

Contents

1 DataStructure	1	10 AnotherVersionMath	19
1.1 2DBIT.cpp	1	10.1 CRT(luoguVersion).cpp	19
1.2 DynamicSegmentTree.cpp	1	10.2 PollardRho.cpp	19
1.3 PbdsGpHashTable.cpp	1	10.3 快速冪.cpp	20
1.4 PbdsPriorityQueue.cpp	1	10.4 數論.cpp	20
1.5 PbdsRope.cpp	2	10.5 篩法.cpp	20
1.6 PbdsTree.cpp	2	11 AnotherVersionString	21
1.7 PersistentSegmentTree.cpp	2	11.1 KMP (2).cpp	21
1.8 Treap.cpp	2	11.2 KMP.cpp	21
2 Math	3	11.3 Manacher (2).cpp	21
2.1 CRT.cpp	3	11.4 Manacher.cpp	21
2.2 CountPrimes.cpp	3	11.5 Z.cpp	22
2.3 FFT.cpp	3	12 AnotherVersionGraph	22
2.4 FWT.cpp	4	12.1 Dijkstra.cpp	22
2.5 Gaussian-Jordan.cpp	4	12.2 SCC.cpp	22
2.6 Generator.cpp	4	12.3 cses 有向圖基環樹森林.cpp	22
2.7 Inv.cpp	5	13 AnotherVersionGeometry	23
2.8 Lucas.cpp	5	13.1 DynamicHull.cpp	23
2.9 Matrix.cpp	5	14 AnotherVersionTree	23
2.10 MillerRabin.cpp	7	14.1 LCA.cpp	23
2.11 Mu.cpp	7		
2.12 NTT.cpp	7		
2.13 PollardRho.cpp	7		
2.14 XorBasis.cpp	8		
3 String	8	1. DataStructure	
3.1 Booth.cpp	8	1.1. 2DBIT.cpp	
3.2 KMP.cpp	8		
3.3 LongestPalindrome.cpp	8		
3.4 Z.cpp	9		
4 Graph	9		
4.1 2-SAT(CSES Planets Cycles).cpp	9		
4.2 Dijkstra.cpp	9		
4.3 Dinic.cpp	10		
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	12		
6 Geometry	12		
6.1 164253Version.cpp	12		
6.2 ConvexHull.cpp	12		
6.3 Inside.cpp	13		
6.4 Intersect.cpp	13		
6.5 MinimumEuclideanDistance.cpp	13		
7 Tree	13		
7.1 HeavyLightDecomposition(modify-and-query-on-path).cpp	13		
7.2 LCA.cpp	14		
8 Misc	14		
8.1 BigNum(luoguP1005).cpp	14		
8.2 Tri-search.cpp	15		
9 AnotherVersionDataStructure	15		
9.1 BIT.cpp	15		
9.2 DSU.cpp	15		
9.3 Treap.cpp	15		
9.4 Treap 但可以多個數縮點 (疑似爛的).cpp	17		
9.5 區間插線段單點查詢李超 (是爛的).cpp	18		
9.6 單點修改動態開點線段樹.cpp	18		
9.7 單點修改無懶標線段樹.cpp	18		
9.8 懶標線段樹.cpp	19		
9.9 純直線單點查詢李超.cpp	19		

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(; j < N; j += lb(j))
13         for(int k = i; k < N; k += lb(k)) bit[k][j] += v;
14 }
15 int qry2(int i, int j) {
16     int ans = 0;
17     for(; j; j -= lb(j))
18         for(int k = i; k; k -= lb(k)) ans += bit[k][j];
19     return ans;
20 }
21 int qry(int y1, int x1, int y2, int x2) {
22     return qry2(y2, x2) - qry2(y2, x1 - 1) - qry2(y1 - 1, x2) +
23         qry2(y1 - 1, x1 - 1);
24 }
25 int main() {
26     int n, q, i = 1, j, y, x;
27     for(scanf("%d %d", &n, &q); getchar(), i <= n; ++i)
28         for(j = 1; j <= n; ++j)
29             if(getchar() == '*') upd(i, j, 1);
30     for(q--; q > 0; --q) {
31         scanf("%d", &i);
32         if(i == 1)
33             scanf("%d %d", &i, &j),
34             upd(i, j, 1 - 2 * qry(i, j, i, j));
35         else
36             scanf("%d %d %d %d", &i, &j, &y, &x),
37             printf("%d\n", qry(i, j, y, x));
38     }
39 }

```

1.2. DynamicSegmentTree.cpp

```

1
2 #define int long long
3 using namespace std;
4
5 int n, q;
6 struct node {
7     int data, lson, rson, tag;
8     int rv() { return data + tag; }
9 };
10
11 node tree[20000005];
12 int a[2000005];
13 int now = 1;
14 int mx = 1000000005;

```

```

15 void push(int index) {
17     if(!tree[index].lson) {
19         tree[index].lson = ++now;
21     }
23     if(!tree[index].rson) {
25         tree[index].rson = ++now;
27     }
29     int lson = tree[index].lson;
31     int rson = tree[index].rson;
33     tree[lson].tag += tree[index].tag;
35     tree[rson].tag += tree[index].tag;
37     tree[index].data = tree[index].rv();
39     tree[index].tag = 0;
41 }

43 void modify(int l, int r, int L, int R, int val, int index) {
45     if(l == L && r == R) {
47         tree[index].tag += val;
49         return;
51     }
53     int mid = (l + r) >> 1;
55     push(index);
57     int lson = tree[index].lson;
59     int rson = tree[index].rson;
61     if(R <= mid) {
63         modify(l, mid, L, R, val, lson);
65     } else if(L > mid) {
67         modify(mid + 1, r, L, R, val, rson);
69     } else {
71         modify(l, mid, L, mid, val, lson);
73         modify(mid + 1, r, mid + 1, R, val, rson);
75     }
77     tree[index].data = tree[lson].rv() + tree[rson].rv();
79 }

81 int query(int l, int r, int L, int R, int index) {
83     // cout << L << " " << R << "\n";
85     if(l == L && r == R) {
87         return tree[index].rv();
89     }
91     int mid = (l + r) >> 1;
93     push(index);
95     int lson = tree[index].lson;
97     int rson = tree[index].rson;
99     if(R <= mid) {
101         return query(l, mid, L, R, lson);
103     }
105     if(L > mid) {
107         return query(mid + 1, r, L, R, rson);
109     }
111     return query(l, mid, L, mid, lson) +
112         query(mid + 1, r, mid + 1, R, rson);
113 }

115 signed main() {
117     ios::sync_with_stdio(0);
119     cin.tie(0);
121     cout.tie(0);
123     cin >> n >> q;
125     for(int i = 1; i <= n; i++) {
127         cin >> a[i];
129         modify(1, mx, a[i], a[i], 1, 1);
131     }
133     while(q--) {
135         char mode;
137         int x, y;
139         cin >> mode;
141         if(mode == '?') {
143             cin >> x >> y;
145             cout << query(1, mx, x, y, 1) << "\n";
147         } else {
149             cin >> x >> y;
151             modify(1, mx, a[x], a[x], -1, 1);
153             a[x] = y;
155             modify(1, mx, a[x], a[x], 1, 1);
157         }
159     }
161 }

```

1.3. PbdsGpHashTable.cpp

```

1 using namespace __gnu_pbds;
3 #define ull unsigned ll
5 mt19937 mt(hash<string>("164253_official_beautiful_fruit"));
7 struct myhash {
9     static ull splitmix64(ull x) {
11         x += 0x9e3779b97f4a7c15;
13         x = (x ^ (x >> 30)) * 0xbf58476d1ce4e5b9;
15         x = (x ^ (x >> 27)) * 0x94d049bb133111eb;
17         return x ^ (x >> 31);
19     }

```

```

13 ull operator()(ull x) const {
15     static const ull FIXED_RANDOM =
17         (ull)make_unique<char>().get() ^
19         chrono::high_resolution_clock::now()
21         .time_since_epoch()
23         .count();
25     // static const ull FIXED_RANDOM=mt();
27     // static const ull
29     // FIXED_RANDOM=chrono::steady_clock::now()
31     // .time_since_epoch().count();
33     return splitmix64(x + FIXED_RANDOM);
35 }
37 };
39 gp_hash_table<ull,ull,myhash> gp;
41 gp[x]=y;
43 if(gp.find(x)!=gp.end())cout<<gp[x];
45 gp.count(); //CE
47 */

```

1.4. PbdsPriorityQueue.cpp

```

1 __gnu_pbds::priority_queue<int> pq;
3 /*
5 push(x); //return iterator
7 pop() top() join(pq2) erase(iter) modify(iter,x)
9 */

```

1.5. PbdsRope.cpp

