

Contents

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

1. DataStructure

1.1. 2DBIT.cpp

```

1
2
3 using namespace std;
4 #define LL long long
5 #define pii pair<int, int>
6 #define N 1005
7 #define F first
8 #define S second
9 int bit[N][N];
10 #define lb(x) (x & -x)
11 void upd(int i, int j, int v) {
12     for(int k = i; k < N; k += lb(k)) bit[k][j] += v;
13 }
14 int qry2(int i, int j) {
15     int ans = 0;
16     for(int k = i; k < N; k += lb(k)) ans += bit[k][j];
17     return ans;
18 }
19 int qry(int y1, int x1, int y2, int x2) {
20     return qry2(y2, x2) - qry2(y2, x1 - 1) - qry2(y1 - 1, x2) +
21         qry2(y1 - 1, x1 - 1);
22 }
23
24 int main() {
25     int n, q, i = 1, j, y, x;
26     for(scanf("%d %d", &n, &q); getchar(), i <= n; ++i)
27         for(j = 1; j <= n; ++j)
28             if(getchar() == '*') upd(i, j, 1);
29     for(; q--;) {
30         scanf("%d", &i);
31         if(i == 1)
32             scanf("%d", &i, &j),
33             upd(i, j, 1 - 2 * qry(i, j, i, j));
34         else
35             scanf("%d%d%d", &i, &j, &y, &x),
36             printf("%d\n", qry(i, j, y, x));
37     }
38 }
39

```

1.2. DynamicSegmentTree.cpp

```

1  #define int long long
3  using namespace std;

5  int n, q;
6  struct node {
7      int data, lson, rson, tag;
8      int rv() { return data + tag; }
9  };

11 node tree[20000005];
12 int a[2000005];
13 int now = 1;
14 int mx = 1000000005;

15 void push(int index) {
16     if(!tree[index].lson) {
17         tree[index].lson = ++now;
18     }
19     if(!tree[index].rson) {
20         tree[index].rson = ++now;
21     }
22     int lson = tree[index].lson;
23     int rson = tree[index].rson;
24     tree[lson].tag += tree[index].tag;
25     tree[rson].tag += tree[index].tag;
26     tree[index].data = tree[index].rv();
27     tree[index].tag = 0;
28 }

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 }

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 signed main() {
70     ios::sync_with_stdio(0);
71     cin.tie(0);
72     cout.tie(0);
73     cin >> n >> q;
74     for(int i = 1; i <= n; i++) {
75         cin >> a[i];
76         modify(1, mx, a[i], a[i], 1, 1);
77     }
78     while(q--) {
79         char mode;
80         int x, y;
81         cin >> mode;
82         if(mode == '?') {
83             cin >> x >> y;
84             cout << query(1, mx, x, y, 1) << "\n";
85         } else {
86             cin >> x >> y;
87             modify(1, mx, a[x], a[x], -1, 1);
88             a[x] = y;
89             modify(1, mx, a[x], a[x], 1, 1);
90         }
91     }

```

```

93 }

```

1.3. PbdsGpHashTable.cpp

```

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

```

1.4. PbdsPriorityQueue.cpp

```

1  __gnu_pbds::priority_queue<int> pq;
3  /*
4  push(x); //return iterator
5  pop() top() join(pq2) erase(iter) modify(iter,x)
6  */

```

1.5. PbdsRope.cpp

```

1  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  using namespace __gnu_pbds;
3  /*
4  tree<int,null_type,less<int>,rb_tree_tag,
5  tree_order_statistics_node_update> tr;
6  //same as rope<int>, except tr.lower_bound(x) and upper_bound
7  tr.find_by_order(k); //return kth iterator; k=[0,tr.size())
8  //out of this will get tr.end()
9  tr.order_of_key(val); //return rank(val);
10 tr.join(tr2); //merge tr
11 and tr2, tr2.clear() tr.split(const int&r, RBTree&tr2); //<r
12 will in tr, >=r will in tr2
13 */

```

1.7. PersistentSegmentTree.cpp

```

1  // cses Range Queries and Copies
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;

```

```

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

```

1.8. Treap.cpp

```

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

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

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

pii split(int rk, int index) {
    if(!index) return {0, 0};
    push(index);
    int lson = treap[index].lson;

```

```

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

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

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

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

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

```

2. Math

2.1. CRT.cpp

```

#define int long long
using namespace std;

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

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

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

int ans = 0;

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

```

```

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

```

2.2. CountPrimes.cpp

```

1 using namespace std;
2 using i64 = long long;
3 i64 count_pi(i64 N) {
4     if(N <= 1) return 0;
5     int v = sqrt(N + 0.5);
6     int n_4 = sqrt(v + 0.5);
7     int T = min((int)sqrt(n_4) * 2, n_4);
8     int K = pow(N, 0.625) / log(N) * 2;
9     K = max(K, v);
10    K = min<i64>(K, N);
11    int B = N / K;
12    B = N / (N / B);
13    B = min<i64>(N / (N / B), K);
14
15    vector<i64> l(v + 1);
16    vector<int> s(K + 1);
17    vector<bool> e(K + 1);
18    vector<int> w(K + 1);
19    for(int i = 1; i <= v; ++i) l[i] = N / i - 1;
20    for(int i = 1; i <= v; ++i) s[i] = i - 1;
21
22    const auto div = [](i64 n, int d) -> int {
23        return double(n) / d;
24    };
25    int p;
26    for(p = 2; p <= T; ++p)
27        if(s[p] != s[p - 1]) {
28            i64 M = N / p;
29            int t = v / p, t0 = s[p - 1];
30            for(int i = 1; i <= t; ++i) l[i] -= l[i * p] - t0;
31            for(int i = t + 1; i <= v; ++i)
32                l[i] -= s[div(M, i)] - t0;
33            for(int i = v, j = t; j >= p; --j)
34                for(int l = j * p; i >= l; --i)
35                    s[i] -= s[j] - t0;
36            for(int i = p * p; i <= K; i += p) e[i] = 1;
37        }
38    e[1] = 1;
39    int cnt = 1;
40    vector<int> roughs(B + 1);
41    for(int i = 1; i <= B; ++i)
42        if(!e[i]) roughs[cnt++] = i;
43    roughs[cnt] = 0x7fffffff;
44    for(int i = 1; i <= K; ++i) w[i] = e[i] + w[i - 1];
45    for(int i = 1; i <= K; ++i) s[i] = w[i] - w[i - (i & -i)];
46
47    const auto query = [&](int x) -> int {
48        int sum = x;
49        while(x) sum -= s[x], x ^= x & -x;
50        return sum;
51    };
52    const auto add = [&](int x) -> void {
53        e[x] = 1;
54        while(x <= K) ++s[x], x += x & -x;
55    };
56    cnt = 1;
57    for(p = 2; p <= n_4; ++p)
58        if(!e[p]) {
59            i64 q = i64(p) * p, M = N / p;
60            while(cnt < q) w[cnt] = query(cnt), cnt++;
61            int t1 = B / p, t2 = min<i64>(B, M / q),
62                t0 = query(p - 1);
63            int id = 1, i = 1;
64            for(; i <= t1; i = roughs[++id])
65                l[i] -= l[i * p] - t0;
66            for(; i <= t2; i = roughs[++id])
67                l[i] -= query(div(M, i)) - t0;
68            for(; i <= B; i = roughs[++id])
69                l[i] -= w[div(M, i)] - t0;
70            for(int i = q; i <= K; i += p)
71                if(!e[i]) add(i);
72        }
73    while(cnt <= v) w[cnt] = query(cnt), cnt++;
74
75    vector<int> primes;
76    primes.push_back(1);
77    for(int i = 2; i <= v; ++i)
78        if(!e[i]) primes.push_back(i);
79    l[1] += i64(w[v] + w[n_4] - 1) * (w[v] - w[n_4]) / 2;

```

```

81    for(int i = w[n_4] + 1; i <= w[B]; ++i)
82        l[1] -= l[primes[i]];
83    for(int i = w[B] + 1; i <= w[v]; ++i)
84        l[1] -= query(N / primes[i]);
85    for(int i = w[n_4] + 1; i <= w[v]; ++i) {
86        int q = primes[i];
87        i64 M = N / q;
88        int e = w[M / q];
89        if(e <= i) break;
90        l[1] += e - i;
91        i64 t = 0;
92        int m = w[sqrt(M + 0.5)];
93        for(int k = i + 1; k <= m; ++k)
94            t += w[div(M, primes[k])];
95        l[1] += 2 * t - (i + m) * (m - i);
96    }
97    return l[1];
98 }

```

2.3. FFT.cpp

```

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

```

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    }
14 }

```

2.5. Formula.tex

2.5.1. Dirichlet Convolution

$$\varepsilon = \mu * 1$$

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

2.5.2. Burnside's Lemma

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

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

Then

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

2.5.3. Pick Theorem

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

2.5.4. Fermat's Little Theorem

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

2.5.5. Wilson's Theorem

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

2.5.6. Legendre Theorem

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

$$\binom{n}{p} := \frac{n}{p \binom{n-1}{p-1}}$$

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

$$v(n!) = \sum_{i=1}^{\infty} \left\lfloor \frac{n}{p^i} \right\rfloor = \frac{n-s(n)}{p-1}$$

2.5.7. Kummer Theorem

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

2.5.8. ext-Kummer Theorem

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

2.5.9. Factorial with mod

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

2.5.10. Properties of nCr with mod

If any i in base p satisfies $n_i < m_i$, then $\binom{n}{m_i} \% p = 0$. 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 $k \in \text{positive number}$, calculate $\binom{n}{m} \% k$ can decompose k by Fundamental Theorem of Arithmetic. And then use crt.

2.5.12. Catalan Number

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

2.5.13. modinv table

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

2.6. Gaussian-Jordan.cpp

```
1  #define int long long
2  using namespace std;
3
4  int n;
5  double a[105][105];
6
7  // n <= m
8  void gaussian(double a[105][105], int n, int m) {
9      int curi = 0;
10     for(int j = 0; j < m; j++) {
11         int i;
12         for(i = curi; i < n; i++) {
13             if(a[i][j]) {
14                 break;
15             }
16         }
17         if(a[i][j] == 0) continue;
18         for(int k = 0; k < m; k++) {
19             swap(a[i][k], a[curi][k]);
20         }
21     }
```

```
23     for(int k = m - 1; k >= j; k--) {
24         a[curi][k] /= a[curi][j];
25     }
26     for(int i = 0; i < n; ++i) {
27         if(i != curi) {
28             for(int k = m - 1; k >= j; k--) {
29                 a[i][k] -= a[curi][k] * a[i][j];
30             }
31         }
32         curi++;
33     }
34 }
```

2.7. Generator.cpp

