = 0;
34     for(; a--; ++i)
35         for(ll j = 1 << t, p = f(i); j--;) cout << p << " ";
36     for(i = f(i); b--;) cout << i << " ";
37 }

```

2.15. mtt.cpp

```

1 #include <bits/stdc++.h>
2 using namespace std;
3 // https://www.luogu.com.cn/article/08nmgxd1
4 namespace poly {
5     long double const pi = acos(-1);
6     struct comp {
7         long double r, i;
8         comp() { r = i = 0; }
9         comp(long double x, long double y) { r = x, i = y; }
10        comp conj() { return comp(r, -i); }
11        friend comp operator+(comp x, comp y) {
12            return comp(x.r + y.r, x.i + y.i);
13        }
14        friend comp operator-(comp x, comp y) {
15            return comp(x.r - y.r, x.i - y.i);
16        }
17        friend comp operator*(comp x, comp y) {
18            return comp(x.r * y.r - x.i * y.i,
19                x.i * y.r + x.r * y.i);
20        }
21    };
22    typedef long long ll;
23    int r[400005];
24    comp a[400005], b[400005], c[400005], d[400005];
25    void fft(comp *f, int n, int op) {
26        for(int i = 1; i < n; i++)
27            r[i] = (r[i] >> 1) >> 1 + ((i & 1) ? (n >> 1) : 0);
28        for(int i = 1; i < n; i++)
29            if(i < r[i]) swap(f[i], f[r[i]]);
30        for(int len = 2; len <= n; len <= 1) {
31            int q = len >> 1;
32            comp wn = comp(cos(pi / q), op * sin(pi / q));
33            for(int i = 0; i < n; i += len) {
34                comp w = comp(1, 0);
35                for(int j = i; j < i + q; j++, w = w * wn) {
36                    comp d = f[j + q] * w;
37                    f[j + q] = f[j] - d;
38                    f[j] = f[j] + d;
39                }
40            }
41        }
42    }
43    void mtt(int *f, int *g, int *h, int n, int p) {
44        for(int i = 0; i < n; i++) {
45            a[i].r = (f[i] >> 15);
46            a[i].i = (f[i] & 32767);
47            c[i].r = (g[i] >> 15);
48            c[i].i = (g[i] & 32767);
49        }
50        fft(a, n, 1), fft(c, n, 1);
51        for(int i = 1; i < n; i++) b[i] = a[n - i].conj();
52        b[0] = a[0].conj();
53        for(int i = 1; i < n; i++) d[i] = c[n - i].conj();
54        d[0] = c[0].conj();
55        for(int i = 0; i < n; i++) {
56            comp aa = (a[i] + b[i]) * comp(0.5, 0);
57            comp bb = (a[i] - b[i]) * comp(0, -0.5);
58            comp cc = (c[i] + d[i]) * comp(0.5, 0);
59            comp dd = (c[i] - d[i]) * comp(0, -0.5);
60            a[i] = aa * cc + comp(0, 1) * (aa * dd + bb * cc);
61            b[i] = bb * dd;
62        }
63        fft(a, n, -1), fft(b, n, -1);
64        for(int i = 0; i < n; i++) {
65            int aa = (ll)(a[i].r / n + 0.5) % p,
66                bb = (ll)(a[i].i / n + 0.5) % p,
67                cc = (ll)(b[i].r / n + 0.5) % p;
68            h[i] = ((ll) * aa * (1 << 30) + ll * bb * (1 << 15) +
69                cc) %
70                p +
71                p;
72        }
73    }
74    // namespace poly
75    using namespace poly;
76    int f[400005], g[400005], h[400005];
77    // 400005 is 2 * (n + m)
78    int main() {
79        int n, m, p;
80        scanf("%d%d%d", &n, &m, &p);
81        for(int i = 0; i <= n; i++) scanf("%d", &f[i]);
82        for(int i = 0; i <= m; i++) scanf("%d", &g[i]);
83        int lim = 1;
84        while(lim <= (n + m)) lim <= 1;
85        mtt(f, g, h, lim, p);
86        for(int i = 0; i <= n + m; i++) printf("%d ", h[i]);
87        return 0;
88    }
89 }

```

3. String

3.1. Booth.cpp

```

1 #define V vector
2 string booth(string s) {
3     s += s;
4     int n = s.size(), k = 0;
5     V<int> f(n, -1);
6     for(int i = 1; i < n; ++i) {
7         int j = f[i - k - 1];
8         for(; j >= 0 && s[j + k + 1] != s[i]; j = f[j])
9             if(s[i] < s[j + k + 1]) k = i - j - 1;
10            if(s[i] != s[j + k + 1]) {
11                if(s[i] < s[k]) k = i;
12                f[i - k] = -1;
13            } else
14                f[i - k] = j + 1;
15        }
16        return s.substr(k, s.size() >> 1);
17    }
18    // 給出循環排列後最小字典序的解

```

3.2. KMP.cpp

```

1 string s, t;
2 int pmt[1000005];
3
4 void init() {
5     for(int i = 1, j = 0; i < t.size(); i++) {
6         while(j && t[j] ^ t[i]) {
7             j = pmt[j - 1];
8         }
9         if(t[j] == t[i]) j++;
10        pmt[i] = j;
11    }
12 }
13
14 int kmp(string s) {
15     int ret = 0;
16     for(int i = 0, j = 0; i < s.size(); i++) {
17         while(j && s[i] ^ t[j]) {
18             j = pmt[j - 1];
19         }
20         if(s[i] == t[j]) {
21             j++;
22         }
23         if(j == t.size()) {
24             ret++;
25             j = pmt[j - 1];
26         }
27     }
28     return ret;
29 }

```

3.3. LongestPalindrome.cpp

```

1 #include <bits/stdc++.h>
2 #define int long long
3 using namespace std;
4
5 string s;
6 string t;
7 int n;
8 int d[2000005];
9 int ans = 0;
10
11 signed main() {
12     cin >> t;
13     n = t.size();
14     for(int i = 0; i < 2 * n + 1; i++) {
15         if(i & 1 ^ 1) {
16             s += '0';
17         } else {
18             s += t[i / 2];
19         }
20     }
21     n = s.size();
22     d[0] = 1;
23     for(int i = 0, l = 0, r = 0; i < n; i++) {
24         if(i > r) {
25             d[i] = 1;
26             bool a = i + d[i] < n;
27             bool b = i - d[i] >= 0;
28             bool c = (s[i + d[i]] == s[i - d[i]]);
29             while (a && b && c) {
30                 d[i]++;
31                 a = i + d[i] < n;
32                 b = i - d[i] >= 0;
33                 c = (s[i + d[i]] == s[i - d[i]]);
34             }
35             l = i - d[i] + 1;
36             r = i + d[i] - 1;
37         }
38     }
39 }

```



```

37     } else {
39         int j = l + r - i;
41         if(j - d[j] + 1 > l) {
43             d[i] = d[j];
45         } else {
47             d[i] = r - i + 1;
49             a = i + d[i] < n;
51             b = i - d[i] >= 0;
53             c = (s[i + d[i]] == s[i - d[i]]);
55             while(a && b && c) {
57                 d[i]++;
59                 a = i + d[i] < n;
61                 b = i - d[i] >= 0;
63                 c = (s[i + d[i]] == s[i - d[i]]);
65             }
67             l = i - d[i] + 1;
69             r = i + d[i] - 1;
71         }
73     }
75     // cout << d[i] << " ";
77     if(d[i] > d[ans]) {
79         ans = i;
81     }
83 }
85 for(int i = ans - d[ans] + 1; i < ans + d[ans]; i++) {
87     if(s[i] ^ '0') {
89         cout << s[i];
91     }
93 }
95 }

```

3.4. Z.cpp

```

1 #include <bits/stdc++.h>
2 #define int long long
3 using namespace std;
4
5 string s, t;
6 int ans = 0;
7
8 int z[2000005];
9
10 signed main() {
11     ios::sync_with_stdio(0);
12     cin.tie(0);
13     cout.tie(0);
14     cin >> s >> t;
15     s = t + '0' + s;
16     int n, m;
17     n = s.size();
18     m = t.size();
19     for(int i = 0, l = 0, r = 0; i < n; i++) {
21         if(z[i - l] < r - i + 1) {
23             z[i] = z[i - l];
25         } else {
27             z[i] = max(r - i + 1, (int)0);
29             while(i + z[i] < n && s[i + z[i]] == s[z[i]]) {
31                 z[i]++;
33             }
35             l = i;
37             r = i + z[i] - 1;
39             if(z[i] == m) {
41                 ans++;
43             }
45         }
47     }
49     cout << ans;
51 }

```

4. Graph

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

```

1 #include <bits/stdc++.h>
2 #define int long long
3 using namespace std;
4
5 int n, m;
6 vector<int> v[200005];
7 int d[200005];
8 int low[200005];
9 int cnt = 0;
10 int now = 0;
11 int scc[200005];
12 stack<int> s;
13 int op[200005];
14 vector<int> v2[200005];
15 int ind[200005];
16 queue<int> q;
17 int ans[200005];
18
19 int no(int x) {

```

```

21     if(x > m) return x - m;
22     return x + m;
23 }
24
25 void dfs(int x) {
26     d[x] = low[x] = ++cnt;
27     s.push(x);
28     for(int i : v[x]) {
29         if(scc[i]) continue;
30         if(d[i]) {
31             low[x] = min(low[x], d[i]);
32         } else {
33             dfs(i);
34             low[x] = min(low[x], low[i]);
35         }
36     }
37     if(d[x] == low[x]) {
38         now++;
39         while(!s.empty()) {
40             int k = s.top();
41             s.pop();
42             scc[k] = now;
43             if(k == x) break;
44         }
45     }
46 }
47
48 signed main() {
49     ios::sync_with_stdio(0);
50     cin.tie(0);
51     cout.tie(0);
52     cin >> n >> m;
53     while(n--) {
54         char a, b;
55         int x, y;
56         cin >> a >> x >> b >> y;
57         if(a == '-') x = no(x);
58         if(b == '-') y = no(y);
59         v[no(x)].push_back(y);
60         v[no(y)].push_back(x);
61     }
62     for(int i = 1; i <= 2 * m; i++) {
63         if(!d[i]) {
64             dfs(i);
65         }
66     }
67     for(int i = 1; i <= m; i++) {
68         if(scc[i] ^ scc[i + m]) {
69             op[scc[i]] = scc[i + m];
70             op[scc[i + m]] = scc[i];
71         } else {
72             cout << "IMPOSSIBLE";
73             exit(0);
74         }
75     }
76     for(int i = 1; i <= 2 * m; i++) {
77         for(int j : v[i]) {
78             if(scc[i] ^ scc[j]) {
79                 v2[scc[j]].push_back(scc[i]);
80                 ind[scc[i]]++;
81             }
82         }
83     }
84     for(int i = 1; i <= now; i++) {
85         if(!ind[i]) {
86             q.push(i);
87         }
88     }
89     while(!q.empty()) {
90         int k = q.front();
91         q.pop();
92         if(!ans[k]) {
93             ans[k] = 1;
94             ans[op[k]] = 2;
95         }
96         for(int i : v2[k]) {
97             ind[i]--;
98             if(!ind[i]) {
99                 q.push(i);
100             }
101         }
102     }
103     for(int i = 1; i <= m; i++) {
104         if(ans[scc[i]] == 1) {
105             cout << "+ ";
106         } else {
107             cout << "- ";
108         }
109     }

```

4.2. Dijkstra.cpp

