

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		
1.5 PbdsRope.cpp	2		
1.6 PbdsTree.cpp	2		
1.7 PersistentSegmentTree.cpp	3	9 AnotherVersionDataStructure	16
1.8 Treap.cpp	3	9.1 BIT.cpp	16
2 Math	3	9.2 DSU.cpp	16
2.1 CRT.cpp	3	9.3 Treap.cpp	16
2.2 CountPrimes.cpp	4	9.4 Treap 可以多個數縮點 (疑似爛的).cpp	17
2.3 FFT.cpp	4	9.5 區間插線段單點查詢李超 (是爛的).cpp	18
2.4 FWT.cpp	5	9.6 單點修改動態開點線段樹.cpp	19
2.5 Formula.tex	5	9.7 單點修改無懶標線段樹.cpp	19
2.5.1 Dirichlet Convolution	5	9.8 懶標線段樹.cpp	19
2.5.2 Burnside's Lemma	5	9.9 純直線單點查詢李超.cpp	20
2.5.3 Pick Theorem	5		
2.5.4 Fermat's Little Theorem	5		
2.5.5 Wilson's Theorem	5		
2.5.6 Legendre Theorem	5		
2.5.7 Kummer Theorem	5	10 AnotherVersionMath	20
2.5.8 ext-Kummer Theorem	5	10.1 CRT(luoguVersion).cpp	20
2.5.9 Factorial with mod	5	10.2 PollardRho.cpp	20
2.5.10 Properties of nCr with mod	5	10.3 快速冪.cpp	20
2.5.11 ext-Lucas' Theorem	5	10.4 數論.cpp	21
2.5.12 Catalan Number	5	10.5 篩法.cpp	21
2.5.13 modinv table	5		
2.5.14 LTE	5		
2.6 Gaussian-Jordan.cpp	5	11 AnotherVersionString	22
2.7 Generator.cpp	5	11.1 KMP (2).cpp	22
2.8 Inv.cpp	6	11.2 KMP.cpp	22
2.9 Lucas.cpp	6	11.3 Manacher (2).cpp	22
2.10 MillerRabin.cpp	6	11.4 Manacher.cpp	22
2.11 Mu.cpp	6	11.5 Z.cpp	22
2.12 NTT.cpp	7		
2.13 PollardRho.cpp	7	12 AnotherVersionGraph	22
2.14 XorBasis.cpp	7	12.1 Dijkstra.cpp	22
2.15 mtt.cpp	8	12.2 SCC.cpp	23
		12.3 cses 有向圖基環樹森林.cpp	23
3 String	8		
3.1 Booth.cpp	8	13 AnotherVersionGeometry	23
3.2 KMP.cpp	8	13.1 DynamicHull.cpp	23
3.3 LongestPalindrome.cpp	8		
3.4 Z.cpp	8	14 AnotherVersionTree	23
		14.1 LCA.cpp	23
4 Graph	8		
4.1 2-SAT(CSES Planets Cycles).cpp	8	1. DataStructure	
4.2 Dijkstra.cpp	8		
4.3 Dinic.cpp	8	1.1 2DBIT.cpp	
4.4 MaximumFlow.cpp	8		
4.5 SCC.cpp	9	1 // cses Forest Queries II	
4.6 VBCC.cpp	9	#include <bits/stdc++.h>	
4.7 one-degree-cycle(CSES Planets Cycles).cpp	9	using namespace std;	
		#define LL long long	
5 DP	9	#define pii pair<int, int>	
5.1 CHO.cpp	10	#define N 1005	
5.2 Li-Chao-SegmentTree.cpp	10	#define F first	
5.3 SOSDP.cpp	10	#define S second	
		int bit[N][N];	
		#define lb(x) (x & -x)	
		void upd(int i, int j, int v) {	
		for(; j < N; j += lb(j))	
		for(int k = i; k < N; k += lb(k)) bit[k][j] += v;	
		}	
		int qry2(int i, int j) {	
		int ans = 0;	
		for(; j; j -= lb(j))	
		for(int k = i; k; k -= lb(k)) ans += bit[k][j];	
		return ans;	
		}	
		int qry(int y1, int x1, int y2, int x2) {	
		return qry2(y2, x2) - qry2(y2, x1 - 1) - qry2(y1 - 1, x2) +	
		qry2(y1 - 1, x1 - 1);	
		}	
		int main() {	
		int n, q, i = 1, j, y, x;	
		for(scanf("%d %d", &n, &q); getchar(), i <= n; ++i)	
		for(j = 1; j <= n; ++j)	
		if(getchar() == '*') upd(i, j, 1);	
		for(; q--;) {	
		scanf("%d", &i);	
		if(i == 1)	
		scanf("%d%d", &i, &j),	
		upd(i, j, 1 - 2 * qry(i, j, i, j));	
		else	

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

1.2. DynamicSegmentTree.cpp

```

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     }
88 }
89

```

```
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 }
```

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)) * 0xbfb58476d1ce4e5b9;
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(iterator) modify(iterator,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())
28     //out of this will get tr.end()
29 tr.order_of_key(val); //return rank(val);
30 tr.join(tr2); //merge tr
31 and tr2, tr2.clear() tr.split(const int&r,RBTree&tr2); //<r
32 will in tr, >=r will in tr2
33 */

```

1.7. PersistentSegmentTree.cpp

```

1 // cses Range Queries and Copies
2 #include <bits/stdc++.h>
3 using namespace std;
4 #define LL long long
5 #define pii pair<int, int>
6 #define N 200005
7 #define F first
8 #define S second
9 int n, ver = 1;
10 LL a[N];
11 struct Seg {
12     LL v = 0;
13     struct Seg *l = NULL, *r = NULL;
14 #define M (L + R >> 1)
15     static const void init(Seg *node, int L = 1, int R = n) {
16         if(L == R) {
17             node->v = a[L];
18             return;
19         }
20         node->l = new Seg();
21         init(node->l, L, M);
22         node->r = new Seg();
23         init(node->r, M + 1, R);
24         node->v = node->l->v + node->r->v;
25     }
26     static const void upd(Seg *node, int x, LL v, int L = 1,
27                           int R = n) {
28         if(L == R) {
29             node->v = v;
30             return;
31         }
32         if(x <= M)
33             node->l = new Seg(*node->l),
34             upd(node->l, x, v, L, M);
35         else
36             node->r = new Seg(*node->r),
37             upd(node->r, x, v, M + 1, R);
38         node->v = node->l->v + node->r->v;
39     }
40     static const LL qry(Seg *node, int l, int r, int L = 1,
41                      int R = n) {
42         if(l <= L && R <= r) return node->v;
43         if(r <= M) return qry(node->l, l, r, L, M);
44         if(M + 1 <= l) return qry(node->r, l, r, M + 1, R);
45         return qry(node->l, l, M, L, M) +
46                qry(node->r, M + 1, r, M + 1, R);
47     }
48 } * tree[N];
49 int main() {
50     ios::sync_with_stdio(0);
51     cin.tie(0);
52     cout.tie(0);
53     int q, i = 1, j, k;
54     for(cin >> n >> q; i <= n; ++i) cin >> a[i];
55     tree[1] = new Seg();
56     Seg::init(tree[1]);
57     for(; q--;) {
58         cin >> i >> k;
59         if(i == 1)
60             cin >> j, Seg::upd(tree[k], i, j);
61         else if(i == 2)
62             cin >> j,
63             cout << Seg::qry(tree[k], i, j) << "\n";
64         else
65             tree[++ver] = new Seg(*tree[k]);
66     }
67 }

```

1.8. Treap.cpp

```

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

```

```

10 };
11 node treap[400005];
12 int cnt = 0;
13 int root = 0;
14
15 void update(int index) {
16     int lson = treap[index].lson;
17     int rson = treap[index].rson;
18     treap[index].si = treap[lson].si + treap[rson].si + 1;
19     treap[index].sum = treap[lson].sum;
20     treap[index].sum += treap[rson].sum;
21     treap[index].sum += treap[index].val;
22 }
23 void push(int index) {
24     if(!treap[index].tag) return;
25     swap(treap[index].lson, treap[index].rson);
26     int lson = treap[index].lson;
27     int rson = treap[index].rson;
28     treap[lson].tag ^= 1;
29     treap[rson].tag ^= 1;
30     treap[index].tag = 0;
31 }
32 pii split(int rk, int index) {
33     if(!index) return {0, 0};
34     push(index);
35     int lson = treap[index].lson;
36     int rson = treap[index].rson;
37     if(rk <= treap[lson].si) {
38         pii temp = split(rk, lson);
39         treap[index].lson = temp.second;
40         update(index);
41         return {temp.first, index};
42     } else {
43         pii temp = split(rk - treap[lson].si - 1, rson);
44         treap[index].rson = temp.first;
45         update(index);
46         return {index, temp.second};
47     }
48 }
49
50 int merge(int x, int y) {
51     if(!x && !y) return 0;
52     if(!x && y) return y;
53     if(x && !y) return x;
54     push(x);
55     push(y);
56     if(treap[x].prio < treap[y].prio) {
57         treap[x].rson = merge(treap[x].rson, y);
58         update(x);
59         return x;
60     } else {
61         treap[y].lson = merge(x, treap[y].lson);
62         update(y);
63         return y;
64     }
65 }
66
67 void insert(int x, int v) {
68     pii temp = split(x - 1, root);
69     cnt++;
70     treap[cnt].val = v;
71     update(cnt);
72     temp.first = merge(temp.first, cnt);
73     root = merge(temp.first, temp.second);
74 }
75
76 int query(int l, int r) {
77     pii R = split(r, root);
78     pii L = split(l - 1, R.first);
79     int ret = treap[L.second].sum;
80     R.first = merge(L.first, L.second);
81     root = merge(R.first, R.second);
82     return ret;
83 }
84
85 void modify(int l, int r) {
86     pii R = split(r, root);
87     pii L = split(l - 1, R.first);
88     treap[L.second].tag ^= 1;
89     R.first = merge(L.first, L.second);
90     root = merge(R.first, R.second);
91 }

```

2. Math

2.1. CRT.cpp

```

1 #include <bits/stdc++.h>
2 #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    }
40    e[1] = 1;
41    int cnt = 1;
42    vector<int> roughs(B + 1);
43    for(int i = 1; i <= B; ++i)
44        if(!e[i]) roughs[cnt++] = i;
45    roughs[cnt] = 0x7fffffff;
46    for(int i = 1; i <= K; ++i) w[i] = e[i] + w[i - 1];
47    for(int i = 1; i <= K; ++i) s[i] = w[i] - w[i - (i & -i)];
48
49    const auto query = [&] (int x) -> int {
50        int sum = x;
51        while(x) sum -= s[x], x ^= x & -x;
52        return sum;
53    };
54    const auto add = [&] (int x) -> void {

```