```

1 using namespace __gnu_cxx;
3 /*
5 rope<int> r;
7 r.erase(pos,k); //r=r.[0,pos)+r.[pos+k,r.length());
9 push_back(x) pop_back() insert(pos,x) clear() find(x)
11 lower_bound(all(r),x) upper_bound //same as vector
13 r.length(); //same as .length
15 r.replace(pos,len=r.length(),x); //r.[pos,pos+len)=x;
17 r.substr(pos,x); //return r.[pos,pos+x);
19 rope<char> s="official_beautiful_fruit";
21 cout<<s; //it's legal
23 */

```

1.6. PbdsTree.cpp

```

1 using namespace __gnu_pbds;
3 /*
5 tree<int,null_type,less<int>,rb_tree_tag,
7 tree_order_statistics_node_update> tr;
9 //same as rope<int>, except tr.lower_bound(x) and upper_bound
11 tr.find_by_order(k); //return kth iterator; k=[0,tr.size())
13 //out of this will get tr.end()
15 tr.order_of_key(val); //return rank(val);
17 tr.join(tr2); //merge tr
19 and tr2, tr2.clear() tr.split(const int&r,RBTree&tr2); //<r
21 will in tr, >=r will in tr2
23 */

```

1.7. PersistentSegmentTree.cpp

```

1 // cses Range Queries and Copies
3 using namespace std;
5 #define LL long long
7 #define pii pair<int, int>
9 #define N 200005
11 #define F first
13 #define S second
15 int n, ver = 1;
17 LL a[N];
19 struct Seg {
21     LL v = 0;
23     struct Seg *l = NULL, *r = NULL;
25 }
27 #define M (L + R >> 1)
29 static const void init(Seg *node, int L = 1, int R = n) {
31     if(L == R) {
33         node->v = a[L];
35         return;
37     }
39     node->l = new Seg();
41     init(node->l, L, M);
43     node->r = new Seg();
45     init(node->r, M + 1, R);
47     node->v = node->l->v + node->r->v;
49 }
51 static const void upd(Seg *node, int x, LL v, int L = 1,
53 int R = n) {
55     if(L == R) {
57         node->v = v;
59     }

```

```

        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[rson].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

```

using namespace std;
using i64 = long long;

```

```

i64 count_pi(i64 N) {
5   if(N <= 1) return 0;
   int v = sqrt(N + 0.5);
7   int n_4 = sqrt(v + 0.5);
   int T = min((int)sqrt(n_4) * 2, n_4);
9   int K = pow(N, 0.625) / log(N) * 2;
   K = max(K, v);
11  K = min<i64>(K, N);
   int B = N / K;
13  B = N / (N / B);
   B = min<i64>(N / (N / B), K);

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

23  const auto div = [](i64 n, int d) -> int {
       return double(n) / d;
   };
25  int p;
   for(p = 2; p <= T; ++p)
27  if(s[p] != s[p - 1]) {
       i64 M = N / p;
       int t = v / p, t0 = s[p - 1];
       for(int i = 1; i <= t; ++i) l[i] -= l[i * p] - t0;
       for(int i = t + 1; i <= v; ++i)
31  l[i] -= s[div(M, i)] - t0;
       for(int i = v, j = t; j >= p; --j)
33  for(int l = j * p; i >= l; --i)
           s[i] -= s[j] - t0;
       for(int i = p * p; i <= K; i += p) e[i] = 1;
37  }
   e[1] = 1;
   int cnt = 1;
   vector<int> roughs(B + 1);
41  for(int i = 1; i <= B; ++i)
       if(!e[i]) roughs[cnt++] = i;
   roughs[cnt] = 0x7fffffff;
43  for(int i = 1; i <= K; ++i) w[i] = e[i] + w[i - 1];
   for(int i = 1; i <= K; ++i) s[i] = w[i] - w[i - (i & -i)];

45  const auto query = [&](int x) -> int {
       int sum = x;
       while(x) sum -= s[x], x ^= x & -x;
51  return sum;
   };
53  const auto add = [&](int x) -> void {
       e[x] = 1;
       while(x <= K) ++s[x], x += x & -x;
55  };
   cnt = 1;
   for(; p <= n_4; ++p)
57  if(!e[p]) {
       i64 q = i64(p) * p, M = N / p;
       while(cnt < q) w[cnt] = query(cnt), cnt++;
61  int t1 = B / p, t2 = min<i64>(B, M / q),
           t0 = query(p - 1);
       int id = 1, i = 1;
       for(; i <= t1; i = roughs[++id])
63  l[i] -= l[i * p] - t0;
       for(; i <= t2; i = roughs[++id])
65  l[i] -= query(div(M, i)) - t0;
       for(; i <= B; i = roughs[++id])
67  l[i] -= w[div(M, i)] - t0;
       for(int i = q; i <= K; i += p)
71  if(!e[i]) add(i);
   }
73  while(cnt <= v) w[cnt] = query(cnt), cnt++;

75  vector<int> primes;
   primes.push_back(1);
77  for(int i = 2; i <= v; ++i)
       if(!e[i]) primes.push_back(i);
   l[1] += i64(w[v] + w[n_4] - 1) * (w[v] - w[n_4]) / 2;
81  for(int i = w[n_4] + 1; i <= w[B]; ++i)
       l[1] -= l[primes[i]];
   for(int i = w[B] + 1; i <= w[v]; ++i)
83  l[1] -= query(N / primes[i]);
   for(int i = w[n_4] + 1; i <= w[v]; ++i) {
       int q = primes[i];
85  i64 M = N / q;
       int e = w[M / q];
       if(e <= i) break;
87  l[1] += e - i;
       i64 t = 0;
       int m = w[sqrt(M + 0.5)];
       for(int k = i + 1; k <= m; ++k)
91  t += w[div(M, primes[k])];
       l[1] += 2 * t - (i + m) * (m - i);
93  }
}

```

```

97  return l[1];
}

```

2.3. FFT.cpp

```

1  using namespace std;
2  inline int read() {
   int ans = 0;
   char c = getchar();
   while(!isdigit(c)) c = getchar();
   while(isdigit(c)) {
       ans = ans * 10 + c - '0';
       c = getchar();
   }
   return ans;
}

13 typedef complex<double> comp;
const int MAXN = 1000005;
15 const comp I(0, 1);
const double PI = acos(-1);
17 comp A[MAXN * 3], B[MAXN * 3], tmp[MAXN * 3], ans[MAXN * 3];
void fft(comp F[], int N, int sgn = 1) {
19  if(N == 1) return;
   memcopy(tmp, F, sizeof(comp) * N);
   for(int i = 0; i < N; i++)
21  *(i % 2 ? F + i / 2 + N / 2 : F + i / 2) = tmp[i];
   fft(F, N / 2, sgn), fft(F + N / 2, N / 2, sgn);
   comp *G = F, *H = F + N / 2;
   comp cur = 1, step = exp(2 * PI / N * sgn * I);
   for(int k = 0; k < N / 2; k++) {
23  tmp[k] = G[k] + cur * H[k];
       tmp[k + N / 2] = G[k] - cur * H[k];
       cur *= step;
   }
   memcopy(F, tmp, sizeof(comp) * N);
}

31 int main() {
   int n = read(), m = read(), N = 1 << __lg(n + m + 1) + 1;
   for(int i = 0; i <= n; ++i) A[i] = read();
   for(int i = 0; i <= m; ++i) B[i] = read();
   fft(A, N), fft(B, N);
   for(int i = 0; i < N; ++i) ans[i] = A[i] * B[i];
   fft(ans, N, -1);
   for(int i = 0; i <= n + m; ++i)
41  printf("%d ", int(ans[i].real() / N + 0.1));
   return 0;
}

```

2.4. FWT.cpp

```

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

```

2.5. Gaussian-Jordan.cpp

```

1  #define int long long
2  using namespace std;
3
5  int n;
double a[105][105];
7
// n <= m
9  void gaussian(double a[105][105], int n, int m) {
   int curi = 0;
   for(int j = 0; j < m; j++) {
       int i;
       for(i = curi; i < n; i++) {
           if(a[i][j]) {
               break;
           }
       }
       if(a[i][j] == 0) continue;
       for(int k = 0; k < m; k++) {
           swap(a[i][k], a[curi][k]);
       }
       for(int k = m - 1; k >= j; k--) {
           a[curi][k] /= a[curi][j];
       }
   }
}

```

```

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

```

2.6. 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 <= phix; i++) {
75         if(__gcd(i, x) != 1) continue;
76         bool ok = 1;
77         for(int j : fact) {
78             if(j != phix && f(i, j, x) == 1) {
79                 ok = 0;

```