```

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

7 void dijkstra(int x, int *dis, vector<pair<int, int>> *v,
  bitset<100005> &vis) {
9     priority_queue<pair<int, int>, vector<pair<int, int>>,
  greater<pair<int, int>>>
11         pq;
  memset(dis, 0x3f, sizeof(dis1));
  vis.reset();
  dis[x] = 0;
  pq.push({0, x});
  while(!pq.empty()) {
17         pair<int, int> now = pq.top();
  pq.pop();
19         if(vis[now.second]) continue;
  vis[now.second] = 1;
21         for(auto [i, w] : v[now.second]) {
  if(vis[i]) continue;
23         if(dis[now.second] + w < dis[i]) {
  dis[i] = dis[now.second] + w;
25         pq.push({dis[i], i});
  }
27     }
29 }

```

4.3. Dinic.cpp

```

1 #include <bits/stdc++.h>
  using namespace std;
3 #define ll long long
  const ll inf = 8e18;
5 #define N 505
  #define pb push_back
7 struct pp {
  int from, to;
  ll cap, flow;
9 };
  int t, lvl[N], p[N];
  vector<int> g[N];
  vector<pp> edge;
11 int bfs(int s) {
  queue<int> q;
  for(q.push(s), lvl[s] = 1; !q.empty(); q.pop()) {
17         int u = q.front();
  for(int e : g[u]) {
19             int v = edge[e].to;
  if(lvl[v] || !edge[e].flow) continue;
21             lvl[v] = lvl[u] + 1;
  q.push(v);
23         }
25     }
  return lvl[t];
27 }
  ll dfs(int u, ll f = inf) {
  if(u == t || !f) return f;
29     ll ans = 0;
  for(int &i = p[u]; i < g[u].size(); ++i) {
31         pp &e = edge[g[u][i]], &b = edge[g[u][i] ^ 1];
  if(lvl[e.to] == lvl[u] + 1) {
33             ll c = dfs(e.to, min(e.flow, f));
  e.flow -= c;
35             b.flow += c;
  f -= c;
37             ans += c;
  }
39     }
  return ans;
41 }
  ll dinic(int s) {
  ll ans = 0;
43     for(; bfs(s); memset(lvl, 0, sizeof lvl))
  for(ll k; k = (memset(p, 0, sizeof(p)), dfs(s));)
45         ans += k;
47     return ans;
49 }
  int main() {
  ios::sync_with_stdio(0); //任意圖上複雜度  $V^2E$ 
51     cin.tie(0); //邊容量為 1 時  $\min(V^{\frac{2}{3}}, \sqrt{E})E$ 
  cout.tie(0); //二分圖最大匹配  $E\sqrt{V}$  (是下面這行的特例)
53     int n, m, cnt = 0; //邊容量 1 時, 除源匯點滿足入度或出度
  for(cin >> n >> m; m--;) { //都是 1, 則為  $E\sqrt{V}$ 
55         int u, v;
  ll f;
57         cin >> u >> v >> f;
  g[u].pb(cnt++);
59         g[v].pb(cnt++);

```

```

61         edge.pb({u, v, f, f});
  edge.pb({v, u, 0, 0});
63     }
  t = n;
  cout << dinic(1);
65 }

```

4.4. MaximumFlow.cpp

```

1 #include <bits/stdc++.h>
  #define int long long
3 using namespace std;

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

11 bool bfs() {
  memset(head, 0, sizeof(head));
  memset(lv, 0, sizeof(lv));
13     queue<int> q;
  q.push(1);
  while(!q.empty()) {
17         int now = q.front();
  q.pop();
19         if(now == n) continue;
  for(int i : v[now]) {
21             if(i != 1 && c[now][i] && !lv[i]) {
  lv[i] = lv[now] + 1;
23             q.push(i);
  }
25     }
27 }
  return lv[n];
29 }

31 int dfs(int x, int flow) {
  int ret = 0;
33     if(x == n) return flow;
  for(int i = head[x]; i < v[x].size(); i++) {
35         int y = v[x][i];
  head[x] = y;
37         if(c[x][y] && lv[y] == lv[x] + 1) {
  int d = dfs(y, min(flow, c[x][y]));
39         flow -= d;
  c[x][y] -= d;
41         c[y][x] += d;
  ret += d;
43     }
45     return ret;
47 }

  signed main() {
49     cin >> n >> m;
  while(m--) {
51         int x, y, z;
  cin >> x >> y >> z;
53         if(c[x][y] || c[y][x]) {
  c[x][y] += z;
55         continue;
  }
57         v[x].push_back(y);
  v[y].push_back(x);
59         c[x][y] = z;
  }
61     while(bfs()) {
  ans += dfs(1, INT_MAX);
63     }
  cout << ans;
65 }

```

4.5. SCC.cpp

```

1 int n, m;
  vector<int> v[100005];
3 int d[100005];
  int low[100005];
5 int cnt = 0;
  stack<int> s;
7 int scc[100005];
  int now = 0;
9 void dfs(int x) {
11     d[x] = low[x] = ++cnt;
  s.push(x);
13     for(int i : v[x]) {
  if(scc[i]) continue;
15         if(d[i]) {
  low[x] = min(low[x], d[i]);

```

```

17     } else {
18         dfs(i);
19         low[x] = min(low[x], low[i]);
20     }
21 }
22 if(d[x] == low[x]) {
23     now++;
24     while(!s.empty()) {
25         int k = s.top();
26         s.pop();
27         scc[k] = now;
28         if(k == x) break;
29     }
30 }
31 }

```

4.6. VBCC.cpp

```

1 #include <bits/stdc++.h>
2 using namespace std;
3 #define pb push_back
4 #define pii pair<int, int>
5 #define N 100005
6 vector<int> adj[N], bcc[N];
7 stack<int> st;
8 int dfn[N], low[N], tag, bc, root;
9 bitset<N> ap;
10 void dfs(int now, int par = -1) {
11     st.push(now);
12     low[now] = dfn[now] = ++tag;
13     int f = 0;
14     for(int e : adj[now] | views::reverse) {
15         if(e == par) continue;
16         if(!dfn[e]) {
17             dfs(e, now), low[now] = min(low[now], low[e]);
18             if(low[e] >= dfn[now]) {
19                 if(++f > 1 || now != root) ap[now] = 1;
20                 ++bc;
21                 for(; st.top() != now; st.pop())
22                     bcc[bc].pb(st.top());
23                 bcc[bc].pb(now);
24             }
25         } else
26             low[now] = min(low[now], dfn[e]);
27     }
28 }
29 int main() {
30     int n, m, u, v;
31     cin >> n >> m;
32     vector<pii> g(m);
33     for(auto &[u, v] : g)
34         cin >> u >> v, adj[u].pb(v), adj[v].pb(u);
35     for(root = 1; root <= n; ++root)
36         if(!dfn[root]) dfs(root);
37     int ans = 0;
38     for(int i : views::iota(1) | views::take(n))
39         if(ap[i]) ++ans;
40     cout << ans << "\n";
41     for(int i : views::iota(1) | views::take(n))
42         if(ap[i]) cout << i << " ";
43 }

```

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

```

1 #include <bits/stdc++.h>
2 #define int long long
3 using namespace std;
4
5 int n, q;
6 int a[200005];
7 int r[200005];
8 int d[200005];
9 int cycle[200005];
10 int len[200005];
11 int cnt = 0;
12 vector<int> v[200005];
13 bitset<200005> vis1;
14 bitset<200005> vis2;
15
16 void findcycle(int x) {
17     while(!vis1[x]) {
18         vis1[x] = 1;
19         x = a[x];
20     }
21     cnt++;
22     cycle[x] = cnt;
23     r[x] = 0;
24     len[cnt] = 1;
25     int temp = a[x];
26     while(temp ^ x) {
27         r[temp] = len[cnt];
28         len[cnt]++;
29         cycle[temp] = cnt;

```

```

temp = a[temp];
}
}
33 void dfs(int x) {
34     if(vis2[x]) return;
35     vis2[x] = 1;
36     for(int i : v[x]) {
37         dfs(i);
38     }
39 }
40
41 void dfs2(int x) {
42     if(cycle[x] || d[x]) return;
43     dfs2(a[x]);
44     d[x] = d[a[x]] + 1;
45     r[x] = r[a[x]];
46     cycle[x] = cycle[a[x]];
47 }
48
49 signed main() {
50     ios::sync_with_stdio(0);
51     cin.tie(0);
52     cout.tie(0);
53     cin >> n;
54     for(int i = 1; i <= n; i++) {
55         cin >> a[i];
56         v[i].push_back(a[i]);
57         v[a[i]].push_back(i);
58     }
59     for(int i = 1; i <= n; i++) {
60         if(!vis2[i]) {
61             findcycle(i);
62             dfs(i);
63         }
64     }
65     for(int i = 1; i <= n; i++) {
66         if(!cycle[i] && !r[i]) {
67             dfs2(i);
68         }
69     }
70     for(int i = 1; i <= n; i++) {
71         cout << d[i] + len[cycle[i]] << " ";
72     }
73 }

```

5. DP

5.1. CHO.cpp