```

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

```

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
14 typedef complex<double> comp;
15 const int MAXN = 1000005;
16 const comp I(0, 1);
17 const double PI = acos(-1);
18 comp A[MAXN * 3], B[MAXN * 3], tmp[MAXN * 3], ans[MAXN * 3];
19 void fft(comp F[], int N, int sgn = 1) {
20     if(N == 1) return;
21     memcpy(tmp, F, sizeof(comp) * N);
22     for(int i = 0; i < N; i++)
23         *(i % 2 ? F + i / 2 + N / 2 : F + i / 2) = tmp[i];
24     fft(F, N / 2, sgn), fft(F + N / 2, N / 2, sgn);
25     comp *G = F, *H = F + N / 2;
26     comp cur = 1, step = exp(2 * PI / N * sgn * I);
27     for(int k = 0; k < N / 2; k++) {
28         tmp[k] = G[k] + cur * H[k];
29         tmp[k + N / 2] = G[k] - cur * H[k];
30         cur *= step;
31     }
32     memcpy(F, tmp, sizeof(comp) * N);
33 }
34
35 int main() {
36     int n = read(), m = read(), N = 1 << __lg(n + m + 1) + 1;
37     for(int i = 0; i <= n; ++i) A[i] = read();
38     for(int i = 0; i <= m; ++i) B[i] = read();
39     fft(A, N), fft(B, N);
40     for(int i = 0; i < N; ++i) ans[i] = A[i] * B[i];
41     fft(ans, N, -1);
42     for(int i = 0; i <= n + m; ++i)
43         printf("%d ", int(ans[i].real() / N + 0.1));
44 }

```

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

$$\begin{aligned}\varepsilon &= \mu * 1 \\ \varphi &= \mu * \text{Id}\end{aligned}$$

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

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

2.5.8. ext-Kummer Theorem

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

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 $k \in \mathbb{Z}$, 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 $C_n = \sum_{k=0}^{n-1} C_k C_{n-1-k} = \binom{2n}{n}$. 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

$$\begin{aligned}p \text{ is odd prime, } n \in \mathbb{N}, x \in \mathbb{Z}, y \in \mathbb{Z} \\ p|(x-y), p \nmid x, p \nmid y \\ v_p(x^n - y^n) = v_p(x-y) + v_p(n) \\ v_2(x^n - y^n) = v_2(x-y) + v_2(x+y) + v_2(n) - 1 \quad \text{if } n \text{ is odd} \\ 4|(x-y) \Rightarrow v_2(x+y) = 1, v_2(x^n - y^n) = v_2(x-y) + v_2(n)\end{aligned}$$

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    }

```

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 << 1] = 1;
37             }
38         }
39     }
40     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 }

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

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

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

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 /* 輸入 x,n,p,ans 會將 ans 修改為 x^n%p
13 對整數/矩陣/不要求精度的浮點 皆有效
14 模板第一個型別是 x,ans 第二個是 n,p(應該放 LL 或 __int128)*/
15 ull pri[7] = {2, 325, 9375, 28178,
16               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 = x - 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 }
```

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        }
16    }
17 }
```

```

15     if(i % p == 0) {
16         mu[i * p] = 0;
17         break;
18     } else {
19         mu[i * p] = mu[i] * mu[p];
20     }
21 }
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        if(sgn == -1) {
39            int invN = qpow(N, MOD - 2);
40            for(int i = 0; i < N; ++i) F[i] = (ll)F[i] * invN % MOD;
41        }
42    }
43 }
```

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 /*input x, n, p, ans, will modify ans to x ^ n % p
17 the first is x, ans and the second is n, p (LL or __int128)
18 */
19 ull pri[7] = {2, 325, 9375, 28178,
20               450775, 9780504, 1795265022}; /*2^64*/
21 // int p[3]={2,7,61};/*2^32*/
22 bool check(const ull x, const ull p) {
23     ull d = x - 1, ans = 1;
24     fastpow(p, d, x, ans);
25     if(ans != 1) return 1;
26     for(; !(d & 1);) {
27         d >>= 1;
28         ans = 1;
29         fastpow(p, d, x, ans);
30         if(ans == x - 1)
31             return 0;
32         else if(ans != 1)
33             return 1;
34     }
35 }
```

```

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 template <class T> T pollard_rho(T n) {
73     if(miller_rabin(n)) return n;
74     T p = n;
75     while(p == n) p = rho(n);
76     return max(pollard_rho(p), pollard_rho(n / p));
77 }
78 int main() {
79     LL t, n, ans;
80     for(cin >> t; t--) {
81         cin >> n;
82         ans = pollard_rho(n);
83         if(ans == n)
84             puts("Prime");
85         else
86             printf("%lld\n", ans);
87     }
88 }
```

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     if(!p) return ans;
17     return f(k, now + 1, p - 1, ans);
18 }
19 int main() {
20     ios::sync_with_stdio(0);
21     cin.tie(0);
22     cout.tie(0);
23     ll n, k;
24     cin >> n >> k;
25     for(ll x, i = 0; i < n; ++i) {
26         cin >> x;
27         for(ll e : v) x = min(x, x ^ e);
28         if(x) v.pb(x);
29     }
30     sort(all(v), greater<ll>());
31     ll t = n - v.size(), a = k >> t,
32         b = k & ((1 << min(t, 20LL)) - 1), i = 0;
33     for(; a-- ; ++i)
34         for(ll j = 1 << t, p = f(i); j--;) cout << p << " ";
35     for(i = f(i); b-- ; cout << i << " ");
36 }
```

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) + 1ll * bb * (1 << 15) +
69                      cc) %
70                      p +
71                      p) %
72                      p;
73        }
74    }
75 } // namespace poly
76 using namespace poly;
77 int f[400005], g[400005], h[400005];
78 // 400005 is 2 * (n + m)
79 int main() {
80     int n, m, p;
81     scanf("%d%d%d", &n, &m, &p);
82     for(int i = 0; i <= n; i++) scanf("%d", &f[i]);
83     for(int i = 0; i <= m; i++) scanf("%d", &g[i]);
84     int lim = 1;
85     while(lim <= (n + m)) lim <= 1;
86     mtt(f, g, h, lim, p);
87     for(int i = 0; i <= n + m; i++) printf("%d ", h[i]);
88     return 0;
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 } // 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) {
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;
40     if(j - d[j] + 1 > l) {
41         d[i] = d[j];
42     } else {
43         d[i] = r - i + 1;
44         a = i + d[i] < n;
45         b = i - d[i] >= 0;
46         c = (s[i + d[i]] == s[i - d[i]]);
47         while(a && b && c) {
48             d[i]++;
49             a = i + d[i] < n;
50             b = i - d[i] >= 0;
51             c = (s[i + d[i]] == s[i - d[i]]);
52         }
53         l = i - d[i] + 1;
54         r = i + d[i] - 1;
55     }
56 // cout << d[i] << " ";
57 if(d[i] > d[ans]) {
58     ans = i;
59 }
60 for(int i = ans - d[ans] + 1; i < ans + d[ans]; i++) {
61     if(s[i] ^ '0') {
62         cout << s[i];
63     }
64 }
65 }

}

```

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++) {
20         if(z[i - l] < r - i + 1) {
21             z[i] = z[i - l];
22         } else {
23             z[i] = max(r - i + 1, (int)0);
24             while(i + z[i] < n && s[i + z[i]] == s[z[i]]) {
25                 z[i]++;
26             }
27             l = i;
28             r = i + z[i] - 1;
29             if(z[i] == m) {
30                 ans++;
31             }
32         }
33     }
34     cout << ans;
35 }

```

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) {
20
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 signed main() {
48     ios::sync_with_stdio(0);
49     cin.tie(0);
50     cout.tie(0);
51     cin >> n >> m;
52     while(n--) {
53         char a, b;
54         int x, y;
55         cin >> a >> x >> b >> y;
56         if(a == '-') x = no(x);
57         if(b == '-') y = no(y);
58         v[no(x)].push_back(y);
59         v[no(y)].push_back(x);
60     }
61     for(int i = 1; i <= 2 * m; i++) {
62         if(!d[i]) {
63             dfs(i);
64         }
65     }
66     for(int i = 1; i <= m; i++) {
67         if(scc[i] ^ scc[i + m]) {
68             op[scc[i]] = scc[i + m];
69             op[scc[i + m]] = scc[i];
70         } else {
71             cout << "IMPOSSIBLE";
72             exit(0);
73         }
74     }
75     for(int i = 1; i <= 2 * m; i++) {
76         for(int j : v[i]) {
77             if(scc[i] ^ scc[j]) {
78                 v2[scc[j]].push_back(scc[i]);
79                 ind[scc[i]]++;
80             }
81         }
82     }
83     for(int i = 1; i <= now; i++) {
84         if(!ind[i]) {
85             q.push(i);
86         }
87     }
88     while(!q.empty()) {
89         int k = q.front();
90         q.pop();
91         if(!ans[k]) {
92             ans[k] = 1;
93             ans[op[k]] = 2;
94         }
95         for(int i : v2[k]) {
96             ind[i]--;
97             if(ind[i]) {
98                 q.push(i);
99             }
100        }
101    }
102    for(int i = 1; i <= m; i++) {
103        if(ans[scc[i]] == 1) {
104            cout << "+" << " ";
105        } else {
106            cout << "-" << " ";
107        }
108    }
109 }

```

4.2. Dijkstra.cpp

```

1 vector<pair<int, int>> v[100005], v2[100005];
2 vector<edge> es;
3 int dis1[100005];
4 int dis2[100005];
5 bitset<100005> vis1, vis2;
6
7 void dijkstra(int x, int *dis, vector<pair<int, int>> *v,
8                 bitset<100005> &vis) {
9     priority_queue<pair<int, int>, vector<pair<int, int>>, greater<pair<int, int>> pq;
10    memset(dis, 0x3f, sizeof(dis));
11    vis.reset();
12    dis[x] = 0;
13    pq.push({0, x});
14    while(!pq.empty()) {
15        pair<int, int> now = pq.top();
16        pq.pop();
17        if(vis[now.second]) continue;
18        vis[now.second] = 1;
19        for(auto [i, w] : v[now.second]) {
20            if(vis[i]) continue;
21            if(dis[now.second] + w < dis[i]) {
22                dis[i] = dis[now.second] + w;
23                pq.push({dis[i], i});
24            }
25        }
26    }
27}
28