```

80         break;
81     }
82 }
83 if(ok) {
84     fst = i;
85     break;
86 }
87 }
88 int now = fst;
89 // cout << fst << "\n";
90 for(int i = 1; i <= phix; i++) {
91     if(__gcd(i, phix) == 1) {
92         ret.push_back(now);
93     }
94     now *= fst;
95     now %= x;
96 }
97 return ret;
98 }
99
100 signed main() {
101     ios::sync_with_stdio(0);
102     cin.tie(0);
103     cout.tie(0);
104     init();
105     cin >> t;
106     while(t--) {
107         cin >> n >> d;
108         vector<int> v = findroot(n);
109         sort(v.begin(), v.end());
110         cout << v.size() << "\n";
111         for(int i = 0; i < v.size(); i++) {
112             if(i % d == d - 1) {
113                 cout << v[i] << " ";
114             }
115         }
116         cout << "\n";
117     }
118 }

```

2.7. 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 }

```

2.8. 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 }

```

2.9. Matrix.cpp

```

1  #define int long long
2  using namespace std;

```



```

5 template <class T> T extgcd(T a, T b, T &x, T &y) {
6     if(!b) {
7         x = 1;
8         y = 0;
9         return a;
10    }
11    T ans = extgcd(b, a % b, y, x);
12    y -= a / b * x;
13    return ans;
14 }
15
16 template <class T> T modeq(T a, T b, T p) {
17     T x, y, d = extgcd(a, p, x, y);
18     if(b % d) return 0;
19     return ((b / d * x) % p + p) % p;
20 }
21
22 template <class T> class Matrix {
23     static const T MOD = 1000000007;
24
25 public:
26     vector<vector<T>> v;
27     Matrix(int n, int m, int identity) {
28         v = vector<vector<T>>(n, vector<T>(m, 0));
29         if(identity)
30             for(int i = 0, k = min(n, m); i < k; ++i)
31                 v[i][i] = 1;
32     }
33     Matrix(Matrix &b) { v = b.v; }
34     void in(int l = 0, int m = -1, int u = 0, int n = -1) {
35         if(n < 0) n = v.size();
36         if(m < 0) m = v[0].size();
37         for(int i = u; i < n; ++i)
38             for(int j = l; j < m; ++j) scanf("%lld", &v[i][j]);
39     }
40     Matrix(int n, int m) {
41         v = vector<vector<T>>(n, vector<T>(m, 0));
42         in();
43     }
44     void out(int l = 0, int m = -1, int u = 0, int n = -1) {
45         if(n < 0) n = v.size();
46         if(m < 0) m = v[0].size();
47         for(int i = u; i < n; ++i)
48             for(int j = l; j < m; ++j)
49                 printf("%lld%c", v[i][j], " \n"[j == m - 1]);
50     }
51     Matrix operator=(Matrix &b) {
52         v = b.v;
53         return *this;
54     }
55     Matrix operator+(Matrix &b) {
56         Matrix ans(*this);
57         int n = v.size(), m = v[0].size();
58         for(int i = 0; i < n; ++i)
59             for(int j = 0; j < m; ++j) {
60                 ans.v[i][j] += b.v[i][j];
61                 if(MOD) {
62                     if(ans.v[i][j] < 0)
63                         ans.v[i][j] =
64                             (ans.v[i][j] % MOD + MOD) % MOD;
65                     if(ans.v[i][j] >= MOD) ans.v[i][j] %= MOD;
66                 }
67             }
68         return ans;
69     }
70     Matrix operator*(T x) {
71         Matrix ans(*this);
72         int n = v.size(), m = v[0].size();
73         for(int i = 0; i < n; ++i)
74             for(int j = 0; j < m; ++j) {
75                 ans.v[i][j] += x;
76                 if(MOD) {
77                     if(ans.v[i][j] < 0)
78                         ans.v[i][j] =
79                             (ans.v[i][j] % MOD + MOD) % MOD;
80                     if(ans.v[i][j] >= MOD) ans.v[i][j] %= MOD;
81                 }
82             }
83         return ans;
84     }
85     Matrix operator-(Matrix &b) {
86         Matrix ans(*this);
87         int n = v.size(), m = v[0].size();
88         for(int i = 0; i < n; ++i)
89             for(int j = 0; j < m; ++j) {
90                 ans.v[i][j] -= b.v[i][j];
91                 if(MOD) {
92                     if(ans.v[i][j] < 0)
93                         ans.v[i][j] =
94                             (ans.v[i][j] % MOD + MOD) % MOD;
95                     if(ans.v[i][j] >= MOD) ans.v[i][j] %= MOD;
96                 }
97             }
98         return ans;
99     }
100     Matrix operator-(T x) {
101         Matrix ans(*this);
102         int n = v.size(), m = v[0].size();
103         for(int i = 0; i < n; ++i)
104             for(int j = 0; j < m; ++j) {
105                 ans.v[i][j] -= x;
106                 if(MOD) {
107                     if(ans.v[i][j] < 0)
108                         ans.v[i][j] =
109                             (ans.v[i][j] % MOD + MOD) % MOD;
110                     if(ans.v[i][j] >= MOD) ans.v[i][j] %= MOD;
111                 }
112             }
113         return ans;
114     }
115     Matrix operator+=(Matrix &b) {
116         int n = v.size(), m = v[0].size();
117         for(int i = 0; i < n; ++i)
118             for(int j = 0; j < m; ++j) {
119                 v[i][j] += b.v[i][j];
120                 if(MOD) {
121                     if(v[i][j] < 0)
122                         v[i][j] = (v[i][j] % MOD + MOD) % MOD;
123                     if(v[i][j] >= MOD) v[i][j] %= MOD;
124                 }
125             }
126         return *this;
127     }
128     Matrix operator*(T x) {
129         int n = v.size(), m = v[0].size();
130         for(int i = 0; i < n; ++i)
131             for(int j = 0; j < m; ++j) {
132                 v[i][j] += x;
133                 if(MOD) {
134                     if(v[i][j] < 0)
135                         v[i][j] = (v[i][j] % MOD + MOD) % MOD;
136                     if(v[i][j] >= MOD) v[i][j] %= MOD;
137                 }
138             }
139         return *this;
140     }
141     Matrix operator-(Matrix &b) {
142         int n = v.size(), m = v[0].size();
143         for(int i = 0; i < n; ++i)
144             for(int j = 0; j < m; ++j) {
145                 v[i][j] -= b.v[i][j];
146                 if(MOD) {
147                     if(v[i][j] < 0)
148                         v[i][j] = (v[i][j] % MOD + MOD) % MOD;
149                     if(v[i][j] >= MOD) v[i][j] %= MOD;
150                 }
151             }
152         return *this;
153     }
154     Matrix operator-(T x) {
155         int n = v.size(), m = v[0].size();
156         for(int i = 0; i < n; ++i)
157             for(int j = 0; j < m; ++j) {
158                 v[i][j] -= x;
159                 if(MOD) {
160                     if(v[i][j] < 0)
161                         v[i][j] = (v[i][j] % MOD + MOD) % MOD;
162                     if(v[i][j] >= MOD) v[i][j] %= MOD;
163                 }
164             }
165         return *this;
166     }
167     Matrix operator*(Matrix &b) {
168         int n = v.size();
169         int p = b.v.size();
170         int m = b.v[0].size();
171         Matrix ans(n, m, 0);
172         for(int i = 0; i < n; ++i)
173             for(int k = 0; k < p; ++k)
174                 for(int j = 0; j < m; ++j) {
175                     ans.v[i][j] += v[i][k] * b.v[k][j];
176                     if(MOD) {
177                         if(ans.v[i][j] < 0)
178                             ans.v[i][j] =
179                                 (ans.v[i][j] % MOD + MOD) % MOD;
180                         if(ans.v[i][j] >= MOD)
181                             ans.v[i][j] %= MOD;
182                     }
183                 }
184         return ans;
185     }
186     Matrix operator*(T x) {
187         Matrix ans(*this);
188         int n = v.size(), m = v[0].size();
189         for(int i = 0; i < n; ++i)

```