```

1 struct line {
2     int a, b;
3     int y(int x) { return a * x + b; }
4 };
5
6 struct CHO {
7     deque<line> dq;
8     int intersect(line x, line y) {
9         int d1 = x.b - y.b;
10        int d2 = y.a - x.a;
11        return d1 / d2;
12    }
13    bool check(line x, line y, line z) {
14        int I12 = intersect(x, y);
15        int I23 = intersect(y, z);
16        return I12 < I23;
17    }
18    void insert(int a, int b) {
19        if(!dq.empty() && a == dq.back().a) return;
20        while(dq.size() >= 2 &&
21            !check(dq[dq.size() - 2], dq[dq.size() - 1],
22                {a, b})) {
23            dq.pop_back();
24        }
25        dq.push_back({a, b});
26    }
27    void update(int x) {
28        while(dq.size() >= 2 && dq[0].y(x) >= dq[1].y(x)) {
29            dq.pop_front();
30        }
31    }
32    int query(int x) {
33        update(x);
34        return dq.front().y(x);
35    }
36 };

```

5.2. Li-Chao-SegmentTree.cpp

```

1 struct line {
2     int a, b = 1000000000000000;

```

```

3   int y(int x) { return a * x + b; }
};

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

11 void update(line ins, int l = 1, int r = 1e6, int index = 1) {
13     if(l == r) {
15         if(ins.y(l) < tree[index].y(l)) {
17             tree[index] = ins;
19             return;
21         }
23         int mid = (l + r) >> 1;
25         if(tree[index].a < ins.a) swap(tree[index], ins);
27         if(tree[index].y(mid) > ins.y(mid)) {
29             swap(tree[index], ins);
31             update(ins, l, mid, index << 1);
33         } else {
35             update(ins, mid + 1, r, index << 1 | 1);
37         }
39     }
}

29 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

```

1   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
3   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

```

1   #include <bits/stdc++.h>
   using namespace std;
3   #define ll long long
   #define pb push_back
5   #define pll pair<int, int>
   #define pdd pair<double, double>
7   #define pll pair<ll, ll>
   #define F first
9   #define S second
   #define eps 1e-6
11  int sign(double x) {
   return fabs(x) < eps ? 0 : x > 0 ? 1 : -1;
13 }
   int sign(ll x) { return !x ? 0 : x > 0 ? 1 : -1; }
15 template <typename T1, typename T2>
   istream &operator>>(istream &s, pair<T1, T2> &p) {
17     auto &a, &b = p;
   s >> a >> b;
19     return s;
21 }
   template <typename T1, typename T2>
   ostream &operator<<(ostream &s, const pair<T1, T2> &p) {
23     auto &a, &b = p;
   s << a << " " << b;
25     return s;
27 }
   pll operator+(const pll a, const pll b) {
   return {a.F + b.F, a.S + b.S};
29 }
   pll operator-(const pll a, const pll b) {
   return {a.F - b.F, a.S - b.S};
31 }
   pll operator*(const pll a) { return {-a.F, -a.S}; }
   pll operator*(const pll a, const pll b) {
33     return {(ll)a.F * b.F, (ll)a.S * b.S};
35 }
   pdd operator/(const pll a, const double x) {
   return {a.F / x, a.S / x};
37 }

```

```

pdd operator*(const pll a, const double x) {
   return {a.F * x, a.S * x};
41 }
43 pdd operator*(const double x, const pll a) {
   return {a.F * x, a.S * x};
45 }
   // 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
47 ll len2(pll p) {
   return (ll)p.F * p.F + (ll)p.S * p.S;
49 } // 1 vector
   double len(pll p) { return sqrt((double)len2(p)); }
51 ll cross(pll a, pll b) {
   return (ll)a.F * b.S - (ll)a.S * b.F;
53 } // 2 vector
   ll cross(pll p1, pll p2, pll p3) {
   return cross(p2 - p1, p3 - p1);
55 } //(b-a) cross (c-a)
57 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);
59 } //(b-a) dot (c-a)
61 ll ori(pll p1, pll p2, pll p3) {
   return sign(cross(p1, p2, p3));
63 } // normalize to {-1,0,1} (b-a) cross (c-a)
   bool btw(pll p1, pll p2, pll p3) {
65     return ori(p3, p1, p2) == 0 && dot(p3, p1, p2) <= 0;
   } // p3 btween p1,p2
67 bool banana(pll p1, pll p2, pll p3,
   pll p4) { // 問兩線段是否香蕉
69     if(btw(p1, p2, p3) || btw(p1, p2, p4) || btw(p3, p4, p1) ||
   btw(p3, p4, p2))
71         return true;
   return ori(p1, p2, p3) * ori(p1, p2, p4) < 0 &&
73     ori(p3, p4, p1) * ori(p3, p4, p2) < 0;
75 }
   pdd banana_point(pll p1, pll p2, pll p3,
   pll p4) { // 分點，算是無限延伸直線的交點
77     // 平行的時候 undefined
   return cross(p2 - p1, p4 - p1) /
79     ((double)cross(p2 - p1, p4 - p3) * p3 -
   cross(p2 - p1, p3 - p1) /
81     ((double)cross(p2 - p1, p4 - p3) * p4;
83 }
   pdd proj(pll p1, pll p2, pll p3) {
   return dot(p1, p2, p3) / ((double)len2(p2 - p1) * (p2 - p1));
85 }
   double min_dis(pll p1, pll p2,
   pll p3) { // min distance of p3 to segment p1,p2
87     if(dot(p1, p2, p3) < 0 || dot(p2, p1, p3) < 0)
   return min(len(p3 - p1), len(p3 - p2));
89     return abs(cross(p1, p2, p3)) / len(p2 - p1);
91 }
   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
93     // 注意是兩倍才可以 ll 避免浮點數
   int n = v.size() - 1;
   ll ans = 0;
95     for(int i = 0; i < n; ++i) ans += cross(v[i], v[i + 1]);
   return abs(ans);
97 }
   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}
101     int n = v.size() - 1, ans = 1;
   for(int i = 0; i < n; ++i)
103         if(btw(v[i], v[i + 1], p)) return 0;
   for(int i = 0; i < n; ++i)
105         if(banana(v[i], v[i + 1], p, {(ll)2e9 + 7, p.S + 1LL}))
   ans -= -1;
107     // 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}
109     // 必須很遠，保證在多邊形外
   return ans;
111 }
   void solve() {
113     int n;
   cin >> n;
   vector<pll> v(n);
   for(pll &e : v) cin >> e;
115     v.pb(v[0]);
   ll ans = area2(v) + 2, ans2 = 0;
   for(int i = 0; i < n; ++i) {
117         if(v[i].F == v[i + 1].F)
   ans2 += abs(v[i].S - v[i + 1].S);
119         else if(v[i].S == v[i + 1].S)
   ans2 += abs(v[i].F - v[i + 1].F);
121         else
   ans2 += gcd(abs(v[i].F - v[i + 1].F),
   abs(v[i].S - v[i + 1].S));
123     }
   cout << (ans - ans2) / 2 << " " << ans2;
125 }
127 int main() {

```

```

133     int t = 1;
134     // cin>>t;
135     for(; t--;) {
136         solve();
137     }
138 }

```

6.2. ConvexHull.cpp

```

1 #include <bits/stdc++.h>
2 #define int long long
3 #define fastio
4     ios_base::sync_with_stdio(0);
5     cin.tie(0);
6     cout.tie(0);
7
8 using namespace std;
9
10 template <typename T>
11 pair<T, T> operator-(pair<T, T> a, pair<T, T> b) {
12     return make_pair(a.first - b.first, a.second - b.second);
13 }
14
15 template <typename T> T cross(pair<T, T> a, pair<T, T> b) {
16     return a.first * b.second - a.second * b.first;
17 }
18
19 template <typename T>
20 vector<pair<T, T>> getCH(vector<pair<T, T>> v) {
21     int n = v.size();
22     sort(v.begin(), v.end());
23     vector<pair<T, T>> hull;
24     for(int i = 0; i < 2; i++) {
25         int t = hull.size();
26         for(auto x : v) {
27             while(hull.size() - t >= 2 &&
28                 cross(hull[hull.size() - 1] -
29                     hull[hull.size() - 2],
30                     x - hull[hull.size() - 2]) <= 0)
31                 hull.pop_back();
32             hull.push_back(x);
33         }
34         hull.pop_back();
35         reverse(v.begin(), v.end());
36     }
37     return hull;
38 }

```

6.3. Inside.cpp

```

1 int inside(point p) {
2     int ans = 0;
3     for(int i = 1; i <= n; i++) {
4         if(onseg(a[i], a[i + 1], {p.x, p.y})) {
5             return -1;
6         }
7         if(intersect({p.x, p.y}, {INF, p.y}, a[i], a[i + 1])) {
8             ans ^= 1;
9         }
10        point temp = a[i].y > a[i + 1].y ? a[i] : a[i + 1];
11        if(temp.y == p.y && temp.x > p.x) {
12            ans ^= 1;
13        }
14    }
15    return ans;
16 }

```

6.4. Intersect.cpp

```

1 struct point {
2     int x, y;
3     point operator+(point b) { return {x + b.x, y + b.y}; }
4     point operator-(point b) { return {x - b.x, y - b.y}; }
5     int operator*(point b) { return x * b.x + y * b.y; }
6     int operator^(point b) { return x * b.y - y * b.x; }
7 };
8
9 bool onseg(point x, point y, point z) {
10     return ((x - z) ^ (y - z)) == 0 && (x - z) * (y - z) <= 0;
11 }
12
13 int dir(point x, point y) {
14     int k = x ^ y;
15     if(k == 0) return 0;
16     if(k > 0) return 1;
17     return -1;
18 }
19
20 bool intersect(point x, point y, point z, point w) {
21     if(onseg(x, y, z) || onseg(x, y, w)) return 1;
22     if(onseg(z, w, x) || onseg(z, w, y)) return 1;
23     if(dir(y - x, z - x) * dir(y - x, w - x) == -1 &&

```

```

24     dir(z - w, x - w) * dir(z - w, y - w) == -1) {
25         return 1;
26     }
27     return 0;
28 }

```

6.5. MinimumEuclideanDistance.cpp