```

4.3. Dinic.cpp

```

1 #include <bits/stdc++.h>
2 using namespace std;
3 #define ll long long
4 const ll inf = 8e18;
5 #define N 505
6 #define pb push_back
7 struct pp {
8     int from, to;
9     ll cap, flow;
10 };
11 int t, lvl[N], p[N];
12 vector<int> g[N];
13 vector<pp> edge;
14 int bfs(int s) {
15     queue<int> q;
16     for(q.push(s), lvl[s] = 1; !q.empty(); q.pop()) {
17         int u = q.front();
18         for(int e : g[u]) {
19             int v = edge[e].to;
20             if(lvl[v] || !edge[e].flow) continue;
21             lvl[v] = lvl[u] + 1;
22             q.push(v);
23         }
24     }
25     return lvl[t];
26 }
27 ll dfs(int u, ll f = inf) {
28     if(u == t || !f) return f;
29     ll ans = 0;
30     for(int &i = p[u]; i < g[u].size(); ++i) {
31         pp &e = edge[g[u][i]], &b = edge[g[u][i] ^ 1];
32         if(lvl[e.to] == lvl[u] + 1) {
33             ll c = dfs(e.to, min(e.flow, f));
34             e.flow -= c;
35             b.flow += c;
36             f -= c;
37             ans += c;
38         }
39     }
40     return ans;
41 }
42 ll dinic(int s) {
43     ll ans = 0;
44     for(; bfs(s); memset(lvl, 0, sizeof lvl))
45         for(ll k; k = (memset(p, 0, sizeof(p)), dfs(s));)
46             ans += k;
47     return ans;
48 }
49 int main() {
50     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^2 \frac{1}{2} \binom{V}{2}, \sqrt{E}) E
52     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, 離源匯點滿足入度或出度
54     for(cin >> n >> m; m--) { //都是 1 則為 E \sqrt{V}
55         int u, v;
56         ll f;
57         cin >> u >> v >> f;
58         g[u].pb(cnt++);
59         g[v].pb(cnt++);
60     }
61 }

```

```

61     edge.pb({u, v, f, f});
62     edge.pb({v, u, 0, 0});
63 }
64 t = n;
65 cout << dinic(1);
66 }

```

4.4. MaximumFlow.cpp

```

1 #include <bits/stdc++.h>
2 #define int long long
3 using namespace std;
4
5 int n, m;
6 vector<int> v[1005];
7 int head[1005];
8 int c[1005][1005];
9 int lv[1005];
10 int ans = 0;
11
12 bool bfs() {
13     memset(head, 0, sizeof(head));
14     memset(lv, 0, sizeof(lv));
15     queue<int> q;
16     q.push(1);
17     while(!q.empty()) {
18         int now = q.front();
19         q.pop();
20         if(now == n) continue;
21         for(int i : v[now]) {
22             if(i != 1 && c[now][i] && !lv[i]) {
23                 lv[i] = lv[now] + 1;
24                 q.push(i);
25             }
26         }
27     }
28     return lv[n];
29 }
30
31 int dfs(int x, int flow) {
32     int ret = 0;
33     if(x == n) return flow;
34     for(int i = head[x]; i < v[x].size(); i++) {
35         int y = v[x][i];
36         head[x] = y;
37         if(c[x][y] && lv[y] == lv[x] + 1) {
38             int d = dfs(y, min(flow, c[x][y]));
39             flow -= d;
40             c[x][y] -= d;
41             c[y][x] += d;
42             ret += d;
43         }
44     }
45     return ret;
46 }
47
48 signed main() {
49     cin >> n >> m;
50     while(m--) {
51         int x, y, z;
52         cin >> x >> y >> z;
53         if(c[x][y] || c[y][x]) {
54             c[x][y] += z;
55             continue;
56         }
57         v[x].push_back(y);
58         v[y].push_back(x);
59         c[x][y] = z;
60     }
61     while(bfs()) {
62         ans += dfs(1, INT_MAX);
63     }
64     cout << ans;
65 }

```

4.5. SCC.cpp

```

1 int n, m;
2 vector<int> v[100005];
3 int d[100005];
4 int low[100005];
5 int cnt = 0;
6 stack<int> s;
7 int scc[100005];
8 int now = 0;
9
10 void dfs(int x) {
11     d[x] = low[x] = ++cnt;
12     s.push(x);
13     for(int i : v[x]) {
14         if(scc[i]) continue;
15         if(d[i]) {
16             low[x] = min(low[x], d[i]);
17         }
18     }
19 }
20
21 int main() {
22     int n, m;
23     cin >> n >> m;
24     vector<vector<int>> adj(n);
25     for(int i = 0; i < m; i++) {
26         int u, v;
27         cin >> u >> v;
28         adj[u].push_back(v);
29     }
30     for(int i = 0; i < n; i++) {
31         if(d[i] == 0) {
32             dfs(i);
33         }
34     }
35     for(int i = 0; i < n; i++) {
36         if(d[i] == low[i]) {
37             int scc_id = ++cnt;
38             while(scc_id <= low[i]) {
39                 scc[s.pop()] = scc_id;
40             }
41         }
42     }
43     cout << "SCC Count: " << cnt << endl;
44 }

```

```

17     } else {
18         dfs(i);
19         low[x] = min(low[x], low[i]);
20     }
21     if(d[x] == low[x]) {
22         now++;
23         while(!s.empty()) {
24             int k = s.top();
25             s.pop();
26             scc[k] = now;
27             if(k == x) break;
28         }
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[u]) 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;
30     }
31 }
32

```

```

31         temp = a[temp];
32     }
33 }
34 void dfs(int x) {
35     if(vis2[x]) return;
36     vis2[x] = 1;
37     for(int i : v[x]) {
38         dfs(i);
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
5 struct CHO {
6     deque<line> dq;
7     int intersect(line x, line y) {
8         int d1 = x.b - y.b;
9         int d2 = y.a - x.a;
10        return d1 / d2;
11    }
12    bool check(line x, line y, line z) {
13        int I12 = intersect(x, y);
14        int I23 = intersect(y, z);
15        return I12 < I23;
16    }
17    void insert(int a, int b) {
18        if(!dq.empty() && a == dq.back().a) return;
19        while(dq.size() >= 2 && !check(dq[dq.size() - 2], dq[dq.size() - 1],
20                                         {a, b})) {
21            dq.pop_back();
22        }
23        dq.push_back({a, b});
24    }
25    void update(int x) {
26        while(dq.size() >= 2 && dq[0].y(x) >= dq[1].y(x)) {
27            dq.pop_front();
28        }
29    }
30    int query(int x) {
31        update(x);
32        return dq.front().y(x);
33    }
34 }
35

```

5.2. Li-Chao-SegmentTree.cpp

```

1 struct line {
2     int a, b = 1000000000000000000;
3 };

```

```

3     int y(int x) { return a * x + b; }
4 };
5
6 line tree[4000005];
7 int n, x;
8 int s[200005];
9 int f[200005];
10 int dp[200005];
11
12 void update(line ins, int l = 1, int r = 1e6, int index = 1) {
13     if(l == r) {
14         if(ins.y(l) < tree[index].y(l)) {
15             tree[index] = ins;
16         }
17         return;
18     }
19     int mid = (l + r) >> 1;
20     if(tree[index].a < ins.a) swap(tree[index], ins);
21     if(tree[index].y(mid) > ins.y(mid)) {
22         swap(tree[index], ins);
23         update(ins, l, mid, index << 1);
24     } else {
25         update(ins, mid + 1, r, index << 1 | 1);
26     }
27 }
28
29 int query(int x, int l = 1, int r = 1000000, int index = 1) {
30     int cur = tree[index].y(x);
31     if(l == r) {
32         return cur;
33     }
34     int mid = (l + r) >> 1;
35     if(x <= mid) {
36         return min(cur, query(x, l, mid, index << 1));
37     } else {
38         return min(cur, query(x, mid + 1, r, index << 1 | 1));
39     }
}

```

5.3. SOSDP.cpp

```

1 for(int i = 0; i < 20; ++i)
2     for(int j = i; j < N; ++j)
3         if(j >> i & 1) dp[j] += dp[j ^ (1 << i)]; // subset
4 for(int i = 0; i < 20; ++i)
5     for(int j = 0; j < N; ++j)
6         if(!(j >> i & 1))
7             dp2[j] += dp2[j | (1 << i)]; // superset

```

6. Geometry

6.1. 164253Version.cpp

```

1 #include <bits/stdc++.h>
2 using namespace std;
3 #define ll long long
4 #define pb push_back
5 #define pll pair<int, int>
6 #define pdd pair<double, double>
7 #define pll pair<ll, ll>
8 #define F first
9 #define S second
10 #define eps 1e-6
11 int sign(double x) {
12     return fabs(x) < eps ? 0 : x > 0 ? 1 : -1;
13 }
14 int sign(ll x) { return !x ? 0 : x > 0 ? 1 : -1; }
15 template <typename T1, typename T2>
16 istream &operator>>(istream &s, pair<T1, T2> &p) {
17     auto &[a, b] = p;
18     s >> a >> b;
19     return s;
20 }
21 template <typename T1, typename T2>
22 ostream &operator<<(ostream &s, const pair<T1, T2> &p) {
23     auto &[a, b] = p;
24     s << a << " " << b;
25     return s;
26 }
27 pll operator+(const pll a, const pll b) {
28     return {a.F + b.F, a.S + b.S};
29 }
30 pll operator-(const pll a, const pll b) {
31     return {a.F - b.F, a.S - b.S};
32 }
33 pll operator-(const pll a) { return {-a.F, -a.S}; }
34 pll operator*(const pll a, const pll b) {
35     return {(ll)a.F * b.F, (ll)a.S * b.S};
36 }
37 pdd operator/(const pll a, const double x) {
38     return {a.F / x, a.S / x};
39 }