```

191     for(int j = 0; j < m; ++j) {
192         ans.v[i][j] *= x;
193         if(MOD) {
194             if(ans.v[i][j] < 0)
195                 ans.v[i][j] =
196                     (ans.v[i][j] % MOD + MOD) % MOD;
197             if(ans.v[i][j] >= MOD) ans.v[i][j] %= MOD;
198         }
199     }
200     return ans;
201 }
202 Matrix operator*(Matrix &b) {
203     int n = v.size();
204     int p = b.v.size();
205     int m = b.v[0].size();
206     Matrix ans(n, m, 0);
207     for(int i = 0; i < n; ++i)
208         for(int k = 0; k < p; ++k)
209             for(int j = 0; j < m; ++j) {
210                 ans.v[i][j] += v[i][k] * b.v[k][j];
211                 if(MOD) {
212                     if(ans.v[i][j] < 0)
213                         ans.v[i][j] =
214                             (ans.v[i][j] % MOD + MOD) % MOD;
215                     if(ans.v[i][j] >= MOD)
216                         ans.v[i][j] %= MOD;
217                 }
218             }
219     v = ans.v;
220     return *this;
221 }
222 Matrix operator*(T x) {
223     int n = v.size(), m = v[0].size();
224     for(int i = 0; i < n; ++i)
225         for(int j = 0; j < m; ++j) {
226             v[i][j] *= x;
227             if(MOD) {
228                 if(v[i][j] < 0)
229                     v[i][j] = (v[i][j] % MOD + MOD) % MOD;
230                 if(v[i][j] >= MOD) v[i][j] %= MOD;
231             }
232         }
233     return *this;
234 }
235 Matrix operator/(T x) {
236     Matrix ans(*this);
237     int n = v.size(), m = v[0].size();
238     for(int i = 0; i < n; ++i)
239         for(int j = 0; j < m; ++j) {
240             if(MOD) {
241                 ans.v[i][j] = modeq(x, (T)1, (T)MOD);
242                 if(ans.v[i][j] < 0)
243                     ans.v[i][j] =
244                         (ans.v[i][j] % MOD + MOD) % MOD;
245                 if(ans.v[i][j] >= MOD) ans.v[i][j] %= MOD;
246             } else
247                 ans.v[i][j] /= x;
248         }
249     return ans;
250 }
251 Matrix operator/(T x) {
252     int n = v.size(), m = v[0].size();
253     for(int i = 0; i < n; ++i)
254         for(int j = 0; j < m; ++j) {
255             if(MOD) {
256                 v[i][j] = modeq(x, (T)1, (T)MOD);
257                 if(v[i][j] < 0)
258                     v[i][j] = (v[i][j] % MOD + MOD) % MOD;
259                 if(v[i][j] >= MOD) v[i][j] %= MOD;
260             } else
261                 v[i][j] /= x;
262         }
263     return *this;
264 }
265 Matrix operator%=(T p) {
266     int n = v.size(), m = v[0].size();
267     for(int i = 0; i < n; ++i)
268         for(int j = 0; j < m; ++j)
269             if(v[i][j] >= p) v[i][j] %= p;
270     return *this;
271 }
272 void gaussian() {
273     int curi = 0;
274     int n = v.size();
275     int m = v[0].size();
276     for(int j = 0; j < m; j++) {
277         int i;
278         for(i = curi; i < n; i++) {
279             if(MOD) {
280                 v[i][j] %= MOD;
281             }
282             if(v[i][j]) {
283                 break;

```

```

283     }
284 }
285 if(i >= n) {
286     continue;
287 }
288 if(v[i][j] == 0) continue;
289 for(int k = 0; k < m; k++) {
290     swap(v[i][k], v[curi][k]);
291 }
292 for(int k = m - 1; k >= j; k--) {
293     if(MOD) {
294         v[curi][k] =
295             modeq(v[curi][k], (T)1, (T)MOD);
296         v[curi][k] = (v[curi][k] % MOD + MOD) % MOD;
297     } else
298         v[curi][k] /= v[curi][j];
299 }
300 for(int i = 0; i < n; ++i) {
301     if(i != curi) {
302         for(int k = m - 1; k >= j; k--) {
303             v[i][k] -= v[curi][k] * v[i][j];
304             if(MOD) {
305                 v[i][k] =
306                     (v[i][k] % MOD + MOD) % MOD;
307             }
308         }
309     }
310 }
311 curi++;
312 }
313 }
};

```

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 >= 1; n /= 2) {
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); d /= 2) {
24         ans = 1;
25         fastpow(p, d, x, ans);
26         if(ans == x - 1)
27             return 0;
28         else if(ans != 1)
29             return 1;
30     }
31     return 0;
32 }
33 bool miller_rabin(const uLL x) {
34     if(x == 1) return 0;
35     for(auto e : pri) {
36         if(e >= x) return 1;
37         if(check(x, e)) return 0;
38     }
39     return 1;
40 }
41 }

```

2.11. Mu.cpp

```

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

```

```

15     vis[i * p] = 1;
16     if(i % p == 0) {
17         mu[i * p] = 0;
18         break;
19     } else {
20         mu[i * p] = mu[i] * mu[p];
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 }
21
22 void ntt(int F[], int N, int sgn) {
23     int bit = __lg(N);
24     for(int i = 0; i < N; ++i) {
25         rev[i] = (rev[i >> 1] >> 1) | ((i & 1) << (bit - 1));
26         if(i < rev[i]) swap(F[i], F[rev[i]]);
27     }
28     for(int l = 1, t = 1; l < N; l <= 1, t++) {
29         int step = qpow(G, ((MOD - 1) >> t) * sgn + MOD - 1);
30         for(int i = 0; i < N; i += l << 1) {
31             for(int k = i, cur = 1; k < i + l; ++k) {
32                 int g = F[k], h = (ll)F[k + l] * cur % MOD;
33                 F[k] = (g + h) % MOD;
34                 F[k + l] = ((g - h) % MOD + MOD) % MOD;
35                 cur = (ll)cur * step % MOD;
36             }
37         }
38     }
39     if(sgn == -1) {
40         int invN = qpow(N, MOD - 2);
41         for(int i = 0; i < N; ++i) F[i] = (ll)F[i] * invN % MOD;
42     }
43 }

```

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
17 /*input x, n, p, ans, will modify ans to x ^ n % p
18 the first is x, ans and the second is n, p (LL or __uint128)
19 */
20 uLL pri[7] = {2, 325, 9375, 28178,
21              450775, 9780504, 1795265022}; /*2^64*/
22 // int p[3]={2,7,61};/*2^32*/
23 bool check(const uLL x, const uLL p) {
24     uLL d = x - 1, ans = 1;
25     fastpow(p, d, x, ans);
26     if(ans != 1) return 1;
27     for(; !(d & 1);) {
28         d >>= 1;
29         ans = 1;
30         fastpow(p, d, x, ans);
31         if(ans == x - 1)
32             return 0;
33         else if(ans != 1)
34             return 1;
35     }
36 }

```

```

35     }
36     return 0;
37 }
38 bool miller_rabin(const uLL x) {
39     if(x == 1) return 0;
40     for(auto e : pri) {
41         if(e >= x) return 1;
42         if(check(x, e)) return 0;
43     }
44     return 1;
45 }
46 template <class T> T gcd(T a, T b) {
47     if(!a) return b;
48     if(!b) return a;
49     if(a & b & 1) return gcd(sub(a, b), min(a, b));
50     if(a & 1) return gcd(a, b >> 1);
51     if(b & 1) return gcd(a >> 1, b);
52     return gcd(a >> 1, b >> 1) << 1;
53 }
54 /*gcd(a,b) denote gcd(a, 0) = a*/
55 mt19937 rnd(time(0));
56 template <class T> T f(T x, T c, T mod) {
57     return (((uLL)x) * x % mod + c) % mod;
58 }
59 template <class T> T rho(T n) {
60     T mod = n, x = rnd() % mod, c = rnd() % (mod - 1) + 1,
61     p = 1;
62     for(T i = 2, j = 2, d = x; ++i) {
63         x = f(x, c, mod);
64         p = ((uLL)p) * sub(x, d) % mod;
65         if(i % 127 == 0 && gcd(p, n) != 1) return gcd(p, n);
66         if(i == j) {
67             j <= 1, d = x;
68             if(gcd(p, n) != 1) return gcd(p, n);
69         }
70     }
71 }
72 template <class T> T pollard_rho(T n) {
73     if(miller_rabin(n)) return n;
74     T p = n;
75     while(p == n) p = rho(n);
76     return max(pollard_rho(p), pollard_rho(n / p));
77 }
78 int main() {
79     LL t, n, ans;
80     for(cin >> t; t--;) {
81         cin >> n;
82         ans = pollard_rho(n);
83         if(ans == n)
84             puts("Prime");
85         else
86             printf("%lld\n", ans);
87     }
88 }

```

2.14. XorBasis.cpp

```

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

```


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    }
17    return s.substr(k, s.size() >> 1);
18 }
19 //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) {
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 = (s[i + d[i]] == s[i - d[i]]);
33             }
34             l = i - d[i] + 1;
35         }
```

```

36         r = i + d[i] - 1;
37     } else {
38         int j = l + r - i;
39         if(j - d[j] + 1 > l) {
40             d[i] = d[j];
41         } else {
42             d[i] = r - i + 1;
43             a = i + d[i] < n;
44             b = i - d[i] >= 0;
45             c = (s[i + d[i]] == s[i - d[i]]);
46             while(a && b && c) {
47                 d[i]++;
48                 a = i + d[i] < n;
49                 b = i - d[i] >= 0;
50                 c = (s[i + d[i]] == s[i - d[i]]);
51             }
52             l = i - d[i] + 1;
53             r = i + d[i] - 1;
54         }
55     }
56     // cout << d[i] << " ";
57     if(d[i] > d[ans]) {
58         ans = i;
59     }
60 }
61 for(int i = ans - d[ans] + 1; i < ans + d[ans]; i++) {
62     if(s[i] ^ '0') {
63         cout << s[i];
64     }
65 }
```