```

1 #include <bits/stdc++.h>
2 #define int long long
3 #define pii pair<int, int>
4 using namespace std;
5
6 int n;
7 vector<pair<int, int>> v;
8 set<pair<int, int>> s;
9 int dd = LONG_LONG_MAX;
10
11 int dis(pii x, pii y) {
12     return (x.first - y.first) * (x.first - y.first) +
13         (x.second - y.second) * (x.second - y.second);
14 }
15
16 signed main() {
17     ios::sync_with_stdio(0);
18     cin.tie(0);
19     cout.tie(0);
20     cin >> n;
21     for(int i = 0; i < n; i++) {
22         int x, y;
23         cin >> x >> y;
24         x += 1000000000;
25         v.push_back({x, y});
26     }
27     sort(v.begin(), v.end());
28     int l = 0;
29     for(int i = 0; i < n; i++) {
30         int d = ceil(sqrt(dd));
31         while(l < i && v[l].first - v[i].first > d) {
32             s.erase({v[l].second, v[l].first});
33             l++;
34         }
35         auto x = s.lower_bound({v[i].second - d, 0});
36         auto y = s.upper_bound({v[i].second + d, 0});
37         for(auto it = x; it != y; it++) {
38             dd = min(dd, dis({it->second, it->first}, v[i]));
39         }
40         s.insert({v[i].second, v[i].first});
41     }
42     cout << dd;
43 }

```

7. Tree

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

```

1 #include <bits/stdc++.h>
2 #define int long long
3 using namespace std;
4
5 int tree[800005];
6
7 int n, q;
8 int a[200005];
9 int st[200005];
10 int tp[200005];
11 int p[200005];
12 int cnt = 0;
13 int d[200005];
14 int si[200005];
15 vector<int> v[200005];
16 int b[200005];
17
18 void build(int l = 1, int r = n, int index = 1) {
19     if(l == r) {
20         tree[index] = b[l];
21         return;
22     }
23     int mid = (l + r) >> 1;
24     build(l, mid, index << 1);
25     build(mid + 1, r, index << 1 | 1);
26     tree[index] = max(tree[index << 1], tree[index << 1 | 1]);
27 }
28
29 int query(int L, int R, int l = 1, int r = n, int index = 1) {
30     if(L == l && R == r) {
31         return tree[index];
32     }
33     int mid = (l + r) >> 1;
34     if(R <= mid) {

```



```

35     return query(L, R, l, mid, index << 1);
36 }
37 if(L > mid) {
38     return query(L, R, mid + 1, r, index << 1 | 1);
39 }
40 return max(query(L, mid, l, mid, index << 1),
41           query(mid + 1, R, mid + 1, r, index << 1 | 1));
42 }
43
44 void modify(int x, int val, int l = 1, int r = n,
45            int index = 1) {
46     if(l == r) {
47         tree[index] = val;
48         return;
49     }
50     int mid = (l + r) >> 1;
51     if(x <= mid) {
52         modify(x, val, l, mid, index << 1);
53     } else {
54         modify(x, val, mid + 1, r, index << 1 | 1);
55     }
56     tree[index] = max(tree[index << 1], tree[index << 1 | 1]);
57 }
58
59 void dfs(int x, int pre) {
60     si[x] = 1;
61     for(int i : v[x]) {
62         if(i == pre) continue;
63         p[i] = x;
64         d[i] = d[x] + 1;
65         dfs(i, x);
66         si[x] += si[i];
67     }
68 }
69
70 void dfs2(int x, int pre, int t) {
71     tp[x] = t;
72     st[x] = ++cnt;
73     int ma = 0;
74     for(int i : v[x]) {
75         if(i == pre) continue;
76         if(si[i] > si[ma]) {
77             ma = i;
78         }
79     }
80     if(!ma) return;
81     dfs2(ma, x, t);
82     for(int i : v[x]) {
83         if(i == pre || i == ma) {
84             continue;
85         }
86         dfs2(i, x, i);
87     }
88 }
89
90 int f(int x, int y) {
91     int ret = 0;
92     while(tp[x] ^ tp[y]) {
93         if(d[tp[x]] < d[tp[y]]) {
94             swap(x, y);
95         }
96         ret = max(ret, query(st[tp[x]], st[x]));
97         x = p[tp[x]];
98     }
99     if(d[x] > d[y]) swap(x, y);
100    ret = max(ret, query(st[x], st[y]));
101    return ret;
102 }
103
104 signed main() {
105     ios::sync_with_stdio(0);
106     cin.tie(0);
107     cout.tie(0);
108     cin >> n >> q;
109     for(int i = 1; i <= n; i++) {
110         cin >> a[i];
111     }
112     for(int i = 1; i < n; i++) {
113         int x, y;
114         cin >> x >> y;
115         v[x].push_back(y);
116         v[y].push_back(x);
117     }
118     dfs(1, 0);
119     dfs2(1, 0, 1);
120     for(int i = 1; i <= n; i++) {
121         b[st[i]] = a[i];
122     }
123     build();
124     while(q--) {
125         int mode, x, y;
126         cin >> mode >> x >> y;
127         if(mode == 1) {

```

```

128             modify(st[x], y);
129         } else {
130             cout << f(x, y) << " ";
131         }
132     }
133 }

```

7.2. LCA.cpp

```

1  #include <bits/stdc++.h>
2  #define int long long
3  using namespace std;
4
5  int n, q;
6  int a[200005][21];
7  int d[200005];
8  vector<int> v[200005];
9
10 void init() {
11     for(int j = 1; j < 21; j++) {
12         for(int i = 1; i <= n; i++) {
13             a[i][j] = a[a[i][j - 1]][j - 1];
14         }
15     }
16 }
17
18 void dfs(int x, int pre) {
19     for(int i : v[x]) {
20         if(i == pre) {
21             continue;
22         }
23         a[i][0] = x;
24         d[i] = d[x] + 1;
25         dfs(i, x);
26     }
27 }
28
29 int lca(int x, int y) {
30     while(d[x] ^ d[y]) {
31         if(d[x] < d[y]) {
32             swap(x, y);
33         }
34         int k = __lg(d[x] - d[y]);
35         x = a[x][k];
36     }
37     if(x == y) {
38         return x;
39     }
40     for(int i = 20; i >= 0; i--) {
41         if(a[x][i] != a[y][i]) {
42             x = a[x][i];
43             y = a[y][i];
44         }
45     }
46     return a[x][0];
47 }
48
49 signed main() {
50     ios::sync_with_stdio(0);
51     cin.tie(0);
52     cout.tie(0);
53     cin >> n >> q;
54     for(int i = 1; i < n; i++) {
55         int x, y;
56         cin >> x >> y;
57         v[x].push_back(y);
58         v[y].push_back(x);
59     }
60     dfs(1, 0);
61     init();
62     while(q--) {
63         int x, y;
64         cin >> x >> y;
65         int k = lca(x, y);
66         cout << (d[x] + d[y] - 2 * d[k]) << "\n";
67     }
68 }

```

8. Misc

8.1. BigNum(luoguP1005).cpp

```

1  // 洛谷 P1005
2  #include <bits/stdc++.h>
3  using namespace std;
4  #define N 85
5  #define LL long long
6  #define pii pair<int, int>
7  #define F first
8  #define S second
9  struct num {
10     const static LL base = 1000000000LL; // base 1e9

```

```

11  LL p[505], len;
12  num() {
13      memset(p, 0, sizeof(p));
14      len = 0;
15  }
16  num(LL x) {
17      memset(p, 0, sizeof(p));
18      len = 0;
19      for(p[len++] = x; p[len - 1] >= base; ++len)
20          p[len] = p[len - 1] / base, p[len - 1] %= base;
21  }
22  num operator=(LL x) {
23      memset(p, 0, sizeof(p));
24      len = 0;
25      for(p[len++] = x; p[len - 1] >= base; ++len)
26          p[len] = p[len - 1] / base, p[len - 1] %= base;
27      return *this;
28  }
29  num max(const num &b) {
30      if(len != b.len) return len > b.len ? *this : b;
31      for(int i = len; i--;)
32          if(p[i] != b.p[i]) return p[i] > b.p[i] ? *this : b;
33      return *this;
34  }
35  num operator+(const num &b) {
36      num c;
37      LL x = 0;
38      for(LL i = c.len; i < len || i < b.len; ++i) {
39          c.p[i] = p[i] + b.p[i] + x;
40          x = c.p[i] / base;
41          c.p[i] %= base;
42      }
43      if(x) c.p[c.len++] = x;
44      return c;
45  }
46  num operator*(LL b) {
47      num c;
48      c.len = len;
49      LL x = 0;
50      for(LL i = 0; i < len; ++i) {
51          c.p[i] = p[i] * b + x;
52          x = c.p[i] / base;
53          c.p[i] %= base;
54      }
55      for(; x; x /= base) c.p[c.len++] = x % base;
56      return c;
57  }
58  } dp[N][N], ans;
59  ostream &operator<<(ostream &s, num a) {
60      if(!a.len) return s << "0";
61      s << a.p[a.len - 1];
62      for(int i = a.len - 1; i--;) {
63          if(!a.p[i])
64              s << "000000000";
65          else {
66              for(int k = 10; k * a.p[i] < (LL)1e9; k *= 10)
67                  s << "0";
68              s << a.p[i];
69          }
70      }
71      return s;
72  }
73  LL a[N];
74  int main() {
75      ios::sync_with_stdio(0);
76      cin.tie(0);
77      cout.tie(0);
78      int n, m, i, j;
79      for(cin >> n >> m; n--;) {
80          for(i = 0; i < m; ++i) cin >> a[i];
81          for(i = 0; i < m; ++i)
82              for(j = 0; j < m; ++j) dp[i][j] = 0;
83          for(i = 0; i < m; ++i) dp[i][i] = a[i] << 1;
84          for(j = 1; j < m; ++j)
85              for(i = 0; i + j < m; ++i)
86                  dp[i][i + j] =
87                      (dp[i][i + j - 1] + a[i + j]) *
88                      .max(dp[i + 1][i + j] + a[i]) *
89                      2;
90          ans = ans + dp[0][m - 1];
91      }
92      cout << ans;
93  }

```

8.2. Tri-search.cpp

```

1  #include <bits/stdc++.h>
2  using namespace std;
3  int n;
4  double a[15], x, y;
5
6  double get(double x) {
7      double ret = 0;

```