```

```

41 pdd operator*(const pll a, const double x) {
42     return {a.F * x, a.S * x};
43 }
44 pdd operator*(const double x, const pll a) {
45     return {a.F * x, a.S * x};
46 }
47 // 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
48 ll len2(pll p) {
49     return (ll)p.F * p.F + (ll)p.S * p.S;
50 }
51 ll cross(pll a, pll b) {
52     return (ll)a.F * b.S - (ll)a.S * b.F;
53 }
54 ll cross(pll p1, pll p2, pll p3) {
55     return cross(p2 - p1, p3 - p1);
56 }
57 ll dot(pll a, pll b, pll c) {
58     return (ll)(b.F - a.F) * (c.F - a.F) +
59         (ll)(b.S - a.S) * (c.S - a.S);
60 }
61 ll ori(pll p1, pll p2, pll p3) {
62     return sign(cross(p1, p2, p3));
63 }
64 bool btw(pll p1, pll p2, pll p3) {
65     return ori(p3, p1, p2) == 0 && dot(p3, p1, p2) <= 0;
66 }
67 bool banana(pll p1, pll p2, pll p3,
68             pll p4) { // 檢兩線段是否香蕉
69     if(btw(p1, p2, p3) || btw(p1, p2, p4) || btw(p3, p4, p1) ||
70         btw(p3, p4, p2))
71         return true;
72     return ori(p1, p2, p3) * ori(p1, p2, p4) < 0 &&
73         ori(p3, p4, p1) * ori(p3, p4, p2) < 0;
74 }
75 pdd banana_point(pll p1, pll p2, pll p3,
76                   pll p4) { // 分點，算的是無限延伸直線的交點
77     return cross(p2 - p1, p4 - p1) /
78         (double)cross(p2 - p1, p4 - p3) * p3 -
79         cross(p2 - p1, p3 - p1) /
80         (double)cross(p2 - p1, p4 - p3) * p4;
81 }
82 pdd proj(pll p1, pll p2, pll p3) {
83     return dot(p1, p2, p3) / (double)len2(p2 - p1) * (p2 - p1);
84 }
85 double min_dis(pll p1, pll p2,
86                 pll p3) { // min distance of p3 to segment p1,p2
87     if(dot(p1, p2, p3) < 0 || dot(p2, p1, p3) < 0)
88         return min(len(p3 - p1), len(p3 - p2));
89     return abs(cross(p1, p2, p3)) / len(p2 - p1);
90 }
91 ll area2(vector<pll> &v) { // 傳入一個多邊形順序的點集
92     // 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
94     int n = v.size() - 1;
95     ll ans = 0;
96     for(int i = 0; i < n; ++i) ans += cross(v[i], v[i + 1]);
97     return abs(ans);
98 }
99 int in_polygon(vector<pll> &v,
100                pll p) { // 傳入多邊形，起點要出現兩次，回傳
101    // {-1:in, 0:on, 1:out}
102    int n = v.size() - 1, ans = 1;
103    for(int i = 0; i < n; ++i)
104        if(btw(v[i], v[i + 1], p)) return 0;
105    for(int i = 0; i < n; ++i)
106        if(banana(v[i], v[i + 1], p, {(ll)2e9 + 7, p.S + 1LL}))
107            ans *= -1;
108    // 對於任意 p 到 {W, p.S+1}
109    // 的向量中不會有整數點存在，其中需要滿足 {W, p.S+1}
110    // 必須很遠，保證在多邊形外
111    return ans;
112 }
113 void solve() {
114     int n;
115     cin >> n;
116     vector<pll> v(n);
117     for(pll &e : v) cin >> e;
118     v.pb(v[0]);
119     ll ans = area2(v) + 2, ans2 = 0;
120     for(int i = 0; i < n; ++i) {
121         if(v[i].F == v[i + 1].F)
122             ans2 += abs(v[i].S - v[i + 1].S);
123         else if(v[i].S == v[i + 1].S)
124             ans2 += abs(v[i].F - v[i + 1].F);
125         else
126             ans2 += gcd(abs(v[i].F - v[i + 1].F),
127                         abs(v[i].S - v[i + 1].S));
128     }
129     cout << (ans - ans2) / 2 << " " << ans2;
130 }
131 int main() {

```

```

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

```

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 &&

```

```

25         dir(z - w, x - w) * dir(z - w, y - w) == -1) {
26             return 1;
27         }
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[i].first - v[l].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);
37   }
38   if(L > mid) {
39     return query(L, R, mid + 1, r, index << 1 | 1);
40   }
41   return max(query(L, mid, l, mid, index << 1),
42             query(mid + 1, R, mid + 1, r, index << 1 | 1));
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 void dfs(int x, int pre) {
59   si[x] = 1;
60   for(int i : v[x]) {
61     if(i == pre) continue;
62     p[i] = x;
63     d[i] = d[x] + 1;
64     dfs(i, x);
65     si[x] += si[i];
66   }
67 }

68 void dfs2(int x, int pre, int t) {
69   tp[x] = t;
70   st[x] = ++cnt;
71   int ma = 0;
72   for(int i : v[x]) {
73     if(i == pre) continue;
74     if(si[i] > si[ma]) {
75       ma = i;
76     }
77   }
78   if(!ma) return;
79   dfs2(ma, x, t);
80   for(int i : v[x]) {
81     if(i == pre || i == ma) {
82       continue;
83     }
84     dfs2(i, x, i);
85   }
86 }

87 int f(int x, int y) {
88   int ret = 0;
89   while(tp[x] ^ tp[y]) {
90     if(d[tp[x]] < d[tp[y]]) {
91       swap(x, y);
92     }
93     ret = max(ret, query(st[tp[x]], st[x]));
94     x = p[tp[x]];
95   }
96   if(d[x] > d[y]) swap(x, y);
97   ret = max(ret, query(st[x], st[y]));
98   return ret;
99 }

100 signed main() {
101   ios::sync_with_stdio(0);
102   cin.tie(0);
103   cout.tie(0);
104   cin >> n >> q;
105   for(int i = 1; i <= n; i++) {
106     cin >> a[i];
107   }
108   for(int i = 1; i < n; i++) {
109     int x, y;
110     cin >> x >> y;
111     v[x].push_back(y);
112     v[y].push_back(x);
113   }
114   dfs(1, 0);
115   dfs2(1, 0, 1);
116   for(int i = 1; i <= n; i++) {
117     b[st[i]] = a[i];
118   }
119   build();
120   while(q--) {
121     int mode, x, y;
122     cin >> mode >> x >> y;
123     if(mode == 1) {
124
125
126
127

```

```

129         modify(st[x], y);
130       } else {
131         cout << f(x, y) << " ";
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;
13 num() {
14     memset(p, 0, sizeof(p));
15     len = 0;
16 }
17 num(LL x) {
18     memset(p, 0, sizeof(p));
19     len = 0;
20     for(p[len++] = x; p[len - 1] >= base; ++len)
21         p[len] = p[len - 1] / base, p[len - 1] %= base;
22 }
23 num operator=(LL x) {
24     memset(p, 0, sizeof(p));
25     len = 0;
26     for(p[len++] = x; p[len - 1] >= base; ++len)
27         p[len] = p[len - 1] / base, p[len - 1] %= base;
28     return *this;
29 }
30 num max(const num &b) {
31     if(len != b.len) return len > b.len ? *this : b;
32     for(int i = len; i--;) {
33         if(p[i] != b.p[i]) return p[i] > b.p[i] ? *this : b;
34     }
35 }
36 num operator+(const num &b) {
37     num c;
38     LL x = 0;
39     for(LL &i = c.len; i < len || i < b.len; ++i) {
40         c.p[i] = p[i] + b.p[i] + x;
41         x = c.p[i] / base;
42         c.p[i] %= base;
43     }
44     if(x) c.p[c.len++] = x;
45     return c;
46 }
47 num operator*(LL b) {
48     num c;
49     c.len = len;
50     LL x = 0;
51     for(LL i = 0; i < len; ++i) {
52         c.p[i] = p[i] * b + x;
53         x = c.p[i] / base;
54         c.p[i] %= base;
55     }
56     for(; x; x /= base) c.p[c.len++] = x % base;
57     return c;
58 }
59 } dp[N][N], ans;
60 ostream &operator<<(ostream &s, num a) {
61     if(!a.len) return s << "0";
62     s << a.p[a.len - 1];
63     for(int i = a.len - 1; i--;) {
64         if(!a.p[i])
65             s << "00000000";
66         else {
67             for(int k = 10; k * a.p[i] < (LL)1e9; k *= 10)
68                 s << "0";
69             s << a.p[i];
70         }
71     }
72     return s;
73 }
74 LL a[N];
75 int main() {
76     ios::sync_with_stdio(0);
77     cin.tie(0);
78     cout.tie(0);
79     int n, m, i, j;
80     for(cin >> n >> m; n--) {
81         for(i = 0; i < m; ++i) cin >> a[i];
82         for(i = 0; i < m; ++i)
83             for(j = 0; j < m; ++j) dp[i][j] = 0;
84         for(i = 0; i < m; ++i) dp[i][i] = a[i] << 1;
85         for(j = 1; j < m; ++j)
86             for(i = 0; i + j < m; ++i)
87                 dp[i][i + j] =
88                     (dp[i][i + j - 1] + a[i + j]) *
89                     .max(dp[i + 1][i + j] + a[i]) *
90                     2;
91         ans = ans + dp[0][m - 1];
92     }
93     cout << ans;
94 }
```

8.2. Tri-search.cpp

```

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

```

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 template <class T> T bi_search(T l, T r, T end) {
17     if(!check(r - end)) return r - end;
18     for(; r - l > end;) {
19         T mid = (l + r) / 2;
20         if(check(mid))
21             r = mid;
22         else
23             l = mid;
24     }
25     return l;
26 }
27 /*check gives 00000001111 find the last 0*/
28 template <class T> T tri_search(T l, T r, T end) {
29     T midl, midr;
30     for(;;) {
31         midl = (l + r) / 2;
32         midr = (midl + r) / 2;
33         if(midl - midr < end) break;
34         if(get(midr) > get(midl))
35             r = midr;
36         else
37             l = midl;
38     }
39     for(; r - l > end;) {
40         midl = (l + r) / 2;
41         if(get(r) > get(l))
42             r = midl;
43         else
44             l = midl;
45     }
46     return l;
47 }
48 /*get gives the value, find the minimum*/
49 int main() {
50     cin >> n >> x >> y;
51     for(int i = n; i >= 0; i--) {
52         cin >> a[i];
53     }
54     cout << fixed << setprecision(7)
55         << tri_search<double>(x, y, 1e-7);
56 }
```