3.4. Z.cpp

```
1 #define int long long
2 using namespace std;
3
4 string s, t;
5 int ans = 0;
6
7 int z[2000005];
8
9 signed main() {
10     ios::sync_with_stdio(0);
11     cin.tie(0);
12     cout.tie(0);
13     cin >> s >> t;
14     s = t + '0' + s;
15     int n, m;
16     n = s.size();
17     m = t.size();
18     for(int i = 0, l = 0, r = 0; i < n; i++) {
19         if(z[i - l] < r - i + 1) {
20             z[i] = z[i - l];
21         } else {
22             z[i] = max(r - i + 1, (int)0);
23             while(i + z[i] < n && s[i + z[i]] == s[z[i]]) {
24                 z[i]++;
25             }
26             l = i;
27             r = i + z[i] - 1;
28             if(z[i] == m) {
29                 ans++;
30             }
31         }
32     }
33     cout << ans;
34 }
```

4. Graph

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

```
1 #define int long long
2 using namespace std;
3
4 int n, m;
5 vector<int> v[200005];
6 int d[200005];
7 int low[200005];
8 int cnt = 0;
9 int now = 0;
10 int scc[200005];
11 stack<int> s;
12 int op[200005];
13 vector<int> v2[200005];
14 int ind[200005];
15 queue<int> q;
16 int ans[200005];
```

```

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

```

4.2. Dijkstra.cpp

```

1 vector<pair<int, int>> v[100005], v2[100005];
2 vector<edge> es;
3 int dis1[100005];
4 int dis2[100005];
5 bitset<100005> vis1, vis2;
6
7 void dijkstra(int x, int *dis, vector<pair<int, int>> *v,
8               bitset<100005> &vis) {
9     priority_queue<pair<int, int>, vector<pair<int, int>>,
10                   greater<pair<int, int>>>
11         pq;
12     memset(dis, 0x3f, sizeof(dis1));
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});
    edge.pb({v, u, 0});
}
t = n;
cout << dinic(1);
}

```

4.4. MaximumFlow.cpp

```

1  #define int long long
3  using namespace std;

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

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

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

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

```

4.5. SCC.cpp

```

1  int n, m;
vector<int> v[100005];
3  int d[100005];
int low[100005];
5  int cnt = 0;
stack<int> s;
7  int scc[100005];
int now = 0;

9  void dfs(int x) {
11     d[x] = low[x] = ++cnt;
    s.push(x);
13     for(int i : v[x]) {
        if(scc[i]) continue;
15         if(d[i]) {
            low[x] = min(low[x], d[i]);

```

```

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

```

4.6. VBCC.cpp

```

1  using namespace std;
3  #define pb push_back
#define pii pair<int, int>
5  #define N 100005
vector<int> adj[N], bcc[N];
7  stack<int> st;
int dfn[N], low[N], tag, bc, root;
9  bitset<N> ap;
void dfs(int now, int par = -1) {
11     st.push(now);
    low[now] = dfn[now] = ++tag;
13     int f = 0;
    for(int e : adj[now] | views::reverse) {
15         if(e == par) continue;
        if(!dfn[e]) {
17             dfs(e, now), low[now] = min(low[now], low[e]);
            if(low[e] >= dfn[now]) {
19                 if(++f > 1 || now != root) ap[now] = 1;
                ++bc;
21                 for(; st.top() != now; st.pop())
                    bcc[bc].pb(st.top());
23                 bcc[bc].pb(now);
            }
25         } else
            low[now] = min(low[now], dfn[e]);
27     }
}

29 int main() {
    int n, m, u, v;
31     cin >> n >> m;
    vector<pii> g(m);
33     for(auto &[u, v] : g)
        cin >> u >> v, adj[u].pb(v), adj[v].pb(u);
35     for(root = 1; root <= n; ++root)
        if(!dfn[root]) dfs(root);
37     int ans = 0;
    for(int i : views::iota(1) | views::take(n))
39         if(ap[i]) ++ans;
    cout << ans << "\n";
41     for(int i : views::iota(1) | views::take(n))
        if(ap[i]) cout << i << " ";
43 }

```

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

```

1  #define int long long
3  using namespace std;

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

15 void findcycle(int x) {
17     while(!vis1[x]) {
        vis1[x] = 1;
19         x = a[x];
    }
21     cnt++;
    cycle[x] = cnt;
23     r[x] = 0;
    len[cnt] = 1;
25     int temp = a[x];
    while(temp ^ x) {
27         r[temp] = len[cnt];
        len[cnt]++;
29         cycle[temp] = cnt;
        temp = a[temp];
    }

```

```

    temp = a[temp];
}
}
33 void dfs(int x) {
35     if(vis2[x]) return;
37     vis2[x] = 1;
39     for(int i : v[x]) {
41         dfs(i);
43     }
45 void dfs2(int x) {
47     if(cycle[x] || d[x]) return;
49     dfs2(a[x]);
51     d[x] = d[a[x]] + 1;
53     r[x] = r[a[x]];
55     cycle[x] = cycle[a[x]];
57 }
59 signed main() {
61     ios::sync_with_stdio(0);
63     cin.tie(0);
65     cout.tie(0);
67     cin >> n;
69     for(int i = 1; i <= n; i++) {
71         cin >> a[i];
73         v[i].push_back(a[i]);
75         v[a[i]].push_back(i);
77     }
79     for(int i = 1; i <= n; i++) {
81         if(!vis2[i]) {
83             findcycle(i);
85             dfs(i);
87         }
89     }
91     for(int i = 1; i <= n; i++) {
93         if(!cycle[i] && !r[i]) {
95             dfs2(i);
97         }
99     }
101     for(int i = 1; i <= n; i++) {
103         cout << d[i] + len[cycle[i]] << " ";
105     }
107 }

```

5. DP

5.1. CHO.cpp

```

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

```

5.2. Li-Chao-SegmentTree.cpp

```

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

```

```

3     int y(int x) { return a * x + b; }
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
5 for(int i = 0; i < 20; ++i)
6     for(int j = 0; j < N; ++j)
7         if(!(j >> i & 1)) 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, &b = 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, &b = 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 }

```

```

41 pdd operator*(const pll a, const double x) {
    return {a.F * x, a.S * x};
}
43 pdd operator*(const double x, const pll a) {
    return {a.F * x, a.S * x};
}
45 //沒有標示幾個 vector 的都是對三個點做事，以第一個點為參考點
47 ll len2(pll p) {
    return (ll)p.F * p.F + (ll)p.S * p.S;
}
49 // 1 vector
double len(pll p) { return sqrt((double)len2(p)); }
51 ll cross(pll a, pll b) {
    return (ll)a.F * b.S - (ll)a.S * b.F;
}
53 // 2 vector
ll cross(pll p1, pll p2, pll p3) {
    return cross(p2 - p1, p3 - p1);
}
55 // (b-a) cross (c-a)
ll dot(pll a, pll b, pll c) {
    return (ll)(b.F - a.F) * (c.F - a.F) +
        (ll)(b.S - a.S) * (c.S - a.S);
}
57 // (b-a) dot (c-a)
ll ori(pll p1, pll p2, pll p3) {
    return sign(cross(p1, p2, p3));
}
63 // normalize to {-1,0,1} (b-a) cross (c-a)
bool btw(pll p1, pll p2, pll p3) {
    return ori(p3, p1, p2) == 0 && dot(p3, p1, p2) <= 0;
}
65 // p3 btween p1,p2
bool banana(pll p1, pll p2, pll p3,
    pll p4) { //問兩線段是否香蕉
69 if(btw(p1, p2, p3) || btw(p1, p2, p4) || btw(p3, p4, p1) ||
    btw(p3, p4, p2))
71 return true;
73 return ori(p1, p2, p3) * ori(p1, p2, p4) < 0 &&
    ori(p3, p4, p1) * ori(p3, p4, p2) < 0;
}
75 pdd banana_point(pll p1, pll p2, pll p3,
    pll p4) { //分點，算是無限延伸直線的交點
77 //平行的時候 undefined
79 return cross(p2 - p1, p4 - p1) /
    (double)cross(p2 - p1, p4 - p3) * p3 -
    cross(p2 - p1, p3 - p1) /
81 (double)cross(p2 - p1, p4 - p3) * p4;
}
83 pdd proj(pll p1, pll p2, pll p3) {
    return dot(p1, p2, p3) / (double)len2(p2 - p1) * (p2 - p1);
}
85 double min_dis(pll p1, pll p2,
    pll p3) { // min distance of p3 to segment p1,p2
87 if(dot(p1, p2, p3) < 0 || dot(p2, p1, p3) < 0)
89 return min(len(p3 - p1), len(p3 - p2));
91 return abs(cross(p1, p2, p3)) / len(p2 - p1);
}
93 ll area2(vector<pll> &v) { //傳入一個多邊形照順序的點集
    //起點要出現兩次，回傳兩倍面積
    //注意是兩倍才可以 ll 避免浮點數
95 int n = v.size() - 1;
97 ll ans = 0;
99 for(int i = 0; i < n; ++i) ans += cross(v[i], v[i + 1]);
101 return ans;
}
103 int in_polygon(vector<pll> &v,
    pll p) { //傳入多邊形，起點要出現兩次，回傳
    //{-1:in, 0:on, 1:out}
105 int n = v.size() - 1, ans = 1;
107 for(int i = 0; i < n; ++i)
109 if(btw(v[i], v[i + 1], p)) return 0;
111 for(int i = 0; i < n; ++i)
113 if(banana(v[i], v[i + 1], p, {(ll)2e9 + 7, p.S + 1LL}))
115 ans *= -1;
117 //對於任意 p 到 {W, p.S+1}
119 //的向量中不會有整數點存在，其中需要滿足 {W, p.S+1}
121 //必須很遠，保證在多邊形外
123 return ans;
}
125 void solve() {
127 int n;
129 cin >> n;
131 vector<pll> v(n);
133 for(pll &e : v) cin >> e;
135 v.pb(v[0]);
137 ll ans = area2(v) + 2, ans2 = 0;
139 for(int i = 0; i < n; ++i) {
141 if(v[i].F == v[i + 1].F)
143 ans2 += abs(v[i].S - v[i + 1].S);
145 else if(v[i].S == v[i + 1].S)
147 ans2 += abs(v[i].F - v[i + 1].F);
149 else
151 ans2 += gcd(abs(v[i].F - v[i + 1].F),
153 abs(v[i].S - v[i + 1].S));
155 }
157 cout << (ans - ans2) / 2 << " " << ans2;
159 }
161 int main() {