```

9      double k = 1;
10     for(int i = 0; i <= n; i++) {
11         ret += k * a[i];
12         k *= x;
13     }
14     return -ret;
15 }
16
17 template <class T> T bi_search(T l, T r, T end) {
18     if(!check(r - end)) return r - end;
19     for(; r - l > end; ) {
20         T mid = (l + r) / 2;
21         if(check(mid))
22             r = mid;
23         else
24             l = mid;
25     }
26     return l;
27 }
28 /*check gives 000000001111 find the last 0*/
29
30 template <class T> T tri_search(T l, T r, T end) {
31     T midl, midr;
32     for(;;) {
33         midl = (l + r) / 2;
34         midr = (midl + r) / 2;
35         if(midr - midl < end) break;
36         if(get(midr) > get(midl))
37             r = midr;
38         else
39             l = midl;
40     }
41     for(; r - l > end; ) {
42         midl = (l + r) / 2;
43         if(get(r) > get(l))
44             r = midl;
45         else
46             l = midl;
47     }
48     return l;
49 }
50 /*get gives the value, find the minimum*/
51
52 int main() {
53     cin >> n >> x >> y;
54     for(int i = n; i >= 0; i--) {
55         cin >> a[i];
56     }
57     cout << fixed << setprecision(7)
58         << tri_search<double>(x, y, 1e-7);
59 }

```

8.3. 對拍.md

```

1  .vimrc
2
3  set ts=4
4  set autoindent
5  ...
6
7  script
8  ...bash
9  set -e
10 g++ ac.cpp -o ac
11 g++ wa.cpp -o wa
12 for ((i=0;;i++))
13 do
14     echo "$i"
15     python3 gen.py > input
16     ./ac < input > ac.out
17     ./wa < input > wa.out
18     diff ac.out wa.out || break
19 done
20 # factor n 可以質因數分解
21 ...
22
23 python random
24 ...python
25 from random import *
26 n = randint(1, 100)
27 ch = chr(ord('a') + randint(0, 25))
28 choiceSet = sample(s, 4)
29 choiceSet = sample(range(1, n+1), 4)
30 shuffle(arr)
31 ...
32
33 python tree
34 ...python
35 from random import *
36 n = randint(3, 6)
37 print(n)
38 for i in range(2, n+1):
39     print(randint(1, i-1), i)

```

```

41  ...
43  簡單連通圖
44  ...python
45  from random import *
46  n = randint(5, 10)
47  m = randint(n-1, n+3)
49  print(n, m)
51  edge = list()
53  #construct tree
54  for i in range(2, n+1):
55      x = randint(1, i-1)
56      y = i
57      edge.append([min(x, y), max(x, y)])
58      print(x, y)
59  #add extra edge
60  for i in range(m-(n-1)):
61      x = randint(1, n)
62      y = randint(1, n)
63      while x == y or [min(x, y), max(x, y)] in edge:
64          x = randint(1, n)
65          y = randint(1, n)
66      print(x, y)
67      edge.append([min(x, y), max(x, y)])
69  ...
71  c++ debug template
72  ...cpp
73  #ifdef LOCAL // ===== Local =====
74  void dbg() { cerr << '\n'; }
75  template<class T, class ...U> void dbg(T a, U ...b)
76  { cerr << a << ' ', dbg(b...); }
77  template<class T> void org(T l, T r)
78  { while (l != r) cerr << *l++ << ' '; cerr << '\n'; }
79  #define debug(args...) \
80  (dbg("#> (" + string(#args) + ") = (" + args, ")"))
81  #define orange(args...) \
82  (cerr << "#> [" + string(#args) + "] = " + org(args))
83  #else // ===== OnlineJudge =====
84  #pragma GCC optimize("Ofast,unroll-loops,no-stack-protector,fast-math")
85  #pragma GCC target("avx2,bmi,bmi2,lzcnt,popcnt")
86  #define debug(...) ((void)0)
87  #define orange(...) ((void)0)
88  #endif
89  ...

```

9. AnotherVersionDataStructure

9.1. BIT.cpp

```

1  template <class T> class BIT {
2  #define lb(x) ((x) & -(x))
3  #define N (int)2e5 + 5
4  public:
5      T bit[N] = {0};
6      void update(T x, T v) {
7          for(; x < N; x += lb(x)) bit[x] += v;
8      }
9      T qry(T x) {
10         T ans = 0;
11         for(; x; x -= lb(x)) ans += bit[x];
12         return ans;
13     }
14 #undef lb
15 #undef N
16 };
17 /*1based bit update 預設是加值 */

```

9.2. DSU.cpp

```

1  template <class T> class Dsu {
2  #define N 2000005
3  public:
4      T dsu[N], size[N];
5      Dsu(T n) {
6          for(; n; --n) dsu[n] = n, size[n] = 1;
7      }
8      T qry(T x) {
9          if(dsu[x] == x) return x;
10         return dsu[x] = qry(dsu[x]);
11     }
12     void merge(T a, T b) {
13         a = qry(a);
14         b = qry(b);
15         if(a == b) return;
16         if(size[a] < size[b])

```

```

17         dsu[a] = b, size[b] += size[a];
18     }
19     else
20         dsu[b] = a, size[a] += size[b];
21 }
22 #undef N
23 /*1based 初始化為 dsu[x]=x 路徑壓縮 + 啟發式合併 */

```

9.3. Treap.cpp

```

1  // treap 模板 洛谷 P3369 【模板】普通平衡树
2  #include <bits/stdc++.h>
3  using namespace std;
4  #define pnn pair<node *, node *>
5  #define F first
6  #define S second
7  mt19937 mt(hash<string>())("official_beautiful_fruit");
8  struct node {
9      node *l, *r;
10     int val, sz;
11     int mx, mn, sum;
12     int rev_tag, add_tag;
13     node(int x)
14         : val(x), l(0), r(0), sz(1), rev_tag(0), add_tag(0),
15           mx(x), mn(x), sum(x) {}
16     node(node *tr)
17         : val(tr->val), l(tr->l), r(tr->r), sz(tr->sz),
18           rev_tag(tr->rev_tag), add_tag(tr->add_tag),
19           mx(tr->mx), mn(tr->mn) {}
20     void pull() {
21         sz = 1;
22         mx = mn = sum = val;
23         if(l)
24             sz += l->sz, mx = max(mx, l->mx),
25             mn = min(mn, l->mn), sum += l->sum;
26         if(r)
27             sz += r->sz, mx = max(mx, r->mx),
28             mn = min(mn, r->mn), sum += r->sum;
29     }
30     void push() {
31         if(rev_tag) swap(l, r);
32         if(l) l->add_tag += add_tag, l->rev_tag ^= rev_tag;
33         if(r) r->add_tag += add_tag, r->rev_tag ^= rev_tag;
34         mx += add_tag;
35         mn += add_tag;
36         sum += add_tag;
37         add_tag = 0;
38         rev_tag = 0;
39     }
40 };
41 void debug(node *tr) {
42     if(!tr) return;
43     tr->push();
44     tr->pull();
45     debug(tr->l);
46     cout << tr->val << " ";
47     debug(tr->r);
48 }
49 void debug2(node *tr) {
50     if(!tr) return;
51     tr->push();
52     tr->pull();
53     cout << tr->val << " ";
54     debug2(tr->l);
55     debug2(tr->r);
56 }
57 int sz(node *tr) { return tr ? tr->sz : 0; }
58 node *merge(node *a, node *b) {
59     if(!a || !b) return a ? b;
60     a->push();
61     b->push();
62     if(mt() % (sz(a) + sz(b)) < sz(a)) {
63         a->r = merge(a->r, b);
64         a->pull();
65         return a;
66     }
67     b->l = merge(a, b->l);
68     b->pull();
69     return b;
70 }
71 pnn split(node *tr, int v) { //(-inf,v),(v,inf)
72     if(!tr) return {0, 0};
73     tr->push();
74     if(tr->val <= v) {
75         auto [l, r] = split(tr->r, v);
76         tr->r = l;
77         tr->pull();
78         return {tr, r};
79     }
80     auto [l, r] = split(tr->l, v);
81     tr->l = r;
82     tr->pull();
83     return {l, tr};

```

```

}
85 pnn splitsz(node *tr, int k) { //[rk.1,rk.k],[rk.k,rk.n]
    if(!tr || sz(tr) <= k) return {tr, 0};
87     tr->push();
    if(k <= sz(tr->l)) {
89         auto [l, r] = splitsz(tr->l, k);
        tr->l = r;
91         tr->pull();
        return {l, tr};
93     } else if(k <= sz(tr->r) + 1) {
        auto r = tr->r;
95         tr->r = 0;
        tr->pull();
97         return {tr, r};
    } else {
99         auto [l, r] = splitsz(tr->r, k - (sz(tr->l) + 1));
        tr->r = l;
101        tr->pull();
        return {tr, r};
103    }
}
105 node *insert(node *tr, int v) {
    auto [l, r] = split(tr, v);
107     return merge(merge(l, new node(v)), r);
}
109 node *insertkth(node *tr, int k) {
    auto [l, r] = splitsz(tr, k - 1);
111     return merge(merge(l, new node(0)),
        r); // new node 拿來區間操作初始化
113 }
115 node *eraseall(node *tr, int v) {
    auto [l, r] = split(tr, v - 1);
    return merge(l, split(r, v).S);
117 }
119 node *eraseone(node *tr, int v) {
    auto [l, r] = split(tr, v - 1);
    return merge(l, splitsz(r, 1).S);
121 }
123 node *eraskth(node *tr, int k) {
    auto [l, r] = splitsz(tr, k - 1);
    return merge(l, splitsz(r, k).S);
125 }
127 int rnk(node *tr, int v) {
    if(!tr) return 0;
    if(tr->val <= v) return sz(tr->l) + 1 + rnk(tr->r, v);
129     return rnk(tr->l, v);
}
131 int kth(node *tr, int k) {
    auto [l, x] = splitsz(tr, k - 1);
133     auto [m, r] = splitsz(x, 1);
    if(!m) return 0;
135     int ans = m->val;
    tr = merge(merge(l, m), r);
137     return ans;
}
139 int count(node *tr, int L, int R) { // count[L,R]
    auto [l, x] = split(tr, L - 1);
141     auto [m, r] = split(x, R);
    int ans = m->sz; // 看要改啥
143     tr = merge(merge(l, m), r);
    return ans;
145 }
147 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);
149     int ans = m->sum; // 看要改啥
    tr = merge(merge(l, m), r);
    return ans;
151 }
153 int prev(node *tr, int v) {
    auto [x, r] = splitsz(tr, v - 1);
155     auto [l, m] = splitsz(x, sz(x) - 1);
    int ans = m->val;
157     tr = merge(merge(l, m), r);
    return ans;
159 }
161 int next(node *tr, int v) {
    auto [l, x] = split(tr, v);
    auto [m, r] = splitsz(x, 1);
163     int ans = m->val;
    tr = merge(merge(l, m), r);
165     return ans;
}
167 int qry(node *tr, int L, int R) { // qry[L,R]
    auto [x, r] = splitsz(tr, R);
169     auto [l, m] = splitsz(x, L - 1);
    int ans = m->sum; // 看要改啥
171     tr = merge(merge(l, m), r);
    return ans;
173 }
175 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);

```