8.3. 對拍.md

```

1 .vimrc
2 `````
3 set ts=4
4 set autoindent
5 `````
6
7 script
8 `````
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
# factor n 可以質因數分解
20 `````
21
22 python random
23 `````
24 from random import *
25 n = randint(1, 100) # 隨機產生 1~100 的整數
26 ch = chr(ord('a') + randint(0, 25)) # 隨機產生 'a'~'z' 其中一個字元
27 choiceSet = sample(s, 4) # 從集合 s 選出 4 個不同的元素
28 choiceSet = sample(range(1, n+1), 4) # 從整數 1~n 選出 4 個不同的元素
29 shuffle(arr) # 把序列 arr 順序打亂
30 `````
31
32 python tree
33 `````
34 from random import *
35 n = randint(3, 6) # 隨機產生 3~6 的整數
36 print(n)
37 for i in range(2, n+1):
38     print(randint(1, i-1), i)
39 `````

```

```

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)
48
49 print(n, m)
50
51 edge = list()
52
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
60 #add extra edge
61 for i in range(m-(n-1)):
62     x = randint(1, n)
63     y = randint(1, n)
64     while x == y or [min(x, y), max(x, y)] in edge:
65         x = randint(1, n)
66         y = randint(1, n)
67     print(x, y)
68     edge.append([min(x, y), max(x, y)])
69
70
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 ````
```

```

17         dsu[a] = b, size[b] += size[a];
18     else
19         dsu[b] = a, size[a] += size[b];
20     }
21 #undef N
22 };
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    {
22        sz = 1;
23        mx = mn = sum = val;
24        if(l)
25            sz += l->sz, mx = max(mx, l->mx),
26            mn = min(mn, l->mn), sum += l->sum;
27        if(r)
28            sz += r->sz, mx = max(mx, r->mx),
29            mn = min(mn, r->mn), sum += r->sum;
30    }
31    void push()
32    {
33        if(rev_tag) swap(l, r);
34        if(l) l->add_tag += add_tag, l->rev_tag ^= rev_tag;
35        if(r) r->add_tag += add_tag, r->rev_tag ^= rev_tag;
36        mx += add_tag;
37        mn += add_tag;
38        sum += add_tag;
39        add_tag = 0;
40        rev_tag = 0;
41    }
42    void debug(node *tr) {
43        if(!tr) return;
44        tr->push();
45        tr->pull();
46        debug(tr->l);
47        cout << tr->val << " ";
48        debug(tr->r);
49    }
50    void debug2(node *tr) {
51        if(!tr) return;
52        tr->push();
53        tr->pull();
54        cout << tr->val << " ";
55        debug2(tr->l);
56        debug2(tr->r);
57    }
58    int sz(node *tr) { return tr ? tr->sz : 0; }
59    node *merge(node *a, node *b) {
60        if(!a || !b) return a ?: b;
61        a->push();
62        b->push();
63        if(mt() % (sz(a) + sz(b)) < sz(a)) {
64            a->r = merge(a->r, b);
65            a->pull();
66            return a;
67        }
68        b->l = merge(a, b->l);
69        b->pull();
70        return b;
71    }
72    pnn split(node *tr, int v) { //(-inf,v],(v,inf)
73        if(!tr) return {0, 0};
74        tr->push();
75        if(tr->val <= v) {
76            auto [l, r] = split(tr->r, v);
77            tr->r = l;
78            tr->pull();
79            return {tr, r};
80        }
81        auto [l, r] = split(tr->l, v);
82        tr->l = r;
83        tr->pull();
84        return {l, tr};
85    }

```

9. Another Version Data Structure

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

```

```

85 } pnn splitsz(node *tr, int k) { // [rk.l, rk.k], (rk.k, rk.n]
86     if(!tr || sz(tr) <= k) return {tr, 0};
87     tr->push();
88     if(k <= sz(tr->l)) {
89         auto [l, r] = splitsz(tr->l, k);
90         tr->l = r;
91         tr->pull();
92         return {l, tr};
93     } else if(k <= sz(tr->l) + 1) {
94         auto r = tr->r;
95         tr->r = 0;
96         tr->pull();
97         return {tr, r};
98     } else {
99         auto [l, r] = splitsz(tr->r, k - (sz(tr->l) + 1));
100        tr->r = l;
101        tr->pull();
102        return {tr, r};
103    }
104 }
105 node *insert(node *tr, int v) {
106     auto [l, r] = split(tr, v);
107     return merge(merge(l, new node(v)), r);
108 }
109 node *insertkth(node *tr, int k) {
110     auto [l, r] = splitsz(tr, k - 1);
111     return merge(merge(l, new node(0)),
112                  r); // new node 拿來區間操作初始化
113 }
114 node *eraseall(node *tr, int v) {
115     auto [l, r] = split(tr, v - 1);
116     return merge(l, split(r, v).S);
117 }
118 node *eraseone(node *tr, int v) {
119     auto [l, r] = split(tr, v - 1);
120     return merge(l, splitsz(r, 1).S);
121 }
122 node *erasekth(node *tr, int k) {
123     auto [l, r] = splitsz(tr, k - 1);
124     return merge(l, splitsz(r, k).S);
125 }
126 int rk(node *tr, int v) {
127     if(!tr) return 0;
128     if(tr->val <= v) return sz(tr->l) + 1 + rk(tr->r, v);
129     return rk(tr->l, v);
130 }
131 int kth(node *&tr, int k) {
132     auto [l, x] = splitsz(tr, k - 1);
133     auto [m, r] = splitsz(x, 1);
134     if(!m) return 0;
135     int ans = m->val;
136     tr = merge(merge(l, m), r);
137     return ans;
138 }
139 int count(node *&tr, int L, int R) { // count[L,R]
140     auto [l, x] = split(tr, L - 1);
141     auto [m, r] = split(x, R);
142     int ans = m->sum; // 看要改啥
143     tr = merge(merge(l, m), r);
144     return ans;
145 }
146 int countkth(node *&tr, int L, int R) { // count[rk.L,rk.R]
147     auto [l, x] = splitsz(tr, L - 1);
148     auto [m, r] = splitsz(x, R - L);
149     int ans = m->sum; // 看要改啥
150     tr = merge(merge(l, m), r);
151     return ans;
152 }
153 int prev(node *&tr, int v) {
154     auto [x, r] = split(tr, v - 1);
155     auto [l, m] = splitsz(x, sz(x) - 1);
156     int ans = m->val;
157     tr = merge(merge(l, m), r);
158     return ans;
159 }
160 int next(node *&tr, int v) {
161     auto [l, x] = split(tr, v);
162     auto [m, r] = splitsz(x, 1);
163     int ans = m->val;
164     tr = merge(merge(l, m), r);
165     return ans;
166 }
167 int qry(node *&tr, int L, int R) { // qry[L,R]
168     auto [x, r] = splitsz(tr, R);
169     auto [l, m] = splitsz(x, L - 1);
170     int ans = m->sum; // 看要改啥
171     tr = merge(merge(l, m), r);
172     return ans;
173 }
174 void modify(node *&tr, int L, int R, int v) { // modify[L,R]
175     auto [x, r] = splitsz(tr, R);
176     auto [l, m] = splitsz(x, L - 1);

```

```

177     m->val += v;
178     m->add_tag += v;
179     m->rev_tag = 1; // 看要改啥
180     tr = merge(merge(l, m), r);
181 }
182 int main() {
183     int t;
184     node *tr = 0;
185     for(cin >> t; t--;) {
186         int op, x;
187         cin >> op >> x;
188         switch(op) {
189             case 1:
190                 tr = insert(tr, x);
191                 break;
192             case 2:
193                 tr = eraseone(tr, x);
194                 break;
195             case 3:
196                 cout << rk(tr, x - 1) + 1 << "\n";
197                 break;
198             case 4:
199                 cout << kth(tr, x) << "\n";
200                 break;
201             case 5:
202                 cout << prev(tr, x) << "\n";
203                 break;
204             case 6:
205                 cout << next(tr, x) << "\n";
206                 break;
207         }
208     }
209 }

```

4.4. 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
2 #include <bits/stdc++.h>
3 using namespace std;
4 #define pnn pair<node *, node *>
5 #define F first
6 #define S second
7 #define int long long
8 mt19937 mt(hash<string>()("official_beautiful_fruit"));
9 struct node {
10     node *l, *r;
11     int val, sz;
12     int mx, mn, sum, num;
13     int rev_tag, add_tag;
14     node(int _val = 0, int _num = 1)
15         : val(_val), l(0), r(0), sz(1), sum(_num), num(_num),
16           mx(_val), mn(_val), rev_tag(0), add_tag(0) {}
17     node(node *tr)
18         : val(tr->val), l(tr->l), r(tr->r), sz(tr->sz) {}
19     void pull() {
20         sz = 1;
21         mx = mn = sum = num;
22         if(l)
23             sz += l->sz, mx = max(mx, l->mx),
24             mn = min(mn, l->mn), sum += l->sum;
25         if(r)
26             sz += r->sz, mx = max(mx, r->mx),
27             mn = min(mn, r->mn), sum += r->sum;
28     }
29     void push() {
30         if(rev_tag) swap(l, r);
31         if(l) l->add_tag += add_tag, l->rev_tag ^= rev_tag;
32         if(r) r->add_tag += add_tag, r->rev_tag ^= rev_tag;
33         mx += add_tag;
34         mn += add_tag;
35         sum += add_tag;
36         add_tag = 0;
37         rev_tag = 0;
38     }
39 };
40 void debug(node *tr) {
41     if(!tr) return;
42     debug(tr->l);
43     cout << tr->val << " ";
44     debug(tr->r);
45 }
46 void debug2(node *tr) {
47     if(!tr) return;
48     cout << tr->val << " ";
49     debug2(tr->l);
50     debug2(tr->r);
51 }
52 int sz(node *tr) { return tr ? tr->sz : 0; }
53 node *merge(node *a, node *b) {
54     if(!a || !b) return a ?: b;
55     if(mt() % (sz(a) + sz(b)) < sz(a)) {
56         a->r = merge(a->r, b);
57         a->pull();
58     } else {
59         b->l = merge(a, b->l);
60         b->pull();
61     }
62     return a;
63 }