```
1  #define int long long
2  using namespace std;
3
4  int t;
5  int n, d;
6  bitset<1000005> exist;
7  bitset<1000005> vis;
8  vector<int> prime;
9  int phi[1000005];
10
11 void init() {
12     phi[1] = 1;
13     for(int i = 2; i <= 1000000; i++) {
14         if(!vis[i]) {
15             prime.push_back(i);
16             phi[i] = i - 1;
17         }
18         for(int j : prime) {
19             if(i * j > 1000000) break;
20             vis[i * j] = 1;
21             if(i % j == 0) {
22                 phi[i * j] = phi[i] * j;
23                 break;
24             } else {
25                 phi[i * j] = phi[i] * phi[j];
26             }
27         }
28     }
29     exist[2] = exist[4] = 1;
30     for(int i : prime) {
31         if(i == 2) continue;
32         for(int j = i; j <= 1000000; j += i) {
33             exist[j] = 1;
34             if(j * 2 <= 1000000) {
35                 exist[j * 2] = 1;
36             }
37         }
38     }
39 }
40
41 vector<int> factors(int x) {
42     vector<int> v;
43     for(int i = 1; i * i <= x; i++) {
44         if(x % i == 0) {
45             v.push_back(i);
46             if(i * i != x) {
47                 v.push_back(x / i);
48             }
49         }
50     }
51     return v;
52 }
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
68 vector<int> findroot(int x) {
69     vector<int> ret;
70     if(!exist[x]) return ret;
71     int phix = phi[x];
72     vector<int> fact = factors(phix);
73     int fst;
74     for(int i = 1; i <= fact.size(); i++) {
75         if(__gcd(i, x) != 1) continue;
76         bool ok = 1;
77     }
```



```

79     for(int j : fact) {
80         if(j != phix && f(i, j, x) == 1) {
81             ok = 0;
82             break;
83         }
84     }
85     if(ok) {
86         fst = i;
87         break;
88     }
89     int now = fst;
90     // cout << fst << "\n";
91     for(int i = 1; i <= phix; i++) {
92         if(__gcd(i, phix) == 1) {
93             ret.push_back(now);
94         }
95         now *= fst;
96         now %= x;
97     }
98     return ret;
99 }

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

```

2.8. Inv.cpp

```

1 int exgcd(int a, int b, int &x, int &y) {
2     if(b == 0) {
3         x = 1;
4         y = 0;
5         return a;
6     }
7     int d = exgcd(b, a % b, y, x);
8     y -= x * (a / b);
9     return d;
10 }
11
12 int inv(int a, int p) {
13     int x, y;
14     exgcd(a, p, x, y);
15     return (x % p + p) % p;
16 }

```

2.9. Lucas.cpp

```

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

```

2.10. MillerRabin.cpp

```

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

```

2.11. Mu.cpp

```

1 vector<int> prime;
2 bitset<1000005> vis;
3 int n;
4 int mu[1000005];
5
6 void init() {
7     for(int i = 2; i <= n; i++) {
8         if(!vis[i]) {
9             prime.push_back(i);
10            mu[i] = -1;
11        }
12        for(int p : prime) {
13            if(i * p > n) break;
14            vis[i * p] = 1;
15            if(i % p == 0) {
16                mu[i * p] = 0;
17                break;
18            } else {
19                mu[i * p] = mu[i] * mu[p];
20            }
21        }
22    }
23 }

```

2.12. NTT.cpp

```

1 #define ll long long
2 using namespace std;
3
4 const int MAXN = 1000005;
5 const int MOD = 998244353, G = 3;
6 int rev[MAXN * 3];
7
8 int qpow(int x, int y) {
9     int ret = 1;
10    while(y) {
11        if(y & 1) {
12            ret *= x;
13            ret %= MOD;
14        }
15        x *= x;
16        x %= MOD;
17        y >>= 1;
18    }
19    return ret;
20 }

```

```

23 void ntt(int F[], int N, int sgn) {
24     int bit = __lg(N);
25     for(int i = 0; i < N; ++i) {
26         rev[i] = (rev[i] >> 1) >> 1 | ((i & 1) << (bit - 1));
27         if(i < rev[i]) swap(F[i], F[rev[i]]);
28     }
29     for(int l = 1, t = 1; l < N; l <= 1, t++) {
30         int step = qpow(G, ((MOD - 1) >> t) * sgn + MOD - 1);
31         for(int i = 0; i < N; i += l << 1) {
32             for(int k = i, cur = 1; k < i + l; ++k) {
33                 int g = F[k], h = (ll)F[k + l] * cur % MOD;
34                 F[k] = (g + h) % MOD;
35                 F[k + l] = ((g - h) % MOD + MOD) % MOD;
36                 cur = (ll)cur * step % MOD;
37             }
38         }
39     }
40     if(sgn == -1) {
41         int invN = qpow(N, MOD - 2);
42         for(int i = 0; i < N; ++i) F[i] = (ll)F[i] * invN % MOD;
43     }
44 }

```

2.13. PollardRho.cpp

```

1 using namespace std;
2 #define LL long long
3 #define uLL __uint128_t
4 #define sub(a, b) ((a) < (b) ? (b) - (a) : (a) - (b))
5 template <class T, class POW>
6 void fastpow(T x, POW n, POW p, T &ans) {
7     for(; n; n >>= 1) {
8         if(n & 1) {
9             ans *= x;
10            ans %= p;
11        }
12        x *= x;
13        x %= p;
14    }
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 __uint128)
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     return 0;
36 }
37 bool miller_rabin(const uLL x) {
38     if(x == 1) return 0;
39     for(auto e : pri) {
40         if(e >= x) return 1;
41         if(check(x, e)) return 0;
42     }
43     return 1;
44 }
45 template <class T> T gcd(T a, T b) {
46     if(!a) return b;
47     if(!b) return a;
48     if(a & b & 1) return gcd(sub(a, b), min(a, b));
49     if(a & 1) return gcd(a, b >> 1);
50     if(b & 1) return gcd(a >> 1, b);
51     return gcd(a >> 1, b >> 1) << 1;
52 }
53 /*gcd(a,b) denote gcd(a, 0) = a*/
54 mt19937 rnd(time(0));
55 template <class T> T f(T x, T c, T mod) {
56     return (((uLL)x) * x % mod + c) % mod;
57 }
58 template <class T> T rho(T n) {
59     T mod = n, x = rnd() % mod, c = rnd() % (mod - 1) + 1,
60     p = 1;
61     for(T i = 2, j = 2, d = x; ++i) {
62         x = f(x, c, mod);
63         p = ((uLL)p) * sub(x, d) % mod;
64         if(i % 127 == 0 && gcd(p, n) != 1) return gcd(p, n);
65         if(i == j) {
66             j <= 1, d = x;
67             if(gcd(p, n) != 1) return gcd(p, n);
68         }
69     }
70 }

```

```

69     }
70 }
71 template <class T> T pollard_rho(T n) {
72     if(miller_rabin(n)) return n;
73     T p = n;
74     while(p == n) p = rho(n);
75     return max(pollard_rho(p), pollard_rho(n / p));
76 }
77 int main() {
78     LL t, n, ans;
79     for(cin >> t; t--;) {
80         cin >> n;
81         ans = pollard_rho(n);
82         if(ans == n)
83             puts("Prime");
84         else
85             printf("%lld\n", ans);
86     }
87 }

```

2.14. XorBasis.cpp

```

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

```

2.15. mtt.cpp

```

1 using namespace std;
2 // https://www.luogu.com.cn/article/08nmgxd1
3 namespace poly {
4     long double const pi = acos(-1);
5     struct comp {
6         long double r, i;
7         comp() { r = i = 0; }
8         comp(long double x, long double y) { r = x, i = y; }
9         comp conj() { return comp(r, -i); }
10        friend comp operator+(comp x, comp y) {
11            return comp(x.r + y.r, x.i + y.i);
12        }
13        friend comp operator-(comp x, comp y) {
14            return comp(x.r - y.r, x.i - y.i);
15        }
16        friend comp operator*(comp x, comp y) {
17            return comp(x.r * y.r - x.i * y.i,
18                x.i * y.r + x.r * y.i);
19        }
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         // ...
32     }
33 }

```

```

31     int q = len >> 1;
32     comp wn = comp(cos(pi / q), op * sin(pi / q));
33     for(int i = 0; i < n; i += len) {
34         comp w = comp(1, 0);
35         for(int j = i; j < i + q; j++, w = w * wn) {
36             comp d = f[j + q] * w;
37             f[j + q] = f[j] - d;
38             f[j] = f[j] + d;
39         }
40     }
41 }
42
43 void mtt(int *f, int *g, int *h, int n, int p) {
44     for(int i = 0; i < n; i++) {
45         a[i].r = (f[i] >> 15);
46         a[i].i = (f[i] & 32767);
47         c[i].r = (g[i] >> 15);
48         c[i].i = (g[i] & 32767);
49     }
50     fft(a, n, 1), fft(c, n, 1);
51     for(int i = 1; i < n; i++) b[i] = a[n - i].conj();
52     b[0] = a[0].conj();
53     for(int i = 1; i < n; i++) d[i] = c[n - i].conj();
54     d[0] = c[0].conj();
55     for(int i = 0; i < n; i++) {
56         comp aa = (a[i] + b[i]) * comp(0.5, 0);
57         comp bb = (a[i] - b[i]) * comp(0, -0.5);
58         comp cc = (c[i] + d[i]) * comp(0.5, 0);
59         comp dd = (c[i] - d[i]) * comp(0, -0.5);
60         a[i] = aa * cc + comp(0, 1) * (aa * dd + bb * cc);
61         b[i] = bb * dd;
62     }
63     fft(a, n, -1), fft(b, n, -1);
64     for(int i = 0; i < n; i++) {
65         int aa = (ll)(a[i].r / n + 0.5) % p,
66             bb = (ll)(a[i].i / n + 0.5) % p,
67             cc = (ll)(b[i].r / n + 0.5) % p;
68         h[i] = ((ll) * aa * (1 << 30) + ll * bb * (1 << 15) +
69             cc) %
70             p +
71             p) %
72             p;
73     }
74 } // namespace poly
75 using namespace poly;
76 int f[400005], g[400005], h[400005];
77 // 400005 is 2 * (n + m)
78 int main() {
79     int n, m, p;
80     scanf("%d%d%d", &n, &m, &p);
81     for(int i = 0; i <= n; i++) scanf("%d", &f[i]);
82     for(int i = 0; i <= m; i++) scanf("%d", &g[i]);
83     int lim = 1;
84     while(lim <= (n + m)) lim <<= 1;
85     mtt(f, g, h, lim, p);
86     for(int i = 0; i <= n + m; i++) printf("%d ", h[i]);
87     return 0;
88 }

```

3. String

3.1. Booth.cpp