```

177     m->val += v;
    m->add_tag += v;
179     m->rev_tag = 1; // 看要改啥
    tr = merge(merge(l, m), r);
181 }
183 int main() {
    int t;
    node *tr = 0;
    for(cin >> t; t--;) {
185         int op, x;
        cin >> op >> x;
        switch(op) {
187             case 1:
                tr = insert(tr, x);
                break;
191             case 2:
                tr = eraseone(tr, x);
                break;
193             case 3:
                cout << rnk(tr, x - 1) + 1 << "\n";
                break;
195             case 4:
                cout << kth(tr, x) << "\n";
                break;
197             case 5:
                cout << prev(tr, x) << "\n";
                break;
201             case 6:
                cout << next(tr, x) << "\n";
                break;
203         }
    }
205 }
207 }
209 }

```

9.4. Treap 但可以多個數縮點 (疑似爛的).cpp

```

1 // treap 模板 洛谷 P3369 【模板】普通平衡树
#include <bits/stdc++.h>
3 using namespace std;
#define pnn pair<node *, node *>
5 #define F first
#define S second
7 #define int long long
mt19937 mt(hash<string>{}("official_beautiful_fruit"));
9 struct node {
    node *l, *r;
11     int val, sz;
    int mx, mn, sum, num;
13     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) {}
15     node(node *tr)
        : val(tr->val), l(tr->l), r(tr->r), sz(tr->sz) {}
17     void pull() {
        sz = 1;
        mx = mn = sum = num;
        if(l)
21             sz += l->sz, mx = max(mx, l->mx),
                mn = min(mn, l->mn), sum += l->sum;
        if(r)
23             sz += r->sz, mx = max(mx, r->mx),
                mn = min(mn, r->mn), sum += r->sum;
25     }
    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;
27     }
};
29 void debug(node *tr) {
    if(!tr) return;
    debug(tr->l);
    cout << tr->val << " ";
    debug(tr->r);
31 }
33 void debug2(node *tr) {
    if(!tr) return;
    cout << tr->val << " ";
    debug2(tr->l);
    debug2(tr->r);
35 }
37 int sz(node *tr) { return tr ? tr->sz : 0; }
39 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();
41     }
    else {
        b->l = merge(b->l, a);
        b->pull();
43     }
    return b;
45 }
47 int main() {
    int n;
    cin >> n;
    node *tr = 0;
    while(n--) {
        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;
        }
    }
    return 0;
}

```

```

    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 *eraskth(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--;) {
        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

```

1 // luogu P4097 區間插線段李超
2 #include <bits/stdc++.h>
3 using namespace std;
4 #define N 50005
5 struct Line {
6     double a, b;
7     int l, r, id; // ax+b{l<=x<=r}
8     Line(double _a = -1e6, double _b = -1, int _l = 1,
9         int _r = N, int _id = 0)
10         : a(_a), b(_b), l(_l), r(_r), id(_id) {}
11     double operator()(int x) { return a * x + b; }
12 } line[N];
13 int seg[N << 2];
14 #define lid (id << 1)
15 #define rid (id << 1 | 1)
16 #define M (L + R >> 1)
17 #define eps 1e-6
18 void ins(int l, int L = 1, int R = N, int id = 1) {
19     // cout<<"ins{"<<line[l].a<<","<<line[l].b<<","<<line[l].l<<","<<line[l].r<<"}\n";
20     // "<<R<<"\n";
21     if(line[l].r < L || R < line[l].l) return;
22     if(L == R) {
23         if(line[l].M - line[seg[id]].M > eps) seg[id] = l;
24         return;
25     }
26     if(line[l].l <= M && M <= line[l].r &&
27         line[l].M - line[seg[id]].M > eps)
28         swap(l, seg[id]);
29     if(line[l].l <= L && R <= line[l].r) {
30         if(line[l].a - line[seg[id]].a > eps)

```



```

31     ins(l, M + 1, R, rid);
32     else
33         ins(l, L, M, lid);
34     }
35     /*if(line[l].a>line[seg[id]].a)*/ ins(l, M + 1, R, rid);
36     /*else */ ins(l, L, M, lid);
37 }
38 int qry(int x, int L = 1, int R = N, int id = 1) {
39     // cout<<"qry"<<x<<"{"<<line[seg[id]].a<<"{"<<line[seg[id]].l<<"{"<<R<<"{"<<id<<"\n";
40     // "<<R<<" "<<id<<"\n";
41     if(L == R) return seg[id];
42     int k = (x <= M ? qry(x, L, M, lid)
43         : qry(x, M + 1, R, rid)),
44         not_k = 0, not_seg = 0;
45     if(line[k].r < x || x < line[k].l) not_k = 1;
46     if(line[seg[id]].r < x || x < line[seg[id]].l) not_seg = 1;
47     if(not_k && not_seg) return 0;
48     if(not_k) return seg[id];
49     if(not_seg) return k;
50     return line[k](x) - line[seg[id]](x) > eps ? k : seg[id];
51 }
52 int main() {
53     int n, ans = 0, p = 1;
54     for(cin >> n; n--;) {
55         int op;
56         cin >> op;
57         if(op) {
58             int x0, y0, x1, y1;
59             cin >> x0 >> y0 >> x1 >> y1;
60             x0 = (x0 + ans - 1) % 39989 + 1;
61             y0 = (y0 + ans - 1) % 1000000000 + 1;
62             x1 = (x1 + ans - 1) % 39989 + 1;
63             y1 = (y1 + ans - 1) % 1000000000 + 1;
64             if(x0 > x1) swap(x0, x1), swap(y0, y1);
65             // cout<<"{"<<((double)y1-y0)/(x1-x0)<<"{"<<
66             // "<<y0-x0*((double)y1-y0)/(x1-x0)<<"\n";
67             if(x0 != x1)
68                 line[p] = Line(((double)y1 - y0) / (x1 - x0),
69                     y0 - x0 * ((double)y1 - y0) / (x1 - x0),
70                     x0, x1, p);
71             else
72                 line[p] = Line(0, max(y0, y1), x0, x1, p);
73             ins(p);
74             ++p;
75         } else {
76             int k;
77             cin >> k;
78             k = (k + ans - 1) % 39989 + 1;
79             cout << (ans = qry(k)) << "\n";
80         }
81     }
82     // cout<<qry(9)<<"\n";
83 }

```

9.6. 單點修改動態開點線段樹.cpp

```

1 #include <bits/stdc++.h>
2 using namespace std;
3 #define N 200005
4 #define M int m = l + r >> 1
5 #define MAX 1000000000
6 int a[N];
7 typedef struct node {
8     struct node *l, *r;
9     int val;
10 };
11 void check(node *tree, int flag) {
12     if(flag && !tree->r)
13         tree->r = (node *)malloc(sizeof(struct node)),
14         tree->r->val = 0;
15     else if(!flag && !tree->l)
16         tree->l = (node *)malloc(sizeof(struct node)),
17         tree->l->val = 0;
18 }
19 void upd(int pos, int val, int l, int r, node *tree) {
20     tree->val += val;
21     if(l == r) return;
22     M;
23     if(pos > m)
24         check(tree, 1), upd(pos, val, m + 1, r, tree->r);
25     else
26         check(tree, 0), upd(pos, val, l, m, tree->l);
27 }
28 int qry(int a, int b, int l, int r, node *tree) {
29     if(!tree) return 0;
30     if(a <= l && r <= b) return tree->val;
31     M;
32     if(a > m) return qry(a, b, m + 1, r, tree->r);
33     if(b <= m) return qry(a, b, l, m, tree->l);
34     return qry(a, b, m + 1, r, tree->r) +
35         qry(a, b, l, m, tree->l);
36 }

```

```

37 int main() {
38     int n, q, i = 1, x;
39     node *root = (node *)malloc(sizeof(struct node));
40     root->val = 0;
41     for(scanf("%d %d", &n, &q); i <= n; ++i)
42         getc(), scanf("%d", &a + i),
43         upd(a[i], 1, 1, MAX, root);
44     // printf("%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, n, 1));
45     for(; q--;) {
46         getc();
47         char c = getc();
48         scanf("%d %d", &x, &i);
49         if(c == '!')
50             upd(a[x], -1, 1, MAX, root),
51             a[x] = i, upd(i, 1, 1, MAX, root);
52     }
53     else
54         printf("%d\n", qry(x, i, 1, MAX, root));
55 }

```

9.7. 單點修改無懶標線段樹.cpp

```

1 template <class T> class Seg {
2     #define lid id << 1
3     #define rid id << 1 | 1
4     #define M (L + R >> 1)
5     #define N 200005
6     public:
7         T a[N], seg[N << 2];
8         Seg() {
9             for(int i = 1; i <= n; ++i) cin >> a[i];
10            init();
11        }
12        T update(int pos, int val, int L = 1, int R = n,
13            int id = 1) {
14            if(L == R) return seg[id] = val;
15            if(pos > M)
16                return seg[id] = seg[lid] +
17                    update(pos, val, M + 1, R, rid);
18            return seg[id] = update(pos, val, L, M, lid) + seg[rid];
19        }
20        T qry(int l, int r, int L = 1, int R = n, int id = 1) {
21            if(l <= L && R <= r) return seg[id];
22            if(L == R) return seg[id];
23            int M = L + R >> 1;
24            if(l > M) return qry(l, r, M + 1, R, rid);
25            if(r <= M) return qry(l, r, L, M, lid);
26            return qry(l, M, L, M, lid) +
27                qry(M + 1, r, M + 1, R, rid);
28        }
29        private:
30            T init(int l = 1, int r = n, int id = 1) {
31                if(l == r) return seg[id] = a[l];
32                int m = l + r >> 1;
33                return seg[id] = init(l, m, lid) + init(m + 1, r, rid);
34            }
35        }
36        #undef lid
37        #undef rid
38        #undef N
39    };
40    /*1based 陣列 1based id 單點修改 預設維護區間和 */