```

```

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

```

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());
22     vector<pair<T, T>> hull;
23     for(int i = 0; i < 2; i++) {
24         int t = hull.size();
25         for(auto x : v) {
26             while(hull.size() - t >= 2 &&
27                 cross(hull[hull.size() - 1] -
28                     hull[hull.size() - 2],
29                     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;
}

```

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 &&

```



```

    dir(z - w, x - w) * dir(z - w, y - w) == -1) {
        return 1;
    }
    return 0;
}

```

6.5. MinimumEuclideanDistance.cpp

```

1  #define int long long
3  #define pii pair<int, int>
5  using namespace std;
7
9  int n;
10 vector<pair<int, int>> v;
11 set<pair<int, int>> s;
12 int dd = LONG_LONG_MAX;
13
14 int dis(pii x, pii y) {
15     return (x.first - y.first) * (x.first - y.first) +
16            (x.second - y.second) * (x.second - y.second);
17 }
18
19 signed main() {
20     ios::sync_with_stdio(0);
21     cin.tie(0);
22     cout.tie(0);
23     cin >> n;
24     for(int i = 0; i < n; i++) {
25         int x, y;
26         cin >> x >> y;
27         x += 1000000000;
28         v.push_back({x, y});
29     }
30     sort(v.begin(), v.end());
31     int l = 0;
32     for(int i = 0; i < n; i++) {
33         int d = ceil(sqrt(dd));
34         while(l < i && v[l].first - v[i].first > d) {
35             s.erase({v[l].second, v[l].first});
36             l++;
37         }
38         auto x = s.lower_bound({v[i].second - d, 0});
39         auto y = s.upper_bound({v[i].second + d, 0});
40         for(auto it = x; it != y; it++) {
41             dd = min(dd, dis({it->second, it->first}, v[i]));
42         }
43         s.insert({v[i].second, v[i].first});
44     }
45     cout << dd;
46 }

```

7. Tree

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

```

1  #define int long long
3  using namespace std;
5
7  int tree[800005];
9
10 int n, q;
11 int a[200005];
12 int st[200005];
13 int tp[200005];
14 int p[200005];
15 int cnt = 0;
16 int d[200005];
17 int si[200005];
18 vector<int> v[200005];
19 int b[200005];
20
21 void build(int l = 1, int r = n, int index = 1) {
22     if(l == r) {
23         tree[index] = b[l];
24         return;
25     }
26     int mid = (l + r) >> 1;
27     build(l, mid, index << 1);
28     build(mid + 1, r, index << 1 | 1);
29     tree[index] = max(tree[index << 1], tree[index << 1 | 1]);
30 }
31
32 int query(int L, int R, int l = 1, int r = n, int index = 1) {
33     if(L == l && R == r) {
34         return tree[index];
35     }
36     int mid = (l + r) >> 1;
37     if(R <= mid) {
38

```

```

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

```

```

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

```

7.2. LCA.cpp

```

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

```

8. Misc

8.1. BigNum(luoguP1005).cpp

```

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

```

```

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

```

8.2. Tri-search.cpp

```

1  using namespace std;
3  int n;
4  double a[15], x, y;
5
6  double get(double x) {
7      double ret = 0;

```

```

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

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

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

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

```

9. AnotherVersionDataStructure

9.1. BIT.cpp

```

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

```

9.2. DSU.cpp

```

template <class T> class Dsu {
#define N 2000005
public:
    T dsu[N], size[N];
    Dsu(T n) {
        for(; n; --n) dsu[n] = n, size[n] = 1;
    }
    T qry(T x) {
        if(dsu[x] == x) return x;
        return dsu[x] = qry(dsu[x]);
    }
    void merge(T a, T b) {
        a = qry(a);
        b = qry(b);
        if(a == b) return;
        if(size[a] < size[b])

```

```

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

```

9.3. Treap.cpp

```

// treap 模板 洛谷 P3369 【模板】普通平衡树

using namespace std;
#define pnn pair<node *, node *>
#define F first
#define S second
mt19937 mt(hash<string>())("official_beautiful_fruit");
struct node {
    node *l, *r;
    int val, sz;
    int mx, mn, sum;
    int rev_tag, add_tag;
    node(int x)
        : val(x), l(0), r(0), sz(1), rev_tag(0), add_tag(0),
          mx(x), mn(x), sum(x) {}
    node(node *tr)
        : val(tr->val), l(tr->l), r(tr->r), sz(tr->sz),
          rev_tag(tr->rev_tag), add_tag(tr->add_tag),
          mx(tr->mx), mn(tr->mn) {}
    void pull() {
        sz = 1;
        mx = mn = sum = val;
        if(l)
            sz += l->sz, mx = max(mx, l->mx),
            mn = min(mn, l->mn), sum += l->sum;
        if(r)
            sz += r->sz, mx = max(mx, r->mx),
            mn = min(mn, r->mn), sum += r->sum;
    }
    void push() {
        if(rev_tag) swap(l, r);
        if(l) l->add_tag += add_tag, l->rev_tag ^= rev_tag;
        if(r) r->add_tag += add_tag, r->rev_tag ^= rev_tag;
        mx += add_tag;
        mn += add_tag;
        sum += add_tag;
        add_tag = 0;
        rev_tag = 0;
    }
};
void debug(node *tr) {
    if(!tr) return;
    tr->push();
    tr->pull();
    debug(tr->l);
    cout << tr->val << " ";
    debug(tr->r);
}
void debug2(node *tr) {
    if(!tr) return;
    tr->push();
    tr->pull();
    cout << tr->val << " ";
    debug2(tr->l);
    debug2(tr->r);
}
int sz(node *tr) { return tr ? tr->sz : 0; }
node *merge(node *a, node *b) {
    if(!a || !b) return a ? b;
    a->push();
    b->push();
    if(mt() % (sz(a) + sz(b)) < sz(a)) {
        a->r = merge(a->r, b);
        a->pull();
        return a;
    }
    b->l = merge(a, b->l);
    b->pull();
    return b;
}
pnn split(node *tr, int v) { //(-inf,v),(v,inf)
    if(!tr) return {0, 0};
    tr->push();
    if(tr->val <= v) {
        auto [l, r] = split(tr->r, v);
        tr->r = l;
        tr->pull();
        return {tr, r};
    }
    auto [l, r] = split(tr->l, v);
    tr->l = r;
    tr->pull();
    return {l, tr};
}

```

```

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

```

```

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

```

9.4. Treap 但可以多個數縮點 (疑似爛的).cpp

```

1 // treap 模板 洛谷 P3369 【模板】普通平衡树
2
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),
            mx(_val), mn(_val), rev_tag(0), add_tag(0) {}
16     node(node *tr)
17         : val(tr->val), l(tr->l), r(tr->r), sz(tr->sz) {}
18     void pull() {
19         sz = 1;
20         mx = mn = sum = num;
21         if(l)
22             sz += l->sz, mx = max(mx, l->mx),
23             mn = min(mn, l->mn), sum += l->sum;
24         if(r)
25             sz += r->sz, mx = max(mx, r->mx),
26             mn = min(mn, r->mn), sum += r->sum;
27     }
28     void push() {
29         if(rev_tag) swap(l, r);
30         if(l) l->add_tag += add_tag, l->rev_tag ^= rev_tag;
31         if(r) r->add_tag += add_tag, r->rev_tag ^= rev_tag;
32         mx += add_tag;
33         mn += add_tag;
34         sum += add_tag;
35         add_tag = 0;
36         rev_tag = 0;
37     }
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();