```

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

```

3.2. KMP.cpp

```

1 string s, t;
2 int pmt[1000005];
3
4 void init() {
5     for(int i = 1, j = 0; i < t.size(); i++) {
6         while(j && t[j] ^ t[i]) {

```

```

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

```

3.3. LongestPalindrome.cpp

```

1 #define int long long
2 using namespace std;
3
4 string s;
5 string t;
6 int n;
7 int d[2000005];
8 int ans = 0;
9
10 signed main() {
11     cin >> t;
12     n = t.size();
13     for(int i = 0; i < 2 * n + 1; i++) {
14         if(i & 1 ^ 1) {
15             s += '0';
16         } else {
17             s += t[i / 2];
18         }
19     }
20     n = s.size();
21     d[0] = 1;
22     for(int i = 0, l = 0, r = 0; i < n; i++) {
23         if(i > r) {
24             d[i] = 1;
25             bool a = i + d[i] < n;
26             bool b = i - d[i] >= 0;
27             bool c = (s[i + d[i]] == s[i - d[i]]);
28             while(a && b && c) {
29                 d[i]++;
30                 a = i + d[i] < n;
31                 b = i - d[i] >= 0;
32                 c = ((i + d[i]) == s[i - d[i]]);
33             }
34             l = i - d[i] + 1;
35             r = i + d[i] - 1;
36         } else {
37             int j = l + r - i;
38             if(j - d[j] + 1 > l) {
39                 d[i] = d[j];
40             } else {
41                 d[i] = r - i + 1;
42                 a = i + d[i] < n;
43                 b = i - d[i] >= 0;
44                 c = (s[i + d[i]] == s[i - d[i]]);
45                 while(a && b && c) {
46                     d[i]++;
47                     a = i + d[i] < n;
48                     b = i - d[i] >= 0;
49                     c = (s[i + d[i]] == s[i - d[i]]);
50                 }
51                 l = i - d[i] + 1;
52                 r = i + d[i] - 1;
53             }
54         }
55         // cout << d[i] << " ";
56         if(d[i] > d[ans]) {
57             ans = i;
58         }
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  #define int long long
3  using namespace std;

5  string s, t;
6  int ans = 0;

7  int z[2000005];

9  signed main() {
11     ios::sync_with_stdio(0);
12     cin.tie(0);
13     cout.tie(0);
14     cin >> s >> t;
15     s = t + '0' + s;
16     int n, m;
17     n = s.size();
18     m = t.size();
19     for(int i = 0, l = 0, r = 0; i < n; i++) {
21         if(z[i - l] < r - i + 1) {
22             z[i] = z[i - l];
23         } else {
24             z[i] = max(r - i + 1, (int)0);
25             while(i + z[i] < n && s[i + z[i]] == s[z[i]]) {
26                 z[i]++;
27             }
28             l = i;
29             r = i + z[i] - 1;
30             if(z[i] == m) {
31                 ans++;
32             }
33         }
34     }
35     cout << ans;
}

```

4. Graph

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

```

1  #define int long long
3  using namespace std;

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];

19 int no(int x) {
20     if(x > m) return x - m;
21     return x + m;
22 }

23 void dfs(int x) {
24     d[x] = low[x] = ++cnt;
25     s.push(x);
26     for(int i : v[x]) {
27         if(scc[i]) continue;
28         if(d[i]) {
29             low[x] = min(low[x], d[i]);
30         } else {
31             dfs(i);
32             low[x] = min(low[x], low[i]);
33         }
34     }
35     if(d[x] == low[x]) {
36         now++;
37         while(!s.empty()) {
38             int k = s.top();
39             s.pop();
40             scc[k] = now;
41             if(k == x) break;
42         }
43     }
44 }

45 }

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;

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>>,
10                  greater<pair<int, int>>>
11         pq;
12     memset(dis, 0x3f, sizeof(dis));
13     vis.reset();
14     dis[x] = 0;
15     pq.push({0, x});
16     while(!pq.empty()) {
17         pair<int, int> now = pq.top();
18         pq.pop();
19         if(vis[now.second]) continue;
20         vis[now.second] = 1;
21         for(auto [i, w] : v[now.second]) {
22             if(vis[i]) continue;
23             if(dis[now.second] + w < dis[i]) {
24                 dis[i] = dis[now.second] + w;
25                 pq.push({dis[i], i});
26             }
27         }
28     }
29 }

```

4.3. Dinic.cpp

```

1 using namespace std;
2 #define ll long long
3 const ll inf = 8e18;
4 #define N 505
5 #define pb push_back
6 struct pp {
7     int from, to;
8     ll flow;
9 };
10 int t, lvl[N], p[N];
11 vector<int> g[N];
12 vector<pp> edge;
13 int bfs(int s) {
14     queue<int> q;
15     for(q.push(s), lvl[s] = 1; !q.empty(); q.pop()) {
16         int u = q.front();
17         for(int e : g[u]) {
18             int v = edge[e].to;
19             if(lvl[v] || !edge[e].flow) continue;
20             lvl[v] = lvl[u] + 1;
21             q.push(v);
22         }
23     }
24     return lvl[t];
25 }
26 ll dfs(int u, ll f = inf) {
27     if(u == t || !f) return f;
28     ll ans = 0;
29     for(int &i = p[u]; i < g[u].size(); ++i) {
30         pp &e = edge[g[u][i]], &b = edge[g[u][i] ^ 1];
31         if(lvl[e.to] == lvl[u] + 1) {
32             ll c = dfs(e.to, min(e.flow, f));
33             e.flow -= c;
34             b.flow += c;
35             f -= c;
36             ans += c;
37         }
38     }
39     return ans;
40 }
41 ll dinic(int s) {
42     ll ans = 0;
43     for(; bfs(s); memset(lvl, 0, sizeof lvl))
44         for(ll k; k = (memset(p, 0, sizeof(p)), dfs(s));)
45             ans += k;
46     return ans;
47 }
48 int main() {
49     ios::sync_with_stdio(0);
50     cin.tie(0);
51     cout.tie(0);
52     int n, m, cnt = 0;
53     for(cin >> n >> m; m--;) {
54         int u, v;
55         ll f;
56         cin >> u >> v >> f;
57         g[u].pb(cnt++);
58         g[v].pb(cnt++);
59         edge.pb({u, v, f});
60         edge.pb({v, u, 0});
61     }
62     t = n;
63     cout << dinic(1);
64 }

```

4.4. MaximumFlow.cpp

```

1 #define int long long
2 using namespace std;
3 int n, m;
4 vector<int> v[1005];
5 int head[1005];
6 int c[1005][1005];
7 int lv[1005];
8 int ans = 0;
9 bool bfs() {
10     memset(head, 0, sizeof(head));
11     memset(lv, 0, sizeof(lv));
12     queue<int> q;
13     q.push(1);
14     while(!q.empty()) {
15         int now = q.front();
16         q.pop();
17         if(now == n) continue;
18         for(int i : v[now]) {
19             if(i != 1 && c[now][i] && !lv[i]) {
20                 lv[i] = lv[now] + 1;
21             }
22         }
23     }
24 }

```

```

25         q.push(i);
26     }
27     return lv[n];
28 }
29 int dfs(int x, int flow) {
30     int ret = 0;
31     if(x == n) return flow;
32     for(int i = head[x]; i < v[x].size(); i++) {
33         int y = v[x][i];
34         head[x] = y;
35         if(c[x][y] && lv[y] == lv[x] + 1) {
36             int d = dfs(y, min(flow, c[x][y]));
37             flow -= d;
38             c[x][y] -= d;
39             c[y][x] += d;
40             ret += d;
41         }
42     }
43     return ret;
44 }
45 signed main() {
46     cin >> n >> m;
47     while(m--) {
48         int x, y, z;
49         cin >> x >> y >> z;
50         if(c[x][y] || c[y][x]) {
51             c[x][y] += z;
52             continue;
53         }
54         v[x].push_back(y);
55         v[y].push_back(x);
56         c[x][y] = z;
57     }
58     while(bfs()) {
59         ans += dfs(1, INT_MAX);
60     }
61     cout << ans;
62 }

```

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 void dfs(int x) {
10     d[x] = low[x] = ++cnt;
11     s.push(x);
12     for(int i : v[x]) {
13         if(scc[i]) continue;
14         if(d[i]) {
15             low[x] = min(low[x], d[i]);
16         } else {
17             dfs(i);
18             low[x] = min(low[x], low[i]);
19         }
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 using namespace std;
2 #define pb push_back
3 #define pii pair<int, int>
4 #define N 100005
5 vector<int> adj[N], bcc[N];
6 stack<int> st;
7 int dfn[N], low[N], tag, bc, root;
8 bitset<N> ap;
9 void dfs(int now, int par = -1) {
10     st.push(now);
11     low[now] = dfn[now] = ++tag;
12     int f = 0;
13     for(int e : adj[now] | views::reverse) {
14         if(dfn[e] < dfn[now]) {
15             dfs(e, now);
16             low[now] = min(low[now], low[e]);
17         } else if(e != par) {
18             f = 1;
19             bcc[bc].push_back(e);
20             if(!f) bc++;
21         }
22     }
23     if(f) bcc[bc].push_back(now);
24     bc++;
25 }

```

```

15     if(e == par) continue;
16     if(!dfn[e]) {
17         dfs(e, now), low[now] = min(low[now], low[e]);
18         if(low[e] >= dfn[now]) {
19             if(++f > 1 || now != root) ap[now] = 1;
20             ++bc;
21             for(; st.top() != now; st.pop())
22                 bcc[bc].pb(st.top());
23             bcc[bc].pb(now);
24         }
25     } else
26         low[now] = min(low[now], dfn[e]);
27 }
28
29 int main() {
30     int n, m, u, v;
31     cin >> n >> m;
32     vector<pii> g(m);
33     for(auto &[u, v] : g)
34         cin >> u >> v, adj[u].pb(v), adj[v].pb(u);
35     for(root = 1; root <= n; ++root)
36         if(!dfn[root]) dfs(root);
37     int ans = 0;
38     for(int i : views::iota(1) | views::take(n))
39         if(ap[i]) ++ans;
40     cout << ans << "\n";
41     for(int i : views::iota(1) | views::take(n))
42         if(ap[i]) cout << i << " ";
43 }