```

9.8. 懶標線段樹.cpp

```

1 struct Seg {
2     #define lid (id << 1)
3     #define rid ((id << 1) | 1)
4     #define M (L + R >> 1)
5     #define N 200005
6     LL seg[N << 2], tag[N << 2];
7     void inline addtag(int id, LL v, int L, int R) {
8         seg[id] += v * (R - L + 1);
9         tag[id] += v;
10    }
11    void inline push(int id, int L, int R) {
12        addtag(lid, tag[id], L, M);
13        addtag(rid, tag[id], M + 1, R);
14        tag[id] = 0;
15    }
16    void inline pull(int id) { seg[id] = seg[lid] + seg[rid]; }
17    void init(int L = 1, int R = n, int id = 1) {
18        if(L == R) {
19            seg[id] = 0;
20            tag[id] = 0;
21            return;
22        }
23        init(L, M, lid);
24        init(M + 1, R, rid);
25        pull(id);
26    }

```

```

27 void upd(int l, int r, LL v, int L = 1, int R = n,
        LL id = 1) {
29     if(l <= L && R <= r) {
        addtag(id, v, L, R);
        return;
31     }
        push(id, L, R);
        if(r <= M)
33             upd(l, r, v, L, M, lid);
        else if(M + 1 <= l)
35             upd(l, r, v, M + 1, R, rid);
        else
37             upd(l, M, v, L, M, lid),
                upd(M + 1, r, v, M + 1, R, rid);
41     pull(id);
}
43 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) +
47         qry(M + 1, r, M + 1, R, rid);
49 }
51 } 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

```

1 // luogu P4254 李超
#include <bits/stdc++.h>
3 using namespace std;
#define N 50005
5 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];
11 #define lid (id << 1)
#define rid (id << 1 | 1)
#define M (L + R >> 1)
13 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)
21         ins(l, M + 1, R, rid);
    else
23         ins(l, L, M, lid);
}
25 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));
29 }
31 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)
39             << "\n";
        } else {
            double s, p;
            cin >> s >> p;
            ins(Line(p, s));
41         }
    }
43 }
45 }

```

10. AnotherVersionMath

10.1. CRT(luoguVersion).cpp

```

1 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;
11 }

```

10.2. PollardRho.cpp

```

1 #include <bits/stdc++.h>
using namespace std;
3 #define LL long long
#define uLL __uint128_t
5 #define sub(a, b) ((a) < (b) ? (b) - (a) : (a) - (b))
template <class T, class POW>
7 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;
    }
}
15 }
17 /* 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)*/
19 uLL pri[7] = {2, 325, 9375, 28178,
                450775, 9780504, 1795265022}; /*2^64*/
// int p[3]={2,7,61};/*2^32*/
23 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;
}
37 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;
}
45 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;
}
53 /*gcd(a,b) 默認 gcd(a,0)=a*/
55 mt19937 rnd(time(0));
template <class T> T f(T x, T c, T mod) {
    return (((uLL)x) * x % mod + c) % mod;
}
57 }
59 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);
        }
    }
}
69 }
71 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));
}
77 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);
    }
87 }

```

10.3. 快速幂.cpp

```

1 template <class T, class POW>

```

```

1 void fastpow(T x, POW n, POW p, T &ans) {
2     for(; n >= 1) {
3         if(n & 1) {
4             ans *= x;
5             ans %= p;
6         }
7         x *= x;
8         x %= p;
9     }
10 }
11 /* 輸入 x,n,p,ans 會將 ans 修改為 x^n%p
12 對整數/矩陣/不要求精度的浮點 皆有效
13 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

```

1 template <class T> T extgcd(T a, T b, T &x, T &y) {
2     if(!b) {
3         x = 1;
4         y = 0;
5         return a;
6     }
7     T ans = extgcd(b, a % b, y, x);
8     y -= a / b * x;
9     return ans;
10 }
11 /*extgcd(a,b,x,y)=ax+by, x 跟 y 是會被修改的參數 */
12 template <class T> T modeq(T a, T b, T p) {
13     T x, y, d = extgcd(a, p, x, y);
14     if(b % d) return 0;
15     return ((b / d * x) % p + p) % p;
16 }
17 /*x=modeq(a,b,n), ax=b(mod n), 0<=x<n
18 modeq(a,1,n) 相當於求 a 在 mod n 下的逆元 */
19 template <class T> T gcd(T a, T b) {
20     if(!a) return b;
21     if(!b) return a;
22     if(a & b & 1) return gcd(abs(a - b), min(a, b));
23     if(a & b & 1) return gcd(a, b >> 1);
24     if(b & 1) return gcd(a >> 1, b);
25     return gcd(a >> 1, b >> 1) << 1;
26 }
27 /*gcd(a,b) 默認 gcd(a,0)=a*/
28 ll crt(V<ll> &p, V<ll> &a) {
29     ll n = 1, ans = 0, k = a.size();
30     for(ll &e : p) n *= e;
31     for(int i = 0; i < k; ++i)
32         ans = (ans + a[i] * n / p[i] % n *
33             modeq(n / p[i], 1LL, p[i]) % n) %
34             n;
35     return (ans % n + n) % n;
36 }
37 /*(a+b)^p ≡ a+b ≡ a^p + b^p (mod p) (小費馬)
38 (p-1)! ≡ -1 (mod p) (威爾遜定理)
39 v(n) := n中p的幕次, (n)_p := n/p^{v(n)},
40 s(n) := p進制下n的所有位數和
41 v(n!) = ∑_{i=1}^n ⌊ n/p^i ⌋ (勒壤得定理)
42 n/p^i = (n-s(n))/p-1 (勒壤得定理)
43 v((n)_p) = s(n) + s((n-n)/p) - s(n) (庫默爾定理)
44 v((m1,m2,...mk)_p) = ∑_{i=1}^k s(mi) - s(n) (庫默爾定理推廣)
45 \[
46     (n!)_p \equiv -1^{\lfloor \frac{n}{p} \rfloor} \pmod p
47 \]
48 \[
49     ((\lfloor \frac{n}{p} \rfloor!)_p \pmod p) \equiv -1^{\lfloor \frac{n}{p^2} \rfloor} \pmod p
50 \]
51 打階乘表 + 迭代這條式子可以 O(p + log_p(n)) (mod p 下階乘)
52 \[
53     \binom{n}{m} \equiv \frac{((n+m)!)_p}{(n!)_p (m!)_p} \pmod p
54 \]
55 把 p 從 C(n, m) 裡面隔離掉了 就能用上面的
56 (n!)_p + 模逆元 (mod p 下階乘推廣至二項式)
57 ((p^q)!)_p ≡ ±1 (mod p)^q (威爾遜定理推廣)
58 \[
59     \binom{n}{m} \equiv \binom{n}{m} \pmod p
60 \]
61 \[
62     \binom{n}{m} \equiv \binom{n}{m} \pmod p
63 \]
64 (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))
65 若 p 進制下任何一位 i 滿足 n_i < m_i 則
66 \[
67     \binom{n}{m} \equiv 0 \pmod p
68 \]
69 則因 \[
70     \binom{n}{m} = \prod_{i=0}^{\max(\log_p(a), \log_p(b))} \binom{n_i}{m_i} \pmod p
71 \] 導致 \[
72     \binom{n}{m} \equiv 0 \pmod p
73 \]
74 設 p = 2 則有 \[
75     \binom{n}{m} \] 是奇數的充要條件為二進制下每一位
76 n < m (Lucas 定理額外性質) Lucas 定理可由此生成函數做法得到
77 不依賴小費馬 對多項式也成立 根據上述
78 \[
79     \binom{n}{m} \equiv k \pmod p
80 \] 可將 k 做唯一質數分解
81 個別做完再做 crt 得到結果 (exlucas 定理)
82 \[
83     C(0) = C(1) = 1, n > 1 \text{ 時 } C(n) = \sum_{k=0}^{n-1} C(k)C(n-1-k) =
84 \]

```

```

75 \[
76     \frac{(2n)!}{n!} \]
77 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
78 C(n)k^n
79 模逆元表 p=i*(p/i)+p%i, -p%i=i*(p/i), inv(i)=-(p/i)*inv(p%i)*/
80 LL fracp[N], invp[N];
81 void fracp_init(LL p) {
82     fracp[0] = 1;
83     for(int i = 1; i < p; ++i) fracp[i] = fracp[i - 1] * i % p;
84 }
85 void invp_init(LL p) {
86     invp[0] = invp[1] = 1;
87     for(int i = 2; i < p; ++i)
88         invp[i] = p - (p / i * invp[p % i]) % p;
89 }
90 /* 階乘表跟模逆元表 之後可以考慮改一下長相 */
91 template <class T> T lucas(T n, T m, T p) {
92     if(!m) return 1;
93     if(m > n || m % p > n % p) return 0;
94     return lucas(n / p, m / p, p) * fracp[n % p] % p *
95         invp[fracp[n % p - m % p]] % p * invp[fracp[m % p]] % p;
96 }
97 /*lucas(n,m,p)=C(n,m)%p 要求要帶階乘表跟模逆元表
98 * 0(p+log_p(n))*/
99 /* 米勒拉賓質數 2,325,9375,28178,450775,9780504,1795265022*/
100 /*crt 質數
101 (2^16)+1 65537 3
102 7*17*(2^23)+1 998244353 3
103 1255*(2^20)+1 1315962881 3
104 51*(2^25)+1 1711276033 29
105 */

```

10.5. 篩法.cpp

```

1 // 待加入分塊篩
2 template <class T> class Prime {
3     #define N (int)1e8 + 9
4     public:
5         vector<T> list, factor;
6         Prime(T n) {
7             eular(n);
8             // eratosthenes(n);
9             // sqrt_sieve
10            // factorize(n);
11        }
12        void show() {
13            for(T e : list) printf("%lld ", e);
14            putchar('\n');
15        }
16        private:
17            bitset<N> notprime; // 1e8<2^27=128MB
18            void eular(T n) {
19                for(T i = 2; i <= n; ++i) {
20                    if(!notprime[i]) list.emplace_back(i);
21                    const T k = n / i;
22                    for(T j : list) {
23                        if(j > k) break;
24                        notprime[i * j] = 1;
25                        if(!(i % j)) break;
26                    }
27                }
28            }
29            void eratosthenes(T n) {
30                for(T i = 2; i <= n; ++i) {
31                    if(!notprime[i]) list.emplace_back(i);
32                    const T k = n / i;
33                    for(T j : list) {
34                        if(j > k) break;
35                        notprime[i * j] = 1;
36                        if(!(i % j)) break;
37                    }
38                }
39            }
40            void sqrt_sieve(T n) {
41                for(T i = 2; i <= n; ++i) {
42                    bool isprime = 1;
43                    for(T j : list) {
44                        if(j > i / j) break;
45                        if(!(i % j)) {
46                            isprime = 0;
47                            break;
48                        }
49                    }
50                    if(isprime) list.emplace_back(i);
51                }
52            }
53            void factorize(T n) {
54                factor = vector<T>(n);
55                if(list.empty()) eular(n);
56                for(T j : list) factor[j] = j;
57                for(T i = 2; i <= n; ++i) {
58                    const T k = n / i;
59                }