```

```

59     return a;
60 }
61 b->l = merge(a, b->l);
62 b->pull();
63 }  

64 pnn split(node *tr, int v) { //(-inf,v],(v,inf)
65     if(!tr) return {0, 0};
66     tr->push();
67     if(tr->val <= v) {
68         auto [l, r] = split(tr->r, v);
69         tr->r = l;
70         tr->pull();
71         return {tr, r};
72     }
73     auto [l, r] = split(tr->l, v);
74     tr->l = r;
75     tr->pull();
76     return {l, tr};
77 }  

78 pnn splitsz(node *tr, int k) { //[rk.1,rk.k],(rk.k,rk.n]
79     if(!tr || sz(tr) <= k) return {tr, 0};
80     tr->push();
81     if(k <= sz(tr->l)) {
82         auto [l, r] = splitsz(tr->l, k);
83         tr->l = r;
84         tr->pull();
85         return {l, tr};
86     } else if(k <= sz(tr->l) + 1) {
87         auto r = tr->r;
88         tr->r = 0;
89         tr->pull();
90         return {tr, r};
91     } else {
92         auto [l, r] = splitsz(tr->r, k - (sz(tr->l) + 1));
93         tr->r = l;
94         tr->pull();
95         return {tr, r};
96     }
97 }  

98 node *insert(node *tr, int val = 0, int num = 1) {
99     auto [l, r] = split(tr, val);
100    return merge(merge(l, new node(val, num)), r);
101 }  

102 node *insertkth(node *tr, int k) {
103     auto [l, r] = splitsz(tr, k - 1);
104     return merge(merge(l, new node()), r); // new node 拿來區間操作初始化
105 }  

106 node *eraseall(node *tr, int v) {
107     auto [l, r] = split(tr, v - 1);
108     return merge(l, split(r, v).S);
109 }  

110 node *eraseone(node *tr, int v) {
111     auto [l, r] = split(tr, v - 1);
112     return merge(l, splitsz(r, 1).S);
113 }  

114 node *erasekth(node *tr, int k) {
115     auto [l, r] = splitsz(tr, k - 1);
116     return merge(l, splitsz(r, k).S);
117 }  

118 int rnk(node *tr, int v) {
119     if(!tr) return 0;
120     if(tr->val <= v) return sz(tr->l) + 1 + rnk(tr->r, v);
121     return rnk(tr->l, v);
122 }  

123 int kth(node *&tr, int k) {
124     auto [l, x] = splitsz(tr, k - 1);
125     auto [m, r] = splitsz(x, 1);
126     if(!m) return 0;
127     int ans = m->val;
128     tr = merge(merge(l, m), r);
129     return ans;
130 }  

131 int count(node *&tr, int L, int R) { // count[L,R]
132     auto [l, x] = split(tr, L - 1);
133     auto [m, r] = split(x, R);
134     int ans = m->sum; // 看要改啥
135     tr = merge(merge(l, m), r);
136     return ans;
137 }  

138 int countkth(node *&tr, int L, int R) { // count[rk.L,rk.R]
139     auto [l, x] = splitsz(tr, L - 1);
140     auto [m, r] = splitsz(x, R - L);
141     int ans = m->sum; // 看要改啥
142     tr = merge(merge(l, m), r);
143     return ans;
144 }  

145 int prev(node *&tr, int v) {
146     auto [x, r] = split(tr, v - 1);
147     auto [l, m] = splitsz(x, sz(x) - 1);
148     int ans = m->val;
149     tr = merge(merge(l, m), r);

```

```

151     return ans;
152 }  

153 int next(node *&tr, int v) {
154     auto [l, x] = split(tr, v);
155     auto [m, r] = splitsz(x, 1);
156     int ans = m->val;
157     tr = merge(merge(l, m), r);
158     return ans;
159 }  

160 int qry(node *&tr, int L, int R) { // qry[L,R]
161     auto [l, x] = splitsz(tr, L - 1);
162     auto [m, r] = splitsz(x, R);
163     int ans = m->sum; // 看要改啥
164     tr = merge(merge(l, m), r);
165     return ans;
166 }  

167 void modify(node *&tr, int L, int R, int v) { // modify[L,R]
168     auto [l, x] = splitsz(tr, L - 1);
169     auto [m, r] = splitsz(x, R);
170     m->val += v;
171     m->add_tag += v; // 看要改啥
172     tr = merge(merge(l, m), r);
173 }  

174 signed main() {
175     vector<node *> tr(2);
176     int n, m;
177     scanf("%lld%lld", &n, &m);
178     for(int i = 1, x; i <= n; ++i)
179         scanf("%lld", &x), (x) && (tr[1] = insert(tr[1], i, x));
180     for(; m--;) {
181         int op = -1, p = -1, x = -1, y = -1;
182         scanf("%lld", &op);
183         if(!op) {
184             scanf("%lld%lld%lld", &p, &x, &y);
185             auto [l, tmp] = split(tr[p], x - 1);
186             auto [m, r] = split(tmp, y);
187             tr[p] = merge(l, r);
188             tr.push_back(m);
189         } else if(op == 1) {
190             scanf("%lld%lld", &p, &x);
191             // cout<<kth(tr[x],1)<<"\n";//break;
192             auto [l, r] = split(tr[p], kth(tr[x], 1));
193             tr[p] = merge(merge(l, tr[x]), r);
194         } else {
195             switch(op) {
196                 case 2:
197                     scanf("%lld%lld%lld", &p, &x, &y);
198                     tr[p] = insert(tr[p], y, x);
199                     break;
200                 case 3:
201                     scanf("%lld%lld%lld", &p, &x, &y);
202                     printf("%lld\n", count(tr[p], x, y));
203                     break;
204                 case 4:
205                     scanf("%lld%lld", &p, &x);
206                     printf("%lld\n", kth(tr[p], x));
207                     break;
208             }
209         }
210     }
211 }
```

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<<",
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

```

```

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]]]
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) /
70                         (x1 - x0),
71                         x0, x1, p);
72     else
73       line[p] = Line(0, max(y0, y1), x0, x1, p);
74     ins(p);
75     ++p;
76   } else {
77     int k;
78     cin >> k;
79     k = (k + ans - 1) % 39989 + 1;
80     cout << (ans = qry(k)) << "\n";
81   }
82 }
83 // cout<<qry(9)<<"\n";

```

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     getchar(), scanf("%d", a + i),
43     upd(a[i], 1, 1, MAX, root);
44   // printf("%d %d %d %d\n"
45   // "<<L<<\n", qry(2,2,1,n,1),qry(3,3,1,n,1),qry(5,5,1,n,1),qry(5,5,1,
46   for(; q--;) {
47     getchar();
48     char c = getchar();
49     scanf(" %d %d", &x, &i);
50     if(c == '!')
51       upd(a[x], -1, 1, MAX, root),
52       a[x] = i, upd(i, 1, 1, MAX, root);
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 #undef lid
36 #undef rid
37 #undef N
38 };
39 /*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,
28         LL id = 1) {
29     if(l <= L && R <= r) {
30         addtag(id, v, L, R);
31         return;
32     }
33     push(id, L, R);
34     if(r <= M)
35         upd(l, r, v, L, M, lid);
36     else if(M + 1 <= l)
37         upd(l, r, v, M + 1, R, rid);
38     else
39         upd(l, M, v, L, M, lid),
40             upd(M + 1, r, v, M + 1, R, rid);
41     pull(id);
42 }
43 LL qry(int l, int r, int L = 1, int R = n, int id = 1) {
44     if(l <= L && R <= r) return seg[id];
45     push(id, L, R);
46     if(r <= M) return qry(l, r, L, M, lid);
47     if(M + 1 <= l) return qry(l, r, M + 1, R, rid);
48     return qry(l, M, L, M, lid) +
49             qry(M + 1, r, M + 1, R, rid);
50 }
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
2 #include <bits/stdc++.h>
3 using namespace std;
4 #define N 50005
5 struct Line {
6     double a, b; // ax+b
7     Line(double _a = -1, double _b = -1e6)
8         : a(_a), b(_b - _a) {}
9     double operator()(int x) { return a * x + b; }
10 } seg[N < 2];
11 #define lid (id << 1)
12 #define rid (id << 1 | 1)
13 #define M (L + R >> 1)
14 void ins(Line l, int L = 1, int R = N, int id = 1) {
15     if(L == R) {
16         if(seg[id].a < 0 || l(M) > seg[id](M)) seg[id] = l;
17         return;
18     }
19     if(l(M) > seg[id](M)) swap(l, seg[id]);
20     if(l.a > seg[id].a)
21         ins(l, M + 1, R, rid);
22     else
23         ins(l, L, M, lid);
24 }
25 double qry(int x, int L = 1, int R = N, int id = 1) {
26     if(L == R) return seg[id](x);
27     if(x <= M) return max(qry(x, L, M, lid), seg[id](x));
28     return max(seg[id](x), qry(x, M + 1, R, rid));
29 }
30 int main() {
31     int n;
32     for(cin >> n; n--) {
33         string s;
34         cin >> s;
35         if(s[0] == 'Q') {
36             int x;
37             cin >> x;
38             cout << max(0, ((int)(qry(x) * 100)) / 10000)
39                 << "\n";
40         } else {
41             double s, p;
42             cin >> s >> p;
43             ins(Line(p, s));
44         }
45     }
46 }

```

10. AnotherVersionMath

10.1. CRT(luoguVersion).cpp

```

1 long long CRT(long long *W, long long *B,
2                 long long k /* 方程组数 */){
3     long long x, y, a = 0, m, n = 1;
4     for(long long i = 0; i < k; i++) n *= W[i];
5     for(long long i = 0; i < k; i++) {
6         m = n / W[i];
7         ext_gcd(W[i], m, x, y);
8         a = (a + y * m * B[i]) % n;
9     }
10    return a > 0 ? a : a + n;
11 }