```

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

```

1  #define int long long
2  using namespace std;
3
4  int n, q;
5  int a[200005];
6  int r[200005];
7  int d[200005];
8  int cycle[200005];
9  int len[200005];
10 int cnt = 0;
11 vector<int> v[200005];
12 bitset<200005> vis1;
13 bitset<200005> vis2;
14
15 void findcycle(int x) {
16     while(!vis1[x]) {
17         vis1[x] = 1;
18         x = a[x];
19     }
20     cnt++;
21     cycle[x] = cnt;
22     r[x] = 0;
23     len[cnt] = 1;
24     int temp = a[x];
25     while(temp ^ x) {
26         r[temp] = len[cnt];
27         len[cnt]++;
28         cycle[temp] = cnt;
29         temp = a[temp];
30     }
31 }
32
33 void dfs(int x) {
34     if(vis2[x]) return;
35     vis2[x] = 1;
36     for(int i : v[x]) {
37         dfs(i);
38     }
39 }
40
41 void dfs2(int x) {
42     if(cycle[x] || d[x]) return;
43     dfs2(a[x]);
44     d[x] = d[a[x]] + 1;
45     r[x] = r[a[x]];
46     cycle[x] = cycle[a[x]];
47 }
48
49 signed main() {
50     ios::sync_with_stdio(0);
51     cin.tie(0);
52     cout.tie(0);
53     cin >> n;
54     for(int i = 1; i <= n; i++) {
55         cin >> a[i];
56         v[i].push_back(a[i]);
57         v[a[i]].push_back(i);
58     }
59     for(int i = 1; i <= n; i++) {
60         if(!vis2[i]) {

```

```

61         findcycle(i);
62         dfs(i);
63     }
64 }
65 for(int i = 1; i <= n; i++) {
66     if(!cycle[i] && !r[i]) {
67         dfs2(i);
68     }
69 }
70 for(int i = 1; i <= n; i++) {
71     cout << d[i] + len[cycle[i]] << " ";
72 }
73 }

```

5. DP

5.1. CHO.cpp

```

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

```

5.2. Li-Chao-SegmentTree.cpp

```

1 struct line {
2     int a, b = 1000000000000000000;
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     }
40 }

```

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  using namespace std;
2  #define ll long long
3  #define pb push_back
4  #define pll pair<int, int>
5  #define pdd pair<double, double>
6  #define pll pair<ll, ll>
7  #define F first
8  #define S second
9  #define eps 1e-6
10 int sign(double x) {
11     return fabs(x) < eps ? 0 : x > 0 ? 1 : -1;
12 }
13 int sign(ll x) { return !x ? 0 : x > 0 ? 1 : -1; }
14 template <typename T1, typename T2>
15 istream &operator>>(istream &s, pair<T1, T2> &p) {
16     auto &a = p;
17     s >> a >> b;
18     return s;
19 }
20 template <typename T1, typename T2>
21 ostream &operator<<(ostream &s, const pair<T1, T2> &p) {
22     auto &a = p;
23     s << a << " " << b;
24     return s;
25 }
26 pll operator+(const pll a, const pll b) {
27     return {a.F + b.F, a.S + b.S};
28 }
29 pll operator-(const pll a, const pll b) {
30     return {a.F - b.F, a.S - b.S};
31 }
32 pll operator-(const pll a) { return {-a.F, -a.S}; }
33 pll operator*(const pll a, const pll b) {
34     return {(ll)a.F * b.F, (ll)a.S * b.S};
35 }
36 pdd operator/(const pll a, const double x) {
37     return {a.F / x, a.S / x};
38 }
39 pdd operator*(const pll a, const double x) {
40     return {a.F * x, a.S * x};
41 }
42 pdd operator*(const double x, const pll a) {
43     return {a.F * x, a.S * x};
44 }
45 // 沒有標示幾個 vector 的都是對三個點做事，以第一個點為參考點
46 ll len2(pll p) {
47     return (ll)p.F * p.F + (ll)p.S * p.S;
48 }
49 // 1 vector
50 double len(pll p) { return sqrt((double)len2(p)); }
51 ll cross(pll a, pll b) {
52     return (ll)a.F * b.S - (ll)a.S * b.F;
53 }
54 // 2 vector
55 ll cross(pll p1, pll p2, pll p3) {
56     return cross(p2 - p1, p3 - p1);
57 }
58 // (b-a) cross (c-a)
59 ll dot(pll a, pll b, pll c) {
60     return (ll)(b.F - a.F) * (c.F - a.F) +
61            (ll)(b.S - a.S) * (c.S - a.S);
62 }
63 // (b-a) dot (c-a)
64 ll ori(pll p1, pll p2, pll p3) {
65     return sign(cross(p1, p2, p3));
66 }
67 // normalize to {-1,0,1} (b-a) cross (c-a)
68 bool btw(pll p1, pll p2, pll p3) {
69     return ori(p3, p1, p2) == 0 && dot(p3, p1, p2) <= 0;
70 }
71 // p3 between p1,p2
72 bool banana(pll p1, pll p2, pll p3,
73             pll p4) { // 問兩線段是否香蕉
74     if(btw(p1, p2, p3) || btw(p1, p2, p4) || btw(p3, p4, p1) ||
75        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，算是無限延伸直線的交點
77     // 平行的時候 undefined
78     return cross(p2 - p1, p4 - p1) /
79            ((double)cross(p2 - p1, p4 - p3) * p3 -
80             cross(p2 - p1, p3 - p1) /
81              (double)cross(p2 - p1, p4 - p3) * p4);
82 }
83 pdd proj(pll p1, pll p2, pll p3) {
84     return dot(p1, p2, p3) / ((double)len2(p2 - p1) * (p2 - p1));
85 }
86 double min_dis(pll p1, pll p2,
87                pll p3) { // min distance of p3 to segment p1,p2
88     if(dot(p1, p2, p3) < 0 || dot(p2, p1, p3) < 0)
89         return min(len(p3 - p1), len(p3 - p2));
90     return abs(cross(p1, p2, p3)) / len(p2 - p1);
91 }
92 ll area2(vector<pll> &v) { // 傳入一個多邊形照順序的點集
93     // 起點要出現兩次，回傳兩倍面積
94     // 注意是兩倍才可以 ll 避免浮點數
95     int n = v.size() - 1;
96     ll ans = 0;
97     for(int i = 0; i < n; ++i) ans += cross(v[i], v[i + 1]);
98     return abs(ans);
99 }
100 int in_polygon(vector<pll> &v,
101                pll p) { // 傳入多邊形，起點要出現兩次，回傳
102     // {-1:in, 0:on, 1:out}
103     int n = v.size() - 1, ans = 1;
104     for(int i = 0; i < n; ++i)
105         if(btw(v[i], v[i + 1], p)) return 0;
106     for(int i = 0; i < n; ++i)
107         if(banana(v[i], v[i + 1], p, {(ll)2e9 + 7, p.S + 1LL}))
108             ans *= -1;
109     // 對於任意 p 到 {W, p.S+1}
110     // 的向量中不會有整數點存在，其中需要滿足 {W, p.S+1}
111     // 必須很遠，保證在多邊形外
112     return ans;
113 }
114 void solve() {
115     int n;
116     cin >> n;
117     vector<pll> v(n);
118     for(pll &e : v) cin >> e;
119     v.pb(v[0]);
120     ll ans = area2(v) + 2, ans2 = 0;
121     for(int i = 0; i < n; ++i) {
122         if(v[i].F == v[i + 1].F)
123             ans2 += abs(v[i].S - v[i + 1].S);
124         else if(v[i].S == v[i + 1].S)
125             ans2 += abs(v[i].F - v[i + 1].F);
126         else
127             ans2 += gcd(abs(v[i].F - v[i + 1].F),
128                        abs(v[i].S - v[i + 1].S));
129     }
130     cout << (ans - ans2) / 2 << " " << ans2;
131 }
132 int main() {
133     int t = 1;
134     // cin>>t;
135     for(; t--;) {
136         solve();
137     }
138 }

```

6.2. ConvexHull.cpp

```

1  #define int long long
2  #define fastio
3  ios_base::sync_with_stdio(0);
4  cin.tie(0);
5  cout.tie(0);
6
7  using namespace std;
8
9  template <typename T>
10 pair<T, T> operator-(pair<T, T> a, pair<T, T> b) {
11     return make_pair(a.first - b.first, a.second - b.second);
12 }
13
14 template <typename T> T cross(pair<T, T> a, pair<T, T> b) {
15     return a.first * b.second - a.second * b.first;
16 }
17
18 template <typename T>
19 vector<pair<T, T>> getCH(vector<pair<T, T>> v) {
20     int n = v.size();
21     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;

```

6.3. Inside.cpp

```

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

```

6.4. Intersect.cpp

```

1 struct point {
2     int x, y;
3     point operator+(point b) { return {x + b.x, y + b.y}; }
4     point operator-(point b) { return {x - b.x, y - b.y}; }
5     int operator*(point b) { return x * b.x + y * b.y; }
6     int operator^(point b) { return x * b.y - y * b.x; }
7 };
8
9 bool onseg(point x, point y, point z) {
10    return ((x - z) ^ (y - z)) == 0 && (x - z) * (y - z) <= 0;
11 }
12
13 int dir(point x, point y) {
14     int k = x ^ y;
15     if(k == 0) return 0;
16     if(k > 0) return 1;
17     return -1;
18 }
19
20 bool intersect(point x, point y, point z, point w) {
21     if(onseg(x, y, z) || onseg(x, y, w)) return 1;
22     if(onseg(z, w, x) || onseg(z, w, y)) return 1;
23     if(dir(y - x, z - x) * dir(y - x, w - x) == -1 &&
24         dir(z - w, x - w) * dir(z - w, y - w) == -1) {
25         return 1;
26     }
27     return 0;
28 }

```

6.5. MinimumEuclideanDistance.cpp

```

1 #define int long long
2 #define pii pair<int, int>
3 using namespace std;
4
5 int n;
6 vector<pair<int, int>> v;
7 set<pair<int, int>> s;
8 int dd = LONG_LONG_MAX;
9
10 int dis(pii x, pii y) {
11     return (x.first - y.first) * (x.first - y.first) +
12            (x.second - y.second) * (x.second - y.second);
13 }
14
15 signed main() {
16     ios::sync_with_stdio(0);
17     cin.tie(0);
18     cout.tie(0);
19     cin >> n;
20     for(int i = 0; i < n; i++) {
21         int x, y;
22         cin >> x >> y;

```

```

23         x += 1000000000;
24         v.push_back({x, y});
25     }
26     sort(v.begin(), v.end());
27     int l = 0;
28     for(int i = 0; i < n; i++) {
29         int d = ceil(sqrt(dd));
30         while(l < i && v[i].first - v[l].first > d) {
31             s.erase({v[l].second, v[l].first});
32             l++;
33         }
34         auto x = s.lower_bound({v[i].second - d, 0});
35         auto y = s.upper_bound({v[i].second + d, 0});
36         for(auto it = x; it != y; it++) {
37             dd = min(dd, dis({it->second, it->first}, v[i]));
38         }
39         s.insert({v[i].second, v[i].first});
40     }
41     cout << dd;
42 }

```

7. Tree

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