```

```

61         for(T j : list) {
62             if(j > k) break;
63             factor[i * j] = j;
64             if(!(i % j)) break;
65         }
66     }
67 #undef N
68 };
69 /*Prime prime(n) 建立打好 1~n 質數表的物件
70 prime.list(一個 vector) 是質數表
71 可修改 define N 決定歐篩/埃篩上限
72 可在建構子選擇篩法 有歐篩/埃篩/根號暴力搜
73 prime.factorize(n) 用歐篩方式得到 1~n 所有數的最小質因數
74 可在 factor(一個 vector) 上一路回溯 logn 得到一個數的質因數分解
75 做 n 個數質因數分解共花 nlogn
76 show() 會以空格隔開 顯示所有 list 內的元素 有尾空格尾換行
77 printf 裡面用%lld 視情況換為%d 或 cout*/

```

11. AnotherVersionString

11.1. KMP (2).cpp

```

1 #define V vector
2 V<int> kmp(string s) {
3     int n = s.size();
4     V<int> f(n);
5     for(int i = 1; i < n; ++i) {
6         int j = f[i - 1];
7         for(; j > 0 && s[j] != s[i];) j = f[j - 1];
8         f[i] = j + (s[j] == s[i]);
9     }
10    return f;
11 }
12 // kmp(s+"#"+t) 得到的陣列中，f[i]=s.size() 的格子代表 t
13 // 中匹配到 s 的結尾位置

```

11.2. KMP.cpp

```

1 class Kmp {
2 #define N 1000005
3 public:
4     int fail[N], p[N];
5     Kmp(char *t, int n) {
6         fail[0] = -1;
7         for(int i = 1; i < n; ++i) {
8             for(fail[i] = fail[i - 1];
9                 t[i] != t[fail[i] + 1] && fail[i] != -1;)
10                fail[i] = fail[fail[i]];
11            if(t[i] == t[fail[i] + 1]) ++fail[i];
12        }
13    }
14    void match(char *s, int n, char *t, int m) {
15        p[0] = (s[0] == t[0]) - 1;
16        for(int i = 1; i < n; ++i) {
17            for(p[i] = p[i - 1];
18                s[i] != t[p[i] + 1] && p[i] != -1;)
19                p[i] = fail[p[i]];
20            if(s[i] == t[p[i] + 1]) ++p[i];
21        }
22    }
23 #undef N
24 };
25 /*Kmp kmp(t) 會建好 t 的失配函數 fail[]
26 * match 會把每格匹配完的失配函數 p[] 建好 */

```

11.3. Manacher (2).cpp

```

1 #define T(x) ((x)&1 ? s[(x) >> 1] : '.')
2 int ex(string &s, int l, int r, int n) {
3     int i = 0;
4     while(l - i >= 0 && r + i < n && T(l - i) == T(r + i)) ++i;
5     return i;
6 }
7 int manacher(string s, int n) {
8     n = 2 * n + 1;
9     int mx = 0;
10    int center = 0;
11    vector<int> r(n);
12    int ans = 1;
13    r[0] = 1;
14    for(int i = 1; i < n; i++) {
15        int ii = center - (i - center);
16        int len = mx - i + 1;
17        if(i > mx) {
18            r[i] = ex(s, i, i, n);
19            center = i;
20            mx = i + r[i] - 1;
21        } else if(r[ii] == len) {
22            r[i] = len + ex(s, i - len, i + len, n);
23            center = i;

```

```

24            mx = i + r[i] - 1;
25        } else {
26            r[i] = min(r[ii], len);
27        }
28        ans = max(ans, r[i]);
29    }
30    return ans - 1;
31 }

```

11.4. Manacher.cpp

```

1 #define V vector
2 string manacher(string t) {
3     int n = t.size() << 1 | 1;
4     string s(n, '#');
5     for(int i = 0, m = t.size(); i < m; ++i)
6         s[i << 1 | 1] = t[i];
7     V<int> p(n);
8     for(int i = 0, m = 0, r = 0; i < n; ++i) {
9         p[i] = r > i ? min(r - i, p[m - (i - m)]) : 1;
10        for(; i - p[i] >= 0 && i + p[i] < n &&
11            s[i - p[i]] == s[i + p[i]];)
12            ++p[i];
13        if(i + p[i] > r) r = i + p[i], m = i;
14    }
15    int k = 0;
16    string ans = "";
17    for(int i = 0; i < n; ++i)
18        if(p[i] > p[k]) k = i;
19    for(int r = k + p[k], l = k - p[k]; ++l < r;)
20        if(s[l] != '#') ans += s[l];
21    return ans;
22 }
23 // manacher(s) 給出 s
24 // 中的最長回文，若有多個則給字典序最小的，p[i] = 以 i
25 // 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

```

1 class Z {
2 public:
3     vector<int> z;
4     Z(string s) {
5         z = vector<int>(s.size());
6         for(int l = 0, i = 1; i < n; ++i) {
7             if(l + z[l] >= i)
8                 z[i] = min(z[l] + l - i, z[i - l]);
9             while(i + z[i] < n && s[z[i]] == s[i + z[i]])
10                ++z[i];
11            if(i + z[i] > l + z[l]) l = i;
12        }
13    }
14 };
15 // Z(s+"#"+t) 得到的陣列中，f[i]=s.size() 的格子代表 t
16 // 中匹配到 s 的開頭位置

```

12. AnotherVersionGraph

12.1. Dijkstra.cpp

```

1 // cses Shortest Routes I
2 #include <bits/stdc++.h>
3 using namespace std;
4 #define N 100005
5 #define LL long long
6 #define pii pair<int, int>
7 #define pil pair<LL, LL>
8 #define F first
9 #define S second
10 #define pb push_back
11 #define DE if(1)
12 #define INF (LL)1e16
13 vector<pil> adj[N];
14 LL d[N];
15 bitset<N> vis;
16 int main() {
17     int n, m, u, v;
18     LL c;
19     priority_queue<pil, vector<pil>, greater<pil>> q;
20     for(cin >> n >> m; m--;)
21         cin >> u >> v >> c, adj[u].pb({v, c});
22     q.push({0, 1});
23     d[1] = 0;
24     for(u = 2; u <= n; ++u) d[u] = INF;
25     for(; !q.empty(); q.pop()) {
26         if(vis[q.top().S]) continue;
27         vis[q.top().S] = 1;
28         for(auto &e : adj[q.top().S]) {
29             if(!vis[e.F] && q.top().F + e.S < d[e.F]) {
30                 d[e.F] = q.top().F + e.S;
31                 q.push({d[e.F], e.F});

```

```

    }
}
for(u = 1; u <= n; ++u) printf("%lld ", d[u]);
}

```

12.2. SCC.cpp

```

#include <bits/stdc++.h>
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

```

// cses Planets Queries II 基環樹森林模板
#include <bits/stdc++.h>
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

```

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

```

#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];
}

```



```

13 int jmp(int x, int k) {
    for(int i = LG; i--;)
        for(; k >= 1 << i; k -= 1 << i) x = par[x][i];
    return x;
}
17 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];
}
25 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);
}
37 // 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
39 // 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
41 // 步的祖先是誰

43 #pragma GCC optimize(
    "Ofast,fast-math,unroll-loops,no-stack-protector")
45 #include <bits/stdc++.h>
47 using namespace std;
49 #define ll long long
51 #define pb push_back
53 #define N 200005
55 #define pii pair<int, int>
57 #define V vector
59 #define inf 1000000000
61 #define M 200005
63 #define LG 18
65 #define pii pair<int, int>
67 #define ppp pair<pii, pii>
69 char buf[1 << 22], *p1, *p2;
71 int p[12];
73 #define gc()
75 (p1 == p2 &&
    (p2 = (p1 = buf) + fread(buf, 1, 1 << 22, stdin),
    p1 == p2)
    ? EOF
    : *p1++)
77 inline int gi() {
    int x = 0;
    for(char c; '0' <= (c = gc()) && c <= '9'; x += c - '0')
        x *= 10;
    return x;
}
79 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);
}
81 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;

```

```

105 auto &u, &v = p2;
107 if(qry(qry, u) != qry(qry, v))
    upd(u, v), adj[u].pb({v, w}), adj[v].pb({u, w});
}
109 vector<vector<int>> par(n + 1, vector<int>(LG)),
    mx(n + 1, vector<int>(LG));
111 vector<int> dep(n + 1);
113 auto dfs = [&par, &mx, &dep, &adj](auto dfs, int now,
    int p = 0,
    int w = 0) -> void {
115     par[now][0] = p;
117     mx[now][0] = w;
119     dep[now] = dep[p] + 1;
121     for(auto &[e, w] : adj[now])
        if(e != p) dfs(dfs, e, now, w);
    };
123 dfs(dfs, 1);
125 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]);
127 auto lca = [&par, &dep](int u, int v) -> int {
129     if(dep[u] > dep[v]) swap(u, v);
131     for(int i = LG; i--;)
        if((1 << i) & (dep[v] - dep[u])) v = par[v][i];
133     if(u == v) return u;
135     for(int i = LG; i--;)
        if(par[u][i] != par[v][i])
            u = par[u][i], v = par[v][i];
    return par[u][0];
}
137 auto path = [&par, &mx, &dep](int k, int x) -> int {
139     int ans = 0;
141     for(int i = LG; i--;)
        if((1 << i) & (dep[x] - dep[k]))
            ans = max(ans, mx[x][i]), x = par[x][i];
    return ans;
}
143 for(auto &p1, &p2 : g) {
145     auto &w, &idx = p1;
147     auto &u, &v = p2;
149     int k = lca(u, v);
    ans[idx] = max(path(k, u), path(k, v)) >= w;
}
151 for(int i = 0; i < m; ++i)
    cout << i << " "
    << (const char[2][5]){ "NO\n", "YES\n" }[ans[i]];
153 cout << "\n";
155 for(int k; q--;) {
157     cin >> k;
159     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];
}
163 }

```