```

10.2. 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     /* 輸入 x,n,p,ans 會將 ans 修改為 x^n%p
17      對整數/矩陣/不要求精度的浮點 皆有效
18      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,
20                   450775, 9780504, 1795265022}; /*2^64*/
21     // int p[3]={2,7,61};/*2^32*/
22     bool check(const uLL x, const uLL p) {
23         uLL d = x - 1, ans = 1;
24         fastpow(p, d, x, ans);
25         if(ans != 1) return 1;
26         for(; !(d & 1); ) {
27             d >>= 1;
28             ans = 1;
29             fastpow(p, d, x, ans);
30             if(ans == x - 1)
31                 return 0;
32             else if(ans != 1)
33                 return 1;
34         }
35     }
36     bool miller_rabin(const uLL x) {
37         if(x == 1) return 0;
38         for(auto e : pri) {
39             if(e >= x) return 1;
40             if(check(x, e)) return 0;
41         }
42         return 1;
43     }
44     template <class T> T gcd(T a, T b) {
45         if(!a) return b;
46         if(!b) return a;
47         if(a & b & 1) return gcd(sub(a, b), min(a, b));
48         if(a & 1) return gcd(a, b >> 1);
49         if(b & 1) return gcd(a >> 1, b);
50         return gcd(a >> 1, b >> 1) << 1;
51     }
52     /*gcd(a,b) 默認 gcd(a,0)=a*/
53     mt19937 rnd(time(0));
54     template <class T> T f(T x, T c, T mod) {
55         return (((uLL)x) * x % mod + c) % mod;
56     }
57     template <class T> T rho(T n) {
58         T mod = n, x = rnd() % mod, c = rnd() % (mod - 1) + 1,
59         p = 1;
60         for(T i = 2, j = 2, d = x;; ++i) {
61             x = f(x, c, mod), p = ((uLL)p) * sub(x, d) % mod;
62             if(i % 127 == 0 && gcd(p, n) != 1) return gcd(p, n);
63             if(i == j) {
64                 j <= 1, d = x;
65                 if(gcd(p, n) != 1) return gcd(p, n);
66             }
67         }
68     }
69 }
70 template <class T> T pollard_rho(T n) {
71     if(miller_rabin(n)) return n;
72     T p = n;
73     while(p == n) p = rho(n);
74     return max(pollard_rho(p), pollard_rho(n / p));
75 }
76 int main() {
77     LL t, n, ans;
78     for(cin >> t; t--;) {
79         cin >> n;
80         ans = pollard_rho(n);
81         if(ans == n)
82             puts("Prime");
83         else
84             printf("%lld\n", ans);
85     }
86 }

```

10.3. 快速幂.cpp

```

1 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  /* 輸入 x,n,p,ans 會將 ans 修改為 x^n%p
13 對整數/矩陣/不要求精度的浮點 皆有效
14 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 & 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 :=  $\frac{n}{p^{v(n)}}$ ,
40 s(n) := p進制下n的所有位數和
41 v(n!) =  $\sum_{i=1}^{\infty} \lfloor \frac{n}{p^i} \rfloor$  (勒壤得定理)
42 v( $\binom{n}{m}$ ) =  $\frac{s(n)+s(m-n)-s(m)}{p-1}$  (庫默爾定理)
43 v( $\binom{n}{m_1, m_2, \dots, m_k}$ ) =
44  $\sum_{i=1}^k s(m_i) - s(n)$  (庫默爾定理推廣)
45 \[
46   (n!)_{\text{p}} \equiv a + b \equiv a^p + b^p \pmod p \quad (\text{小費馬})
47   (p-1)! \equiv -1 \pmod p \quad (\text{威爾遜定理})
48   v(n) := n中p的幕次, (n)_p :=  $\frac{n}{p^{v(n)}}$ , s(n) := p進制下n的所有位數和
49   v(n!) =  $\sum_{i=1}^{\infty} \lfloor \frac{n}{p^i} \rfloor$  (勒壤得定理)
50   v( $\binom{n}{m}$ ) =  $\frac{s(n)+s(m-n)-s(m)}{p-1}$  (庫默爾定理)
51   v( $\binom{n}{m_1, m_2, \dots, m_k}$ ) =
52    $\sum_{i=1}^k s(m_i) - s(n)$  (庫默爾定理推廣)
53 \]
54   \[
55     (n!)_{\text{p}} \equiv a + b \equiv a^p + b^p \pmod p \quad (\text{小費馬})
56     (p-1)! \equiv -1 \pmod p \quad (\text{威爾遜定理})
57     v(n) := n中p的幕次, (n)_p :=  $\frac{n}{p^{v(n)}}$ , s(n) := p進制下n的所有位數和
58     v(n!) =  $\sum_{i=1}^{\infty} \lfloor \frac{n}{p^i} \rfloor$  (勒壤得定理)
59     v( $\binom{n}{m}$ ) =  $\frac{s(n)+s(m-n)-s(m)}{p-1}$  (庫默爾定理)
60     v( $\binom{n}{m_1, m_2, \dots, m_k}$ ) =
61   \]
62   \[
63     (n!)_{\text{p}} \equiv a + b \equiv a^p + b^p \pmod p \quad (\text{小費馬})
64     (p-1)! \equiv -1 \pmod p \quad (\text{威爾遜定理})
65     v(n) := n中p的幕次, (n)_p :=  $\frac{n}{p^{v(n)}}$ , s(n) := p進制下n的所有位數和
66     v(n!) =  $\sum_{i=1}^{\infty} \lfloor \frac{n}{p^i} \rfloor$  (勒壤得定理)
67     v( $\binom{n}{m}$ ) =  $\frac{s(n)+s(m-n)-s(m)}{p-1}$  (庫默爾定理)
68     v( $\binom{n}{m_1, m_2, \dots, m_k}$ ) =
69   \]
70   \[
71     (n!)_{\text{p}} \equiv a + b \equiv a^p + b^p \pmod p \quad (\text{小費馬})
72     (p-1)! \equiv -1 \pmod p \quad (\text{威爾遜定理})
73     v(n) := n中p的幕次, (n)_p :=  $\frac{n}{p^{v(n)}}$ , s(n) := p進制下n的所有位數和
74     v(n!) =  $\sum_{i=1}^{\infty} \lfloor \frac{n}{p^i} \rfloor$  (勒壤得定理)
75     v( $\binom{n}{m}$ ) =  $\frac{s(n)+s(m-n)-s(m)}{p-1}$  (庫默爾定理)
76     v( $\binom{n}{m_1, m_2, \dots, m_k}$ ) =
77   \]

```

(2ⁿ)
同時 n 對括號的合法放置數即是 C(n) 若有任意 k 種括號可選 則 C(n)kⁿ

模逆元表 p=i*(p/i)+p%i,-p%i=i*(p/i),inv(i)=-(p/i)*inv(p%i)/
LL frACP[N], invP[N];

```

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

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
17  private:
18    bitset<N> notprime; // 1e8<2^27=128MB
19    void eular(T n) {
20      for(T i = 2; i <= n; ++i) {
21        if(notprime[i]) list.emplace_back(i);
22        const T k = n / i;
23        for(T j : list) {
24          if(j > k) break;
25          notprime[i * j] = 1;
26          if(!(i % j)) break;
27        }
28      }
29    }
30    void eratosthenes(T n) {
31      for(T i = 2; i <= n; ++i) {
32        if(!notprime[i]) list.emplace_back(i);
33        const T k = n / i;
34        for(T j : list) {
35          if(j > k) break;
36          notprime[i * j] = 1;
37          if(!(i % j)) break;
38        }
39      }
40    }
41    void sqrt_sieve(T n) {
42      for(T i = 2; i <= n; ++i) {
43        bool isprime = 1;
44        for(T j : list) {
45          if(j > i / j) break;
46          if(!(i % j)) {
47            isprime = 0;
48            break;
49          }
50        }
51        if(isprime) list.emplace_back(i);
52      }
53    }
54    void factorize(T n) {
55      factor = vector<T>(n);
56      if(list.empty()) eular(n);
57      for(T j : list) factor[j] = j;
58      for(T i = 2; i <= n; ++i) {
59        const T k = n / i;
60        for(T j : list) {
61          if(j > k) break;
62          factor[i] *= j;
63        }
64      }
65    }
66  }
```

```

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

```

11. AnotherVersionString

11.1. KMP (2).cpp

```

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

```

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 /*Kmp kmp(t) 會建好 t 的失配函數 fail[]
 * match 會把每格匹配完的失配函數 p[] 建好 */
25

```

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        }
25    }
26    for(T j : list) {
27        if(j > k) break;
28        factor[i * j] = j;
29        if(!(i % j)) break;
30    }
31 } #undef N
32 /*Prime prime(n) 建立打好 1~n 質數表的物件
prime.list(一個 vector) 是質數表
可修改 define N 決定歐篩/埃篩上限
可在建構子選擇篩法 有歐篩/埃篩/根號暴力搜
prime.factorize(n) 用歐篩方式得到 1~n 所有數的最小質因數
可在 factor(一個 vector) 上一路回溯 logn 得到一個數的質因數分解
做 n 個數質因數分解共花 nlogn
show() 會以空格隔開 顯示所有 list 內的元素 有尾空格尾換行
printf 裡面用%lld 視情況換為%d 或 cout*/

```

```

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

```

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[i] >= 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     }
23     q.push({0, 1});
24     d[1] = 0;
25     for(u = 2; u <= n; ++u) d[u] = INF;
26     for(; !q.empty(); q.pop()) {
27         if(vis[q.top().S]) continue;
28         vis[q.top().S] = 1;
29         for(auto e : adj[q.top().S]) {
30             if(!vis[e.F] && q.top().F + e.S < d[e.F]) {
31                 d[e.F] = q.top().F + e.S;
32                 q.push({d[e.F], e.F});
33             }
34         }
35     }
36 }

```

```

33     }
34 }
35 for(u = 1; u <= n; ++u) printf("%lld ", d[u]);
}

```

12.2. SCC.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];
7 stack<int> st;
8 int dfn[N], low[N], tag, scc[N], sc;
9 bitset<N> in;
10 void dfs(int now, int par = -1) {
11     st.push(now);
12     in[now] = 1;
13     low[now] = dfn[now] = ++tag;
14     for(int e : adj[now]) {
15         if(e == par) continue;
16         if(!dfn[e])
17             dfs(e, now), low[now] = min(low[now], low[e]);
18         else if(in[e])
19             low[now] = min(low[now], dfn[e]);
20     }
21     if(dfn[now] == low[now]) {
22         ++sc;
23         for(; st.top() != now; st.pop())
24             scc[st.top()] = sc, in[st.top()] = 0;
25         st.pop();
26         scc[now] = sc;
27         in[now] = 0;
28         schead[sc] = now;
29     }
30 }
31 int main() {
32     int n, m, u, v;
33     cin >> n >> m;
34     vector<pii> g(m);
35     for(auto &[u, v] : g)
36         cin >> u >> v, adj[u].pb(v), adj[v].pb(u);
37     for(u = 1; u <= n; ++u)
38         if(!dfn[u]) dfs(u);
39     int ans = 0;
40     for(auto &[u, v] : g)
41         if(scc[u] != scc[v]) ++ans; //=eBCC
42     cout << ans << "\n";
43     for(auto &[u, v] : g)
44         if(scc[u] != scc[v]) cout << u << " " << v << "\n";
45 }

```

12.3. cses 有向圖基環樹森林.cpp