```

1 #define int long long
2 using namespace std;
3
4 int tree[800005];
5
6 int n, q;
7 int a[200005];
8 int st[200005];
9 int tp[200005];
10 int p[200005];
11 int cnt = 0;
12 int d[200005];
13 int si[200005];
14 vector<int> v[200005];
15 int b[200005];
16
17 void build(int l = 1, int r = n, int index = 1) {
18     if(l == r) {
19         tree[index] = b[l];
20         return;
21     }
22     int mid = (l + r) >> 1;
23     build(l, mid, index << 1);
24     build(mid + 1, r, index << 1 | 1);
25     tree[index] = max(tree[index << 1], tree[index << 1 | 1]);
26 }
27
28 int query(int L, int R, int l = 1, int r = n, int index = 1) {
29     if(L == l && R == r) {
30         return tree[index];
31     }
32     int mid = (l + r) >> 1;
33     if(R <= mid) {
34         return query(L, R, l, mid, index << 1);
35     }
36     if(L > mid) {
37         return query(L, R, mid + 1, r, index << 1 | 1);
38     }
39     return max(query(L, mid, l, mid, index << 1),
40               query(mid + 1, R, mid + 1, r, index << 1 | 1));
41 }
42
43 void modify(int x, int val, int l = 1, int r = n,
44             int index = 1) {
45     if(l == r) {
46         tree[index] = val;
47         return;
48     }
49     int mid = (l + r) >> 1;
50     if(x <= mid) {
51         modify(x, val, l, mid, index << 1);
52     } else {
53         modify(x, val, mid + 1, r, index << 1 | 1);
54     }
55     tree[index] = max(tree[index << 1], tree[index << 1 | 1]);
56 }
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);

```



```

    si[x] += si[i];
}
}
69 void dfs2(int x, int pre, int t) {
71     tp[x] = t;
73     st[x] = ++cnt;
75     int ma = 0;
77     for(int i : v[x]) {
79         if(i == pre) continue;
81         if(si[i] > si[ma]) {
83             ma = i;
85         }
87     }
89     if(!ma) return;
91     dfs2(ma, x, t);
93     for(int i : v[x]) {
95         if(i == pre || i == ma) {
97             continue;
99         }
101         dfs2(i, x, i);
103     }
105     int f(int x, int y) {
107         int ret = 0;
109         while(tp[x] ^ tp[y]) {
111             if(d[tp[x]] < d[tp[y]]) {
113                 swap(x, y);
115             }
117             ret = max(ret, query(st[tp[x]], st[x]));
119             x = p[tp[x]];
121         }
123         if(d[x] > d[y]) swap(x, y);
125         ret = max(ret, query(st[x], st[y]));
127         return ret;
129     }
131     signed main() {
133         ios::sync_with_stdio(0);
135         cin.tie(0);
137         cout.tie(0);
139         cin >> n >> q;
141         for(int i = 1; i <= n; i++) {
143             cin >> a[i];
145         }
147         for(int i = 1; i < n; i++) {
149             int x, y;
151             cin >> x >> y;
153             v[x].push_back(y);
155             v[y].push_back(x);
157         }
159         dfs(1, 0);
161         dfs2(1, 0, 1);
163         for(int i = 1; i <= n; i++) {
165             b[st[i]] = a[i];
167         }
169         build();
171         while(q--) {
173             int mode, x, y;
175             cin >> mode >> x >> y;
177             if(mode == 1) {
179                 modify(st[x], y);
181             } else {
183                 cout << f(x, y) << " ";
185             }
187         }
189     }
191 }

```

7.2. LCA.cpp

```

1 // 洛谷 P1005
2 using namespace std;
3 #define int long long
4 int n, q;
5 int a[200005][21];
6 int d[200005];
7 vector<int> v[200005];
8 void init() {
9     for(int j = 1; j < 21; j++) {
10         for(int i = 1; i <= n; i++) {
11             a[i][j] = a[a[i][j-1]][j-1];
12         }
13     }
14 }
15 void dfs(int x, int pre) {
16     for(int i : v[x]) {
17         if(i == pre) continue;
18         dfs(i, x);
19     }
20     d[x] = d[pre] + 1;
21 }

```

```

23     a[i][0] = x;
25     d[i] = d[x] + 1;
27     dfs(i, x);
29 }
31 int lca(int x, int y) {
33     while(d[x] ^ d[y]) {
35         if(d[x] < d[y]) {
37             swap(x, y);
39         }
41         int k = _lg(d[x] - d[y]);
43         x = a[x][k];
45     }
47     if(x == y) {
49         return x;
51     }
53     for(int i = 20; i >= 0; i--) {
55         if(a[x][i] != a[y][i]) {
57             x = a[x][i];
59             y = a[y][i];
61         }
63     }
65     return a[x][0];
67 }
69 signed main() {
71     ios::sync_with_stdio(0);
73     cin.tie(0);
75     cout.tie(0);
77     cin >> n >> q;
79     for(int i = 1; i < n; i++) {
81         int x, y;
83         cin >> x >> y;
85         v[x].push_back(y);
87         v[y].push_back(x);
89     }
91     dfs(1, 0);
93     init();
95     while(q--) {
97         int x, y;
99         cin >> x >> y;
101         int k = lca(x, y);
103         cout << (d[x] + d[y] - 2 * d[k]) << "\n";
105     }
107 }

```

8. Misc

8.1. BigNum(luoguP1005).cpp

```

1 // 洛谷 P1005
2 using namespace std;
3 #define N 85
4 #define LL long long
5 #define pii pair<int, int>
6 #define F first
7 #define S second
8 struct num {
9     const static LL base = 1000000000LL; // base 1e9
10     LL p[505], len;
11     num() {
12         memset(p, 0, sizeof(p));
13         len = 0;
14     }
15     num(LL x) {
16         memset(p, 0, sizeof(p));
17         len = 0;
18         for(p[len++] = x; p[len-1] >= base; ++len)
19             p[len] = p[len-1] / base, p[len-1] %= base;
20     }
21     num operator=(LL x) {
22         memset(p, 0, sizeof(p));
23         len = 0;
24         for(p[len++] = x; p[len-1] >= base; ++len)
25             p[len] = p[len-1] / base, p[len-1] %= base;
26         return *this;
27     }
28     num max(const num &b) {
29         if(len != b.len) return len > b.len ? *this : b;
30         for(int i = len; i--;)
31             if(p[i] != b.p[i]) return p[i] > b.p[i] ? *this : b;
32         return *this;
33     }
34     num operator+(const num &b) {
35         num c;
36         LL x = 0;
37         for(LL i = c.len; i < len || i < b.len; ++i) {
38             c.p[i] = p[i] + b.p[i] + x;
39             x = c.p[i] / base;
40             c.p[i] %= base;
41         }
42     }

```

```

    }
    if(x) c.p[c.len++] = x;
    return c;
}
num operator*(LL b) {
    num c;
    c.len = len;
    LL x = 0;
    for(LL i = 0; i < len; ++i) {
        c.p[i] = p[i] * b + x;
        x = c.p[i] / base;
        c.p[i] %= base;
    }
    for(; x; x /= base) c.p[c.len++] = x % base;
    return c;
}
} dp[N][N], ans;
ostream &operator<<(ostream &s, num a) {
    if(!a.len) return s << "0";
    s << a.p[a.len - 1];
    for(int i = a.len - 1; i--;) {
        if(!a.p[i])
            s << "000000000";
        else {
            for(int k = 10; k * a.p[i] < (LL)1e9; k *= 10)
                s << "0";
            s << a.p[i];
        }
    }
    return s;
}
LL a[N];
int main() {
    ios::sync_with_stdio(0);
    cin.tie(0);
    cout.tie(0);
    int n, m, i, j;
    for(cin >> n >> m; n--;) {
        for(i = 0; i < m; ++i) cin >> a[i];
        for(i = 0; i < m; ++i)
            for(j = 0; j < m; ++j) dp[i][j] = 0;
        for(i = 0; i < m; ++i) dp[i][i] = a[i] << 1;
        for(j = 1; j < m; ++j)
            for(i = 0; i + j < m; ++i)
                dp[i][i + j] =
                    (dp[i][i + j - 1] + a[i + j]) *
                    .max(dp[i + 1][i + j] + a[i]) *
                    2;
        ans = ans + dp[0][m - 1];
    }
    cout << ans;
}

```

8.2. Tri-search.cpp

```

1 using namespace std;
2 int n;
3 double a[15], x, y;
4
5 double get(double x) {
6     double ret = 0;
7     double k = 1;
8     for(int i = 0; i <= n; i++) {
9         ret += k * a[i];
10        k *= x;
11    }
12    return -ret;
13 }
14
15 template <class T> T bi_search(T l, T r, T end) {
16     if(!check(r - end)) return r - end;
17     for(; r - l > end;) {
18         T mid = (l + r) / 2;
19         if(check(mid))
20             r = mid;
21         else
22             l = mid;
23     }
24     return l;
25 }
26 /*check gives 000000001111 find the last 0*/
27
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(midr - midl < end) break;
34         if(get(midr) > get(midl))
35             r = midr;
36         else
37             l = midl;
38     }
39 }

```

```

39 }
40 for(; r - l > end;) {
41     midl = (l + r) / 2;
42     if(get(r) > get(l))
43         r = midl;
44     else
45         l = midl;
46 }
47 return l;
48 }
49 /*get gives the value, find the minimum*/
50
51 int main() {
52     cin >> n >> x >> y;
53     for(int i = n; i >= 0; i--) {
54         cin >> a[i];
55     }
56     cout << fixed << setprecision(7)
57         << tri_search<double>(x, y, 1e-7);
58 }

```

9. AnotherVersionDataStructure

9.1. BIT.cpp

```

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

```

9.2. DSU.cpp

```

1 template <class T> class Dsu {
2     #define N 2000005
3     public:
4         T dsu[N], size[N];
5         Dsu(T n) {
6             for(n; --n; dsu[n] = n, size[n] = 1;
7         }
8         T qry(T x) {
9             if(dsu[x] == x) return x;
10            return dsu[x] = qry(dsu[x]);
11        }
12        void merge(T a, T b) {
13            a = qry(a);
14            b = qry(b);
15            if(a == b) return;
16            if(size[a] < size[b])
17                dsu[a] = b, size[b] += size[a];
18            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
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

```