```

```

    return a;
}
b->l = merge(a, b->l);
b->pull();
return b;
}
pnn split(node *tr, int v) { //(-inf,v),(v,inf)
    if(!tr) return {0, 0};
    tr->push();
    if(tr->val <= v) {
        auto [l, r] = split(tr->r, v);
        tr->r = l;
        tr->pull();
        return {tr, r};
    }
    auto [l, r] = split(tr->l, v);
    tr->l = r;
    tr->pull();
    return {l, tr};
}
pnn splitsz(node *tr, int k) { //[rk.1,rk.k],[rk.k,rk.n]
    if(!tr || sz(tr) <= k) return {tr, 0};
    tr->push();
    if(k <= sz(tr->l)) {
        auto [l, r] = splitsz(tr->l, k);
        tr->l = r;
        tr->pull();
        return {l, tr};
    } else if(k <= sz(tr->l) + 1) {
        auto r = tr->r;
        tr->r = 0;
        tr->pull();
        return {tr, r};
    } else {
        auto [l, r] = splitsz(tr->r, k - (sz(tr->l) + 1));
        tr->r = l;
        tr->pull();
        return {tr, r};
    }
}
node *insert(node *tr, int val = 0, int num = 1) {
    auto [l, r] = split(tr, val);
    return merge(merge(l, new node(val, num)), r);
}
node *insertkth(node *tr, int k) {
    auto [l, r] = splitsz(tr, k - 1);
    return merge(merge(l, new node()),
        r); // new node 拿來區間操作初始化
}
node *eraseall(node *tr, int v) {
    auto [l, r] = split(tr, v - 1);
    return merge(l, split(r, v).S);
}
node *eraseone(node *tr, int v) {
    auto [l, r] = split(tr, v - 1);
    return merge(l, splitsz(r, 1).S);
}
node *eraskth(node *tr, int k) {
    auto [l, r] = splitsz(tr, k - 1);
    return merge(l, splitsz(r, k).S);
}
}
int rnk(node *tr, int v) {
    if(!tr) return 0;
    if(tr->val <= v) return sz(tr->l) + 1 + rnk(tr->r, v);
    return rnk(tr->l, v);
}
int kth(node *tr, int k) {
    auto [l, x] = splitsz(tr, k - 1);
    auto [m, r] = splitsz(x, 1);
    if(!m) return 0;
    int ans = m->val;
    tr = merge(merge(l, m), r);
    return ans;
}
int count(node *tr, int L, int R) { // count[L,R]
    auto [l, x] = split(tr, L - 1);
    auto [m, r] = split(x, R);
    int ans = m->sum; //看要改啥
    tr = merge(merge(l, m), r);
    return ans;
}
int countkth(node *tr, int L, int R) { // count[rk.L,rk.R]
    auto [l, x] = splitsz(tr, L - 1);
    auto [m, r] = splitsz(x, R - L);
    int ans = m->sum; //看要改啥
    tr = merge(merge(l, m), r);
    return ans;
}
int prev(node *tr, int v) {
    auto [x, r] = split(tr, v - 1);
    auto [l, m] = splitsz(x, sz(x) - 1);
    int ans = m->val;
    tr = merge(merge(l, m), r);
}

```

```

    return ans;
}
int next(node *tr, int v) {
    auto [l, x] = split(tr, v);
    auto [m, r] = splitsz(x, 1);
    int ans = m->val;
    tr = merge(merge(l, m), r);
    return ans;
}
int qry(node *tr, int L, int R) { // qry[L,R]
    auto [l, x] = splitsz(tr, L - 1);
    auto [m, r] = splitsz(x, R);
    int ans = m->sum; //看要改啥
    tr = merge(merge(l, m), r);
    return ans;
}
void modify(node *tr, int L, int R, int v) { // modify[L,R]
    auto [l, x] = splitsz(tr, L - 1);
    auto [m, r] = splitsz(x, R);
    m->val += v;
    m->add_tag += v; //看要改啥
    tr = merge(merge(l, m), r);
}
}
signed main() {
    vector<node *> tr(2);
    int n, m;
    scanf("%lld%lld", &n, &m);
    for(int i = 1, x; i <= n; ++i)
        scanf("%lld", &x), (x) && (tr[1] = insert(tr[1], i, x));
    for(; m--; ) {
        int op = -1, p = -1, x = -1, y = -1;
        scanf("%lld", &op);
        if(!op) {
            scanf("%lld%lld%lld", &p, &x, &y);
            auto [l, tmp] = split(tr[p], x - 1);
            auto [m, r] = split(tmp, y);
            tr[p] = merge(l, r);
            tr.push_back(m);
        } else if(op == 1) {
            scanf("%lld%lld", &p, &x);
            // cout<<kth(tr[x],1)<<"\n";//break;
            auto [l, r] = split(tr[p], kth(tr[x], 1));
            tr[p] = merge(merge(l, tr[x]), r);
        } else
            switch(op) {
                case 2:
                    scanf("%lld%lld%lld", &p, &x, &y);
                    tr[p] = insert(tr[p], y, x);
                    break;
                case 3:
                    scanf("%lld%lld%lld", &p, &x, &y);
                    printf("%lld\n", count(tr[p], x, y));
                    break;
                case 4:
                    scanf("%lld%lld", &p, &x);
                    printf("%lld\n", kth(tr[p], x));
                    break;
            }
    }
}

```

9.5. 區間插線段單點查詢李超 (是爛的).cpp

```

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

```



```

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

```

9.6. 單點修改動態開點線段樹.cpp

```

1 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 };
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 }

```

```

37 int main() {
38     int n, q, i = 1, x;
39     node *root = (node *)malloc(sizeof(struct node));
40     root->val = 0;
41     for(scanf("%d %d", &n, &q); i <= n; ++i)
42         getc(), scanf("%d", &a + i),
43         upd(a[i], 1, 1, MAX, root);
44     // printf("%d %d %d\n", qry(2, 2, 1, n, 1), qry(3, 3, 1, n, 1), qry(5, 5, 1, n, 1), qry(5, 5, 1, n, 1));
45     for(; q--;) {
46         getc();
47         char c = getc();
48         scanf("%d %d", &x, &i);
49         if(c == '!')
50             upd(a[x], -1, 1, MAX, root),
51             a[x] = i, upd(i, 1, 1, MAX, root);
52         else
53             printf("%d\n", qry(x, i, 1, MAX, root));
54     }
55 }

```

9.7. 單點修改無懶標線段樹.cpp

```

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

```

9.8. 懶標線段樹.cpp

```

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

```

```

27 void upd(int l, int r, LL v, int L = 1, int R = n,
28         LL id = 1) {
29     if(l <= L && R <= r) {
30         addtag(id, v, L, R);
31         return;
32     }
33     push(id, L, R);
34     if(r <= M)
35         upd(l, r, v, L, M, lid);
36     else if(M + 1 <= l)
37         upd(l, r, v, M + 1, R, rid);
38     else
39         upd(l, M, v, L, M, lid),
40         upd(M + 1, r, v, M + 1, R, rid);
41     pull(id);
42 }
43 LL qry(int l, int r, int L = 1, int R = n, int id = 1) {
44     if(l <= L && R <= r) return seg[id];
45     push(id, L, R);
46     if(r <= M) return qry(l, r, L, M, lid);
47     if(M + 1 <= l) return qry(l, r, M + 1, R, rid);
48     return qry(l, M, L, M, lid) +
49         qry(M + 1, r, M + 1, R, rid);
50 }
51 } seg;
/*1based 陣列 1based id 區間修改 預設維護區間和 */

```

9.9. 純直線單點查詢李超.cpp

```

1 // luogu P4254 李超
2
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; 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         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) 默認 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), p = ((uLL)p * sub(x, d) % mod);
63         if(i % 127 == 0 && gcd(p, n) != 1) return gcd(p, n);
64         if(i == j) {
65             j <<= 1, d = x;
66             if(gcd(p, n) != 1) return gcd(p, n);
67         }
68     }
69 }
70 template <class T> T pollard_rho(T n) {
71     if(miller_rabin(n)) return n;
72     T p = n;
73     while(p == n) p = rho(n);
74     return max(pollard_rho(p), pollard_rho(n / p));
75 }
76 int main() {
77     LL t, n, ans;
78     for(cin >> t; t--;) {
79         cin >> n;
80         ans = pollard_rho(n);
81         if(ans == n)
82             puts("Prime");
83         else
84             printf("%lld\n", ans);
85     }
86 }