```

1 // cses Planets Queries II 基環樹森林模板
2 #include <bits/stdc++.h>
3 using namespace std;
4 #define N 200005
5 #define pb push_back
6 // int cyc[i]=1~n 代表 i 屬於哪顆樹
7 // bitset incyc[i]=0/1 代表 i 是否在環上
8 // int len[k]=1~n 代表第 k 裸樹的環長度
9 // int num[i]=1~n 如果 incyc[i] 代表的是在環上的編號
10 // 否則代表的是環上最近的點的編號 int dis[i]=0~n-1
11 // 代表到環上最近點的距離 若 i 在環上則為 0
12 int tag = 1, cyc[N], len[N], num[N], dis[N], nxt[N][19];
13 bitset<N> vis, incyc;
14 vector<int> path;
15 void dfs(int now) {
16     if(vis[now]) {
17         int i = 1;
18         for(int k; k = path.back(), path.pop_back(),
19              k != now && !path.empty());
20         ++i;
21         cyc[k] = tag;
22         incyc[k] = 1;
23         num[k] = i;
24     }
25     cyc[now] = tag;
26     incyc[now] = 1;
27     num[now] = i;
28     len[tag] = i;
29     ++tag;
30     return;
31 }
32 vis[now] = 1;
33 path.pb(now);
34 if(!cyc[nxt[now][0]]) dfs(nxt[now][0]);
35 if(cyc[now]) return;
36 cyc[now] = cyc[nxt[now][0]];
37 num[now] = num[nxt[now][0]];

```

```

39     dis[now] = dis[nxt[now][0]] + 1;
40 }
41 int jmp(int a, int x) {
42     for(int k = 19; k--;) {
43         for(; 1 << k <= x;) x -= 1 << k, a = nxt[a][k];
44     }
45 }
46 int main() {
47     ios::sync_with_stdio(0);
48     cin.tie(0);
49     cout.tie(0);
50     int n, q, i = 1, u, v;
51     for(cin >> n >> q; i <= n; ++i) cin >> nxt[i][0];
52     for(int k = 1; k < 19; ++k)
53         for(i = 1; i <= n; ++i)
54             nxt[i][k] = nxt[nxt[i][k - 1]][k - 1];
55     for(i = 1; i <= n; ++i)
56         if(!cyc[i]) path.clear(), dfs(i);
57     for(; q--;) {
58         cin >> u >> v;
59         if(cyc[u] == cyc[v]) {
60             if(incyc[v])
61                 cout << (!incyc[u] ? dis[u] : 0) +
62                     (num[u] - num[v] + len[cyc[u]]) %
63                     len[cyc[u]] << "\n";
64         } else if(num[u] == num[v] && dis[u] >= dis[v] &&
65                    jmp(u, dis[u] - dis[v]) == v)
66             cout << dis[u] - dis[v] << "\n";
67         else
68             cout << "-1\n";
69     } else
70         cout << "-1\n";
71 }

```

13. AnotherVersionGeometry

13.1. DynamicHull.cpp

```

1 struct Line {
2     mutable int a, b, r;
3     bool operator<(const Line &o) const { return a < o.a; }
4     bool operator<(const int o) const { return r < o; }
5 };
6
7 struct DynamicHull : multiset<Line, less<> {
8     inline int Div(int a, int b) {
9         return a / b - ((a ^ b) < 0 && a % b);
10    }
11    inline bool intersect(iterator x, iterator y) {
12        if(y == end()) {
13            x->r = inf;
14            return false;
15        }
16        if(x->a == y->a)
17            x->r = (x->b) > (y->b) ? inf : -inf;
18        else
19            x->r = Div((y->b) - (x->b), (x->a) - (y->a));
20        return (x->r) >= (y->r);
21    }
22    void Insert(int a, int b) {
23        auto y = insert({a, b, 0}), z = next(y), x = y;
24        while(intersect(y, z)) z = erase(z);
25        if(x != begin() && intersect(--x, y))
26            intersect(x, y = erase(y));
27        while((y = x) != begin() && ((--x)->r) >= (y->r))
28            intersect(x, erase(y));
29    }
30    int query(int x) const {
31        auto l = *lower_bound(x);
32        return (l.a) * x + (l.b);
33    }

```

14. AnotherVersionTree

14.1. LCA.cpp

```

1 #define N 100005
2 #define LG 15
3 int dep[N], par[N][LG], sub[N];
4 vector<int> g[N];
5 void dfs(int now = 1, int pre = 0) {
6     dep[now] = dep[pre] + 1;
7     par[now][0] = pre;
8     sub[now] = 1;
9     for(int e : g[now])
10         if(e != pre) dfs(e, now), sub[now] += sub[e];
11 }

```

```

13 int jmp(int x, int k) {
14     for(int i = LG; i--;) {
15         for(; k >= 1 << i; k -= 1 << i) x = par[x][i];
16     }
17     return x;
18 }
19 int lca(int a, int b) {
20     if(dep[a] > dep[b]) swap(a, b);
21     b = jmp(b, dep[b] - dep[a]);
22     if(a == b) return a;
23     for(int i = LG; i--;) {
24         for(; par[a][i] != par[b][i]; b = par[b][i])
25             a = par[a][i];
26     }
27     return par[a][0];
28 }
29 int main() {
30     int n;
31     cin >> n;
32     for(int i = n, u, v; --i;) {
33         cin >> u >> v, g[u].pb(v), g[v].pb(u);
34     }
35     dfs();
36     for(int i = 1; i < LG; ++i)
37         for(int j = 1; j <= n; ++j)
38             par[j][i] = par[par[j][i - 1]][i - 1];
39     int k = lca(1, n);
40
41 // 點編號 1~n，建的無向圖但改 dfs
42 // 就能變有向，改有向記得邊要反著建 dep[n] 代表 n 的深度 (1
43 // base)，par[i][j] 代表 i 往上 1<<j 步的祖先是誰，不存在則是
44 // 0，sub[i] 代表 i 的子樹大小 jmp(i,j) 代表 i 往上 j
45 // 步的祖先是誰
46
47 #pragma GCC optimize(
48     "Ofast,fast-math,unroll-loops,no-stack-protector")
49 #include <bits/stdc++.h>
50 using namespace std;
51 #define ll long long
52 #define pb push_back
53 #define N 200005
54 #define pii pair<int, int>
55 #define ppi pair<pii, pii>
56 char buf[1 << 22], *p1, *p2;
57 int p[12];
58 #define gc()
59     (p1 == p2 &&
60      (p2 = (p1 = buf) + fread(buf, 1, 1 << 22, stdin)),
61      p1 == p2)
62     ? EOF
63     : *p1++)
64 inline int gi() {
65     int x = 0;
66     for(char c; '0' <= (c = gc()) && c <= '9'; x += c - '0')
67         x *= 10;
68     return x;
69 }
70 inline void pi(int x, char c = ' ') {
71     if(!x) putchar('0');
72     int i = 0;
73     for(; x; x /= 10) p[i++] = x % 10;
74     for(; i--;) putchar(p[i] + '0');
75     putchar(c);
76 }
77 int main() {
78     cin.tie(0)->sync_with_stdio(0);
79     int n, m, q;
80     cin >> n >> m >> q;
81     vector<ppp> g(m);
82     bitset<M> ans;
83     vector<vector<pii>> adj(n + 1, vector<pii>());
84     for(int i = 0; i < m; ++i) {
85         auto &[p1, p2] = g[i];
86         auto &[w, idx] = p1;
87         auto &[u, v] = p2;
88         cin >> u >> v >> w;
89         idx = i;
90     }
91     sort(g.begin(), g.end());
92     vector<ll> dsu(n + 1, -1);
93     auto qry = [&dsu](auto qry, int x) -> int {
94         return dsu[x] < 0 ? x : dsu[x] = qry(qry, dsu[x]);
95     };
96     auto upd = [&dsu, &qry](int u, int v) -> void {
97         if(dsu[u = qry(qry, u)] > dsu[v = qry(qry, v)])
98             swap(u, v);
99         dsu[u] += dsu[v];
100        dsu[v] = u;
101    };
102    for(auto &[p1, p2] : g) {
103        auto &[w, idx] = p1;
104        auto &[u, v] = p2;
105        if(qry(qry, u) != qry(qry, v))
106            upd(u, v), adj[u].pb({v, w}), adj[v].pb({u, w});
107    }
108 }
109 vector<vector<int>> par(n + 1, vector<int>(LG));
110 mx(n + 1, vector<int>(LG));
111 vector<int> dep(n + 1);
112 auto dfs = [&par, &mx, &dep, &adj](auto dfs, int now,
113                                         int p = 0,
114                                         int w = 0) -> void {
115     par[now][0] = p;
116     mx[now][0] = w;
117     dep[now] = dep[p] + 1;
118     for(auto &[e, w] : adj[now])
119         if(e != p) dfs(dfs, e, now, w);
120 }
121 dfs(dfs, 1);
122 for(int i = 1; i < LG; ++i)
123     for(int j = 1; j <= n; ++j)
124         par[j][i] = par[par[j][i - 1]][i - 1],
125         mx[j][i] =
126             max(mx[j][i - 1], mx[par[j][i - 1]][i - 1]);
127 auto lca = [&par, &dep](int u, int v) -> int {
128     if(dep[u] > dep[v]) swap(u, v);
129     for(int i = LG; i--;) {
130         if((1 << i) & (dep[v] - dep[u])) v = par[v][i];
131         if(u == v) return u;
132         for(int i = LG; i--;) {
133             if(par[u][i] != par[v][i])
134                 u = par[u][i], v = par[v][i];
135         }
136     }
137     auto path = [&par, &mx, &dep](int k, int x) -> int {
138         int ans = 0;
139         for(int i = LG; i--;) {
140             if((1 << i) & (dep[x] - dep[k]))
141                 ans = max(ans, mx[x][i]), x = par[x][i];
142         }
143     };
144     for(auto &[p1, p2] : g) {
145         auto &[w, idx] = p1;
146         auto &[u, v] = p2;
147         int k = lca(u, v);
148         ans[idx] = max(path(k, u), path(k, v)) >= w;
149     }
150     for(int i = 0; i < m; ++i)
151         cout << i << " "
152         << (const char[2][5]){"NO\n", "YES\n"}[ans[i]];
153     cout << "\n";
154     for(int k; q--;) {
155         cin >> k;
156         int flag = 1;
157         for(int x; k--;) {
158             cin >> x;
159             if(!ans[x - 1]) flag = 0;
160         }
161         cout << (const char[2][5]){"NO\n", "YES\n"}[flag];
162     }
163 }

```