```

21     sz = 1;
22     mx = mn = sum = val;
23     if(l)
24         sz += l->sz, mx = max(mx, l->mx),
25         mn = min(mn, l->mn), sum += l->sum;
26     if(r)
27         sz += r->sz, mx = max(mx, r->mx),
28         mn = min(mn, r->mn), sum += r->sum;
29 }
30 void push() {
31     if(rev_tag) swap(l, r);
32     if(l) l->add_tag ^= add_tag, l->rev_tag ^= rev_tag;
33     if(r) r->add_tag ^= add_tag, r->rev_tag ^= rev_tag;
34     mx ^= add_tag;
35     mn ^= add_tag;
36     sum ^= add_tag;
37     add_tag = 0;
38     rev_tag = 0;
39 }
40 };
41 void debug(node *tr) {
42     if(!tr) return;
43     tr->push();
44     tr->pull();
45     debug(tr->l);
46     cout << tr->val << " ";
47     debug(tr->r);
48 }
49 void debug2(node *tr) {
50     if(!tr) return;
51     tr->push();
52     tr->pull();
53     cout << tr->val << " ";
54     debug2(tr->l);
55     debug2(tr->r);
56 }
57 int sz(node *tr) { return tr ? tr->sz : 0; }
58 node *merge(node *a, node *b) {
59     if(!a || !b) return a ? b;
60     a->push();
61     b->push();
62     if(mt() % (sz(a) + sz(b)) < sz(a)) {
63         a->r = merge(a->r, b);
64         a->pull();
65         return a;
66     }
67     b->l = merge(a, b->l);
68     b->pull();
69     return b;
70 }
71 pnn split(node *tr, int v) { //(-inf,v),(v,inf)
72     if(!tr) return {0, 0};
73     tr->push();
74     if(tr->val <= v) {
75         auto [l, r] = split(tr->r, v);
76         tr->r = l;
77         tr->pull();
78         return {tr, r};
79     }
80     auto [l, r] = split(tr->l, v);
81     tr->l = r;
82     tr->pull();
83     return {l, tr};
84 }
85 pnn splitsz(node *tr, int k) { //[rk.1,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 *eraskth(node *tr, int k) {
123     auto [l, r] = splitsz(tr, k - 1);
124     return merge(l, splitsz(r, k).S);
125 }
126 int rnk(node *tr, int v) {
127     if(!tr) return 0;
128     if(tr->val <= v) return sz(tr->l) + 1 + rnk(tr->r, v);
129     return rnk(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->sz; // 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 << rnk(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     }
209 }

```

9.4. Treap 但可以多個數縮點 (疑似爛的).cpp

```

1 // treap 模板 洛谷 P3369 【模板】普通平衡树
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         return a;
59     }
60     b->l = merge(a, b->l);
61     b->pull();
62     return b;
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;

```

```

89         tr->r = 0;
90         tr->pull();
91         return {tr, r};
92     } else {
93         auto [l, r] = splitsz(tr->r, k - (sz(tr->l) + 1));
94         tr->r = l;
95         tr->pull();
96         return {tr, r};
97     }
98 }
99 node *insert(node *tr, int val = 0, int num = 1) {
100     auto [l, r] = split(tr, val);
101     return merge(merge(l, new node(val, num)), r);
102 }
103 node *insertkth(node *tr, int k) {
104     auto [l, r] = splitsz(tr, k - 1);
105     return merge(merge(l, new node()),
106                 r); // new node 拿來區間操作初始化
107 }
108 node *eraseall(node *tr, int v) {
109     auto [l, r] = split(tr, v - 1);
110     return merge(l, split(r, v).S);
111 }
112 node *eraseone(node *tr, int v) {
113     auto [l, r] = split(tr, v - 1);
114     return merge(l, splitsz(r, 1).S);
115 }
116 node *erasekth(node *tr, int k) {
117     auto [l, r] = splitsz(tr, k - 1);
118     return merge(l, splitsz(r, k).S);
119 }
120 int rnk(node *tr, int v) {
121     if(!tr) return 0;
122     if(tr->val <= v) return sz(tr->l) + 1 + rnk(tr->r, v);
123     return rnk(tr->l, v);
124 }
125 int kth(node *tr, int k) {
126     auto [l, x] = splitsz(tr, k - 1);
127     auto [m, r] = splitsz(x, 1);
128     if(!m) return 0;
129     int ans = m->val;
130     tr = merge(merge(l, m), r);
131     return ans;
132 }
133 int count(node *tr, int L, int R) { // count[L,R]
134     auto [l, x] = split(tr, L - 1);
135     auto [m, r] = split(x, R);
136     int ans = m->sum; // 看要改啥
137     tr = merge(merge(l, m), r);
138     return ans;
139 }
140 int countkth(node *tr, int L, int R) { // count[rk.L,rk.R]
141     auto [l, x] = splitsz(tr, L - 1);
142     auto [m, r] = splitsz(x, R - L);
143     int ans = m->sum; // 看要改啥
144     tr = merge(merge(l, m), r);
145     return ans;
146 }
147 int prev(node *tr, int v) {
148     auto [x, r] = split(tr, v - 1);
149     auto [l, m] = splitsz(x, sz(x) - 1);
150     int ans = m->val;
151     tr = merge(merge(l, m), r);
152     return ans;
153 }
154 int next(node *tr, int v) {
155     auto [l, x] = split(tr, v);
156     auto [m, r] = splitsz(x, 1);
157     int ans = m->val;
158     tr = merge(merge(l, m), r);
159     return ans;
160 }
161 int qry(node *tr, int L, int R) { // qry[L,R]
162     auto [l, x] = splitsz(tr, L - 1);
163     auto [m, r] = splitsz(x, R - L);
164     int ans = m->sum; // 看要改啥
165     tr = merge(merge(l, m), r);
166     return ans;
167 }
168 void modify(node *tr, int L, int R, int v) { // modify[L,R]
169     auto [l, x] = splitsz(tr, L - 1);
170     auto [m, r] = splitsz(x, R);
171     m->val += v;
172     m->add_tag += v; // 看要改啥
173     tr = merge(merge(l, m), r);
174 }
175 signed main() {
176     vector<node *> tr(2);
177     int n, m;
178     scanf("%lld%lld", &n, &m);
179     for(int i = 1, x; i <= n; ++i)
180         scanf("%lld", &x), (x) && (tr[i] = insert(tr[i], i, x));
181     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 }

```

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
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) : a(_a), b(_b), l(_l), r(_r), id(_id) {}
10     double operator()(int x) { return a * x + b; }
11 } line[N];
12 int seg[N << 2];
13 #define lid (id << 1)
14 #define rid (id << 1 | 1)
15 #define M (L + R >> 1)
16 #define eps 1e-6
17 void ins(int l, int L = 1, int R = N, int id = 1) {
18     // cout<<"ins{"<<line[l].a<<","<<line[l].b<<","<<line[l].l<<","<<line[l].r<<"}<<"\n";
19     // "<<R<<"<<"\n";
20     if(line[l].r < L || R < line[l].l) return;
21     if(L == R) {
22         if(line[l].M - line[seg[id]].M > eps) seg[id] = l;
23         return;
24     }
25     if(line[l].l <= M && M <= line[l].r &&
26         line[l].M - line[seg[id]].M > eps)
27         swap(l, seg[id]);
28     if(line[l].l <= L && R <= line[l].r) {
29         if(line[l].a - line[seg[id]].a > eps)
30             ins(l, M + 1, R, rid);
31         else
32             ins(l, L, M, lid);
33     }
34     /*if(line[l].a>line[seg[id]].a)*/ ins(l, M + 1, R, rid);
35     /*else */ ins(l, L, M, lid);
36 }
37 int qry(int x, int L = 1, int R = N, int id = 1) {
38     // cout<<"qry"<<x<<"}<<line[seg[id]].a<<","<<line[seg[id]].b<<"}<<"\n";
39     // "<<R<<" "<<id<<"<<"\n";
40     if(L == R) return seg[id];
41     int k = (x <= M ? qry(x, L, M, lid) : qry(x, M + 1, R, rid)),
42         not_k = 0, not_seg = 0;
43     if(line[k].r < x || x < line[k].l) not_k = 1;
44     if(line[seg[id]].r < x || x < line[seg[id]].l) not_seg = 1;
45     if(not_k && not_seg) return 0;
46     if(not_k) return seg[id];
47     if(not_seg) return k;
48     return line[k](x) - line[seg[id]](x) > eps ? k : seg[id];
49 }
50 int main() {
51     int n, ans = 0, p = 1;
52     for(cin >> n; n--;) {
53         int op;
54         cin >> op;
55         if(op) {
56             int x0, y0, x1, y1;
57             cin >> x0 >> y0 >> x1 >> y1;
58             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)<<"\n";
66         // "<<y0-x0*((double)y1-y0)/(x1-x0)<<"\n";
67         if(x0 != x1)
68             line[p] = Line(((double)y1 - y0) / (x1 - x0),
69                 y0 - x0 * ((double)y1 - y0) / (x1 - x0),
70                 x0, x1, p);
71         else
72             line[p] = Line(0, max(y0, y1), x0, x1, p);
73         ins(p);
74         ++p;
75     } else {
76         int k;
77         cin >> k;
78         k = (k + ans - 1) % 39989 + 1;
79         cout << (ans = qry(k)) << "\n";
80     }
81 }
82 // cout<<qry(9)<<"\n";
83 }

```

9.6. 單點修改動態開點線段樹.cpp

```

1 using namespace std;
2 #define N 200005
3 #define M int m = l + r >> 1
4 #define MAX 1000000000
5 int a[N];
6 typedef struct node {
7     struct node *l, *r;
8     int val;
9 } node;
10 void check(node *tree, int flag) {
11     if(flag && !tree->r)
12         tree->r = (node *)malloc(sizeof(struct node)),
13         tree->r->val = 0;
14     else if(!flag && !tree->l)
15         tree->l = (node *)malloc(sizeof(struct node)),
16         tree->l->val = 0;
17 }
18 void upd(int pos, int val, int l, int r, node *tree) {
19     tree->val += val;
20     if(l == r) return;
21     M;
22     if(pos > m)
23         check(tree, 1), upd(pos, val, m + 1, r, tree->r);
24     else
25         check(tree, 0), upd(pos, val, l, m, tree->l);
26 }
27 int qry(int a, int b, int l, int r, node *tree) {
28     if(!tree) return 0;
29     if(a <= l && r <= b) return tree->val;
30     M;
31     if(a > m) return qry(a, b, m + 1, r, tree->r);
32     if(b <= m) return qry(a, b, l, m, tree->l);
33     return qry(a, b, m + 1, r, tree->r) +
34         qry(a, b, l, m, tree->l);
35 }
36 int main() {
37     int n, q, i = 1, x;
38     node *root = (node *)malloc(sizeof(struct node));
39     root->val = 0;
40     for(scanf("%d %d", &n, &q); i <= n; ++i)
41         getchar(), scanf("%d", &a[i]),
42         upd(a[i], 1, 1, MAX, root);
43     // printf("%d %d %d\n", a[1], a[2], a[3]);
44     // "<<L<<" "<<R<<" "<<id<<"<<"\n";
45     for(q--; q > 0; --q) {
46         int op;
47         scanf("%d", &op);
48         if(op == 1) {
49             int x, y;
50             scanf("%d %d", &x, &y);
51             if(x <= 1 && y <= MAX)
52                 upd(a[x], -1, 1, MAX, root),
53                 a[x] = y, upd(y, 1, 1, MAX, root);
54             else
55                 printf("%d\n", qry(x, i, 1, MAX, root));
56         }
57     }
58 }

```

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
30 private:
31  T init(int l = 1, int r = n, int id = 1) {
32      if(l == r) return seg[id] = a[l];
33      int m = l + r >> 1;
34      return seg[id] = init(l, m, lid) + init(m + 1, r, rid);
35  }
36  #undef lid
37  #undef rid
38  #undef N
39  };
40  /*1based 陣列 1based id 單點修改 預設維護區間和 */

```

9.8. 懶標線段樹.cpp

```

1  struct Seg {
2      #define lid (id << 1)
3      #define rid ((id << 1) | 1)
4      #define M (L + R >> 1)
5      #define N 200005
6      LL seg[N << 2], tag[N << 2];
7      void inline addtag(int id, LL v, int L, int R) {
8          seg[id] += v * (R - L + 1);
9          tag[id] += v;
10     }
11     void inline push(int id, int L, int R) {
12         addtag(lid, tag[id], L, M);
13         addtag(rid, tag[id], M + 1, R);
14         tag[id] = 0;
15     }
16     void inline pull(int id) { seg[id] = seg[lid] + seg[rid]; }
17     void init(int L = 1, int R = n, int id = 1) {
18         if(L == R) {
19             seg[id] = 0;
20             tag[id] = 0;
21             return;
22         }
23         init(L, M, lid);
24         init(M + 1, R, rid);
25         pull(id);
26     }
27     void upd(int l, int r, LL v, int L = 1, int R = n,
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;
52 /*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
```

```

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 using namespace std;
2 #define LL long long
3 #define uLL __uint128_t
4 #define sub(a, b) ((a) < (b) ? (b) - (a) : (a) - (b))
5 template <class T, class POW>
6 void fastpow(T x, POW n, POW p, T &ans) {
7     for(; n >= 1; ) {
8         if(n & 1) {
9             ans *= x;
10            ans %= p;
11        }
12        x *= x;
13        x %= p;
14    }
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

```

```

29     d >= 1;
    ans = 1;
    fastpow(p, d, x, ans);
31     if(ans == x - 1)
        return 0;
33     else if(ans != 1)
        return 1;
35 }
    return 0;
37 }
bool miller_rabin(const uLL x) {
39     if(x == 1) return 0;
    for(auto e : pri) {
41         if(e >= x) return 1;
        if(check(x, e)) return 0;
43     }
    return 1;
45 }
template <class T> T gcd(T a, T b) {
47     if(!a) return b;
    if(!b) return a;
49     if(a & b & 1) return gcd(sub(a, b), min(a, b));
    if(a & 1) return gcd(a, b >> 1);
51     if(b & 1) return gcd(a >> 1, b);
    return gcd(a >> 1, b >> 1) << 1;
53 }
/*gcd(a,b) 默認 gcd(a,0)=a*/
mt19937 rnd(time(0));
template <class T> T f(T x, T c, T mod) {
57     return (((uLL)x) * x % mod + c) % mod;
59 }
template <class T> T rho(T n) {
    T mod = n, x = rnd() % mod, c = rnd() % (mod - 1) + 1,
61     p = 1;
    for(T i = 2, j = 2, d = x; ++i) {
63         x = f(x, c, mod), p = ((uLL)p) * sub(x, d) % mod;
        if(i % 127 == 0 && gcd(p, n) != 1) return gcd(p, n);
65         if(i == j) {
            j <= 1, d = x;
67             if(gcd(p, n) != 1) return gcd(p, n);
        }
69     }
71 }
template <class T> T pollard_rho(T n) {
    if(miller_rabin(n)) return n;
73     T p = n;
    while(p == n) p = rho(n);
75     return max(pollard_rho(p), pollard_rho(n / p));
77 }
int main() {
    LL t, n, ans;
79     for(cin >> t; t--;) {
        cin >> n;
81         ans = pollard_rho(n);
        if(ans == n)
83             puts("Prime");
        else
85             printf("%lld\n", ans);
87     }
}

```

10.3. 快速幂.cpp

```

1  template <class T, class POW>
    void fastpow(T x, POW n, POW p, T &ans) {
3      for(; n; n >>= 1) {
          if(n & 1) {
5              ans *= x;
              ans %= p;
7          }
          x *= x;
          x %= p;
9      }
11 }
/* 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
    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) {
    if(!b) {
3      x = 1;
      y = 0;
5      return a;
    }
    T ans = extgcd(b, a % b, y, x);
7     y -= a / b * x;
9     return ans;
11 }
/*extgcd(a,b,x,y)=ax+by,x 跟 y 是會被修改的參數 */
template <class T> T modeq(T a, T b, T p) {

```

```

13     T x, y, d = extgcd(a, p, x, y);
    if(b % d) return 0;
15     return ((b / d * x) % p + p) % p;
17 }
/*x=modeq(a,b,n),ax=b(mod n),0<=x<n
    modeq(a,1,n) 相當於求 a 在 mod n 下的逆元 */
19 template <class T> T gcd(T a, T b) {
    if(!a) return b;
21     if(!b) return a;
    if(a & b & 1) return gcd(abs(a - b), min(a, b));
23     if(a & 1) return gcd(a, b >> 1);
    if(b & 1) return gcd(a >> 1, b);
25     return gcd(a >> 1, b >> 1) << 1;
27 }
/*gcd(a,b) 默認 gcd(a,0)=a*/
ll crt(V<ll> &p, V<ll> &a) {
29     ll n = 1, ans = 0, k = a.size();
    for(ll &e : p) n *= e;
31     for(int i = 0; i < k; ++i)
        ans = (ans + a[i] * n / p[i] % n *
33             modeq(n / p[i], 1LL, p[i]) % n) %
        n;
35     return (ans % n + n) % n;
37 }
/*(a+b)^p ≡ a+b ≡ a^p+b^p (mod p) (小費馬)
    (p-1)! ≡ -1 (mod p) (威爾遜定理)
39 v(n) := n中p的幕次, (n)_p := n! / p^{v(n)},
    s(n) := p進制下n的所有位數和
41 v(n!) = ∑_{i=1}^∞ ⌊ n / p^i ⌋ (勒讓德定理)
    ⌊ n / p^i ⌋ = (n-s(n)) / (p-1) (庫默爾定理)
43 v((n choose m)) = (s(n)+s(m-n)-s(m)) / (p-1) (庫默爾定理)
    v((m1, m2, ..., mk) choose n) = ∑_{i=1}^k (s(mi)-s(n)) / (p-1) (庫默爾定理推廣)
45 \[
    (n!)_p \equiv -1^{\lfloor \frac{n}{p} \rfloor} \pmod p
47 \]
49 \[
    ((\lfloor \frac{n}{p} \rfloor)!)_p \equiv (-1)^{\lfloor \frac{n}{p} \rfloor} \pmod p
51 \]
    打階乘表 + 迭代這條式子可以 O(p + log_p(n)) (mod 下階乘)
53 \[ \binom{n}{m} \equiv \frac{((n+m)!)_p}{(n!)_p (m!)_p} \pmod{p^q} \]
    把 p 從 C(n, m) 裡面隔離掉了 就能用上面的
55 (n!)_p 模逆元 (mod 下階乘推廣至二項式)
57 ((p^q)!)_p ≡ ±1 (mod p^q) (威爾遜定理推廣)
59 \[ \binom{n}{m} \equiv \binom{\lfloor \frac{n}{p} \rfloor}{\lfloor \frac{m}{p} \rfloor} \pmod{p^q} \]
61 \[ \lfloor \frac{n}{p} \rfloor! \equiv (-1)^{\lfloor \frac{n}{p} \rfloor} \pmod p \]
    (Lucas 定理) 打階乘表跟模逆元表 + 迭代這條式子可以 O(p + log_p(n))
63 若 p 進制下任何一位 i 滿足 n_i < m_i 則
    (n_i choose m_i) % p = 0
65 則因 (n choose m) = ∏_{i=0}^{max(log_p(a), log_p(b))} (n_i choose m_i) % p
    (n_i choose m_i) % p 導致 (n choose m) % p = 0
67 設 p = 2 則有 (n choose m) 是奇數的充要條件為二進制下每一位
    n < m (Lucas 定理額外性質) Lucas 定理可由此生成函數做法得到
69 不依賴小費馬 對多項式也成立 根據上述
    (n choose m) % k 可將 k 做唯一質數分解
71 個別做完再做 crt 得到結果 (exLucas 定理)
73 \[
    \text{卡特蘭數 } C(0)=C(1)=1, n>1 \text{ 時 } C(n)=\sum_{k=0}^{n-1} C(k)C(n-1-k)=\frac{1}{n} \binom{2n-2}{n-1}
75 \]
    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
77 C(n)k^n
    模逆元表 p=i*(p/i)+p%i, -p%i=i*(p/i), inv(i)=-(p/i)*inv(p%i)*/
79 LL fracp[N], invp[N];
    void fracp_init(LL p) {
81         fracp[0] = 1;
        for(int i = 1; i < p; ++i) fracp[i] = fracp[i - 1] * i % p;
83     }
    void invp_init(LL p) {
85         invp[0] = invp[1] = 1;
        for(int i = 2; i < p; ++i)
87             invp[i] = p - (p / i * invp[p % i]) % p;
89 }
/* 階乘表跟模逆元表 之後可以考慮改一下長相 */
template <class T> T lucas(T n, T m, T p) {
91     if(!m) return 1;
    if(m > n || m % p > n % p) return 0;
93     return lucas(n / p, m / p, p) * fracp[n % p] % p *
        invp[fracp[m % p]] % p;
95 }
97 /*lucas(n,m,p)=C(n,m)%p 要求要帶階乘表跟模逆元表
    * O(p+log_p(n))*/
99 /* 米勒拉賓質數 2,325,9375,28178,450775,9780504,1795265022*/
/*crt 質數
101 (2^16)+1 65537 3

```

```

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

```

10.5. 篩法.cpp

```

1 // 待加入分塊篩
2 template <class T> class Prime {
3 #define N (int)1e8 + 9
4 public:
5     vector<T> list, factor;
6     Prime(T n) {
7         eular(n);
8         // eratosthenes(n);
9         // sqrt_sieve
10        // factorize(n);
11    }
12    void show() {
13        for(T e : list) printf("%lld ", e);
14        putchar('\n');
15    }
16 private:
17     bitset<N> notprime; // 1e8<2^27=128MB
18     void eular(T n) {
19         for(T i = 2; i <= n; ++i) {
20             if(!notprime[i]) list.emplace_back(i);
21             const T k = n / i;
22             for(T j : list) {
23                 if(j > k) break;
24                 notprime[i * j] = 1;
25                 if(!(i % j)) break;
26             }
27         }
28     }
29     void eratosthenes(T n) {
30         for(T i = 2; i <= n; ++i) {
31             if(!notprime[i]) list.emplace_back(i);
32             const T k = n / i;
33             for(T j : list) {
34                 if(j > k) break;
35                 notprime[i * j] = 1;
36                 if(!(i % j)) break;
37             }
38         }
39     }
40     void sqrt_sieve(T n) {
41         for(T i = 2; i <= n; ++i) {
42             bool isprime = 1;
43             for(T j : list) {
44                 if(j > i / j) break;
45                 if(!(i % j)) {
46                     isprime = 0;
47                     break;
48                 }
49             }
50             if(isprime) list.emplace_back(i);
51         }
52     }
53     void factorize(T n) {
54         factor = vector<T>(n);
55         if(list.empty()) eular(n);
56         for(T j : list) factor[j] = j;
57         for(T i = 2; i <= n; ++i) {
58             const T k = n / i;
59             for(T j : list) {
60                 if(j > k) break;
61                 factor[i * j] = j;
62                 if(!(i % j)) break;
63             }
64         }
65     }
66 #undef N
67 };
68 /*Prime prime(n) 建立打好 1~n 質數表的物件
69 prime.list(一個 vector) 是質數表
70 可修改 define N 決定歐篩/埃篩上限
71 可在建構子選擇篩法 有歐篩/埃篩/根號暴力搜
72 prime.factorize(n) 用歐篩方式得到 1~n 所有數的最小質因數
73 可在 factor(一個 vector) 上一路回溯 logn 得到一個數的質因數分解
74 做 n 個數質因數分解共花 nlogn
75 show() 會以空格隔開 顯示所有 list 內的元素 有尾空格尾換行
76 printf 裡面用%lld 視情況換為%d 或 cout*/

```

11. AnotherVersionString

11.1. KMP (2).cpp

```

1 #define V vector
2 V<int> kmp(string s) {

```

```

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

```

11.2. KMP.cpp

```

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

```

11.3. Manacher (2).cpp

```

1 #define T(x) ((x)&1 ? s[(x) >> 1] : '.')
2 int ex(string &s, int l, int r, int n) {
3     int i = 0;
4     while(l - i >= 0 && r + i < n && T(l - i) == T(r + i)) ++i;
5     return i;
6 }
7 int manacher(string s, int n) {
8     n = 2 * n + 1;
9     int mx = 0;
10    int center = 0;
11    vector<int> r(n);
12    int ans = 1;
13    r[0] = 1;
14    for(int i = 1; i < n; i++) {
15        int ii = center - (i - center);
16        int len = mx - i + 1;
17        if(i > mx) {
18            r[i] = ex(s, i, i, n);
19            center = i;
20            mx = i + r[i] - 1;
21        } else if(r[ii] == len) {
22            r[i] = len + ex(s, i - len, i + len, n);
23            center = i;
24            mx = i + r[i] - 1;
25        } else {
26            r[i] = min(r[ii], len);
27        }
28        ans = max(ans, r[i]);
29    }
30    return ans - 1;
31 }

```

11.4. Manacher.cpp

```

1 #define V vector
2 string manacher(string t) {
3     int n = t.size() << 1 | 1;
4     string s(n, '#');
5     for(int i = 0, m = t.size(); i < m; ++i)
6         s[i << 1 | 1] = t[i];
7     V<int> p(n);
8     for(int i = 0, m = 0, r = 0; i < n; ++i) {
9         p[i] = r > i ? min(r - i, p[m - (i - m)]) : 1;
10        for(; i - p[i] >= 0 && i + p[i] < n &&
11            s[i - p[i]] == s[i + p[i]];)
12            ++p[i];
13        if(i + p[i] > r) r = i + p[i], m = i;
14    }
15    int k = 0;

```

```

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

```

11.5. Z.cpp

```

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

```

12. AnotherVersionGraph

12.1. Dijkstra.cpp

```

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

```

12.2. SCC.cpp

```

using namespace std;
#define pb push_back
#define pii pair<int, int>
#define N 100005
vector<int> adj[N];
stack<int> st;
int dfn[N], low[N], tag, scc[N], scchead[N], sc;
bitset<N> in;
void dfs(int now, int par = -1) {
    st.push(now);
    in[now] = 1;
    low[now] = dfn[now] = ++tag;
    for(int e : adj[now]) {
        if(e == par) continue;
        if(!dfn[e])
            dfs(e, now), low[now] = min(low[now], low[e]);
        else if(in[e])

```

```

        low[now] = min(low[now], dfn[e]);
    }
    if(dfn[now] == low[now]) {
        ++sc;
        for(; st.top() != now; st.pop())
            scc[st.top()] = sc, in[st.top()] = 0;
        st.pop();
        scc[now] = sc;
        in[now] = 0;
        scchead[sc] = now;
    }
}
int main() {
    int n, m, u, v;
    cin >> n >> m;
    vector<pii> g(m);
    for(auto &[u, v] : g)
        cin >> u >> v, adj[u].pb(v), adj[v].pb(u);
    for(u = 1; u <= n; ++u)
        if(!dfn[u]) dfs(u);
    int ans = 0;
    for(auto &[u, v] : g)
        if(scc[u] != scc[v]) ++ans; //eBCC
    cout << ans << "\n";
    for(auto &[u, v] : g)
        if(scc[u] != scc[v]) cout << u << " " << v << "\n";
}

```

12.3. cses 有向圖基環樹森林.cpp

```

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

```

```

        else if(num[u] == num[v] && dis[u] >= dis[v] &&
            jmp(u, dis[u] - dis[v]) == v)
            cout << dis[u] - dis[v] << "\n";
        else
            cout << "-1\n";
    } else
        cout << "-1\n";
}
}
}

```

13. AnotherVersionGeometry

13.1. DynamicHull.cpp

```

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    }
34 };

```

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 }
12 int jmp(int x, int k) {
13     for(int i = LG; i--;)
14         for(; k >= 1 << i; k -= 1 << i) x = par[x][i];
15     return x;
16 }
17 int lca(int a, int b) {
18     if(dep[a] > dep[b]) swap(a, b);
19     b = jmp(b, dep[b] - dep[a]);
20     if(a == b) return a;
21     for(int i = LG; i--;)
22         for(; par[a][i] != par[b][i]; b = par[b][i])
23             a = par[a][i];
24     return par[a][0];
25 }
26 int main() {
27     int n;
28     cin >> n;
29     for(int i = n, u, v; --i;)
30         cin >> u >> v, g[u].pb(v), g[v].pb(u);
31     dfs();
32     for(int i = 1; i < LG; ++i)
33         for(int j = 1; j <= n; ++j)
34             par[j][i] = par[par[j][i - 1]][i - 1];
35     int k = lca(1, n);
36 }
37 // 點編號 1~n，建的無向圖但改 dfs

```

```

// 就能變有向，改有向記得邊要反著建 dep[n] 代表 n 的深度 (1
// base) : par[i][j] 代表 i 往上 1<<j 步的祖先是誰，不存在則是
// 0，sub[i] 代表 i 的子樹大小 jmp(i,j) 代表 i 往上 j
// 步的祖先是誰
43 #pragma GCC optimize(
44     "Ofast,fast-math,unroll-loops,no-stack-protector")
45
46 using namespace std;
47 #define ll long long
48 #define pb push_back
49 #define N 200005
50 #define pii pair<int, int>
51 #define V vector
52 #define inf 1000000007
53 #define M 200005
54 #define LG 18
55 #define pii pair<int, int>
56 #define ppp pair<pii, pii>
57 char buf[1 << 22], *p1, *p2;
58 int p[12];
59 #define gc()
60     (p1 == p2 &&
61         (p2 = (p1 = buf) + fread(buf, 1, 1 << 22, stdin),
62         p1 == p2)
63         ? EOF
64         : *p1++)
65 inline int gi() {
66     int x = 0;
67     for(char c; '0' <= (c = gc()) && c <= '9'; x += c - '0')
68         x *= 10;
69     return x;
70 }
71 inline void pi(int x, char c = ' ') {
72     if(!x) putchar('0');
73     int i = 0;
74     for(; x; x /= 10) p[i++] = x % 10;
75     for(; i--;) putchar(p[i] + '0');
76     putchar(c);
77 }
78 int main() {
79     cin.tie(0)->sync_with_stdio(0);
80     int n, m, q;
81     cin >> n >> m >> q;
82     vector<ppp> g(m);
83     bitset<M> ans;
84     vector<vector<pii>> adj(n + 1, vector<pii>());
85     for(int i = 0; i < m; ++i) {
86         auto &p1, &p2 = gi();
87         auto &w, &idx = p1;
88         auto &u, &v = p2;
89         cin >> u >> v >> w;
90         idx = i;
91     }
92     sort(g.begin(), g.end());
93     vector<ll> dsu(n + 1, -1);
94     auto qry = [&dsu](auto qry, int x) -> int {
95         return dsu[x] < 0 ? x : dsu[x] = qry(qry, dsu[x]);
96     };
97     auto upd = [&dsu, &qry](int u, int v) -> void {
98         if(dsu[u] = qry(qry, u)) > dsu[v] = qry(qry, v))
99             swap(u, v);
100         dsu[u] += dsu[v];
101         dsu[v] = u;
102     };
103     for(auto &p1, &p2 : g) {
104         auto &w, &idx = p1;
105         auto &u, &v = p2;
106         if(qry(qry, u) != qry(qry, v))
107             upd(u, v), adj[u].pb({v, w}), adj[v].pb({u, w});
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] = max(mx[j][i - 1], mx[par[j][i - 1]][i - 1]);
126     auto lca = [&par, &dep](int u, int v) -> int {
127         if(dep[u] > dep[v]) swap(u, v);
128         for(int i = LG; i--;)
129             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     return par[u][0];
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     return ans;
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 }

```