```

10.3. 快速幂.cpp

```

1 template <class T, class POW>

```

```

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

```

10.4. 數論.cpp

```

1 template <class T> T extgcd(T a, T b, T &x, T &y) {
2     if(!b) {
3         x = 1;
4         y = 0;
5         return a;
6     }
7     T ans = extgcd(b, a % b, y, x);
8     y -= a / b * x;
9     return ans;
10 }
11 /*extgcd(a,b,x,y)=ax+by, x 跟 y 是會被修改的參數 */
12 template <class T> T modeq(T a, T b, T p) {
13     T x, y, d = extgcd(a, p, x, y);
14     if(b % d) return 0;
15     return ((b / d * x) % p + p) % p;
16 }
17 /*x=modeq(a,b,n), ax=b(mod n), 0<=x<n
modeq(a,1,n) 相當於求 a 在 mod n 下的逆元 */
18 template <class T> T gcd(T a, T b) {
19     if(!a) return b;
20     if(!b) return a;
21     if(a & b & 1) return gcd(abs(a - b), min(a, b));
22     if(a & 1) return gcd(a, b >> 1);
23     if(b & 1) return gcd(a >> 1, b);
24     return gcd(a >> 1, b >> 1) << 1;
25 }
26 /*gcd(a,b) 默認 gcd(a,0)=a*/
27 ll crt(V<ll> &p, V<ll> &a) {
28     ll n = 1, ans = 0, k = a.size();
29     for(ll &e : p) n *= e;
30     for(int i = 0; i < k; ++i)
31         ans = (ans + a[i] * n / p[i] % n *
32             modeq(n / p[i], 1LL, p[i]) % n) %
33             n;
34     return (ans % n + n) % n;
35 }
36 /*(a+b)^p ≡ a+b ≡ a^p + b^p (mod p) (小費馬)
(p-1)! ≡ -1 (mod p) (威爾遜定理)
\\(v(n):=n 中 p 的幕次, (n)_p:=\\frac{n!}{p^{v(n)}}\\)
\\(s(n):=p 進制下 n 的所有位數和\\)
\\(v(n!)=\\sum_{i=1}^n \\lfloor \\frac{n}{p^i} \\rfloor \\)
\\(\\frac{n!}{p^{v(n)}} \\equiv (-1)^{s(n)} \\pmod{p}\\) (勒壤得定理)
v(\\frac{n!}{p^{v(n)}}) = \\frac{s(n)+s(m-n)-s(m)}{p-1} (庫默爾定理)
\\(v(\\frac{n!}{p^{v(n)}}) \\equiv \\sum_{i=1}^n \\lfloor \\frac{n}{p^i} \\rfloor \\)
\\(\\frac{n!}{p^{v(n)}} \\equiv (-1)^{s(n)} \\pmod{p}\\) (庫默爾定理推廣)
\\(n! \\equiv (-1)^{s(n)} \\pmod{p}\\)
\\(\\frac{n!}{p^{v(n)}} \\equiv (-1)^{s(n)} \\pmod{p}\\)
打階乘表 + 迭代這公式可以 O(p+log_p(n)) (mod 下階乘)
\\(\\frac{n!}{p^{v(n)}} \\equiv (-1)^{s(n)} \\pmod{p}\\)
把 p 從 C(n,m) 裡面隔離掉了 就能用上面的
(n!)_p+ 模逆元 (mod 下階乘推廣至二項式)
((p^q)!)_p ≡ ±1 (mod p^q) (威爾遜定理推廣)
\\(\\frac{n!}{p^{v(n)}} \\equiv (-1)^{s(n)} \\pmod{p}\\)
\\(\\frac{n!}{p^{v(n)}} \\equiv (-1)^{s(n)} \\pmod{p}\\)
(lucas 定理) 打階乘表跟模逆元表 + 迭代這公式可以 O(p+log_p(n))
若 p 進制下任何一位 i 滿足 n_i < m_i 則
\\(\\frac{n!}{p^{v(n)}} \\equiv (-1)^{s(n)} \\pmod{p}\\)
則因 \\(\\frac{n!}{p^{v(n)}} \\equiv (-1)^{s(n)} \\pmod{p}\\)
\\(\\frac{n!}{p^{v(n)}} \\equiv (-1)^{s(n)} \\pmod{p}\\)
設 p=2 則有 \\(\\frac{n!}{p^{v(n)}} \\equiv (-1)^{s(n)} \\pmod{p}\\)
n < m (lucas 定理額外性質) lucas 定理可由此生成函數做法得到
不依賴小費馬 對多項式也成立 根據上述
\\(\\frac{n!}{p^{v(n)}} \\equiv (-1)^{s(n)} \\pmod{p}\\) 可將 k 做唯一質數分解
個別做完再做 crt 得到結果 (exlucas 定理)
卡特蘭數 \\(C(n)=\\frac{1}{n+1} \\binom{2n}{n}\\) 時 C(n)=\\sum_{k=0}^n C(k)C(n-1-k)
\\(\\frac{n!}{p^{v(n)}} \\equiv (-1)^{s(n)} \\pmod{p}\\)
同時 n 對括號的合法放置數即是 C(n) 若有任意 k 種括號可選 則
C(n)k^n
模逆元表 p=i*(p/i)+p%i, -p%i=i*(p/i), inv(i)=-(p/i)*inv(p%i)*/
LL fracp[N], invp[N];
void fracp_init(LL p) {
    fracp[0] = 1;
    for(int i = 1; i < p; ++i) fracp[i] = fracp[i - 1] * i % p;
}
void invp_init(LL p) {

```

```

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

```

10.5. 篩法.cpp

```

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 }
67 #undef N
68 };
69 /*Prime prime(n) 建立打好 1~n 質數表的物件

```

```

prime.list(一個 vector) 是質數表
可修改 define N 決定歐篩/埃篩上限
可在建構子選擇篩法 有歐篩/埃篩/根號暴力搜
73 prime.factorize(n) 用歐篩方式得到 1~n 所有數的最小質因數
可在 factor(一個 vector) 上一路回溯 logn 得到一個數的質因數分解
75 做 n 個數質因數分解共花 nlogn
show() 會以空格隔開 顯示所有 list 內的元素 有尾空格尾換行
77 printf 裡面用%lld 視情況換為%d 或 cout*/

```

11. AnotherVersionString

11.1. KMP (2).cpp

```

1 #define V vector
V<int> kmp(string s) {
3     int n = s.size();
    V<int> f(n);
5     for(int i = 1; i < n; ++i) {
        int j = f[i - 1];
7         for(; j > 0 && s[j] != s[i];) j = f[j - 1];
        f[i] = j + (s[j] == s[i]);
9     }
    return f;
11 }
// 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 {
    #define N 1000005
    public:
        int fail[N], p[N];
        Kmp(char *t, int n) {
            fail[0] = -1;
7             for(int i = 1; i < n; ++i) {
                for(fail[i] = fail[i - 1];
9                 t[i] != t[fail[i] + 1] && fail[i] != -1;)
                    fail[i] = fail[fail[i]];
11                if(t[i] == t[fail[i] + 1]) ++fail[i];
            }
13        }
        void match(char *s, int n, char *t, int m) {
            p[0] = (s[0] == t[0]) - 1;
15            for(int i = 1; i < n; ++i) {
                for(p[i] = p[i - 1];
17                 s[i] != t[p[i] + 1] && p[i] != -1;)
                    p[i] = fail[p[i]];
19                if(s[i] == t[p[i] + 1]) ++p[i];
            }
21        }
23    #undef N
};
/*Kmp kmp(t) 會建好 t 的失配函數 fail[]
25 * 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] : '.')
int ex(string &s, int l, int r, int n) {
3     int i = 0;
    while(l - i >= 0 && r + i < n && T(l - i) == T(r + i)) ++i;
5     return i;
}
7 int manacher(string s, int n) {
    n = 2 * n + 1;
9     int mx = 0;
    int center = 0;
    vector<int> r(n);
11    int ans = 1;
    r[0] = 1;
13    for(int i = 1; i < n; i++) {
        int ii = center - (i - center);
15        int len = mx - i + 1;
        if(i > mx) {
            r[i] = ex(s, i, i, n);
            center = i;
            mx = i + r[i] - 1;
17        } else if(r[ii] == len) {
            r[i] = len + ex(s, i - len, i + len, n);
            center = i;
            mx = i + r[i] - 1;
23        } else {
            r[i] = min(r[ii], len);
25        }
        ans = max(ans, r[i]);
27    }
    return ans - 1;
29 }
31 }

```

11.4. Manacher.cpp

```

1 #define V vector
string manacher(string t) {
3     int n = t.size() << 1 | 1;
    string s(n, '#');
5     for(int i = 0, m = t.size(); i < m; ++i)
        s[i << 1 | 1] = t[i];
7     V<int> p(n);
    for(int i = 0, m = 0, r = 0; i < n; ++i) {
9         p[i] = r > i ? min(r - i, p[m - (i - m)]) : 1;
        for(; i - p[i] >= 0 && i + p[i] < n &&
11             s[i - p[i]] == s[i + p[i]];)
            ++p[i];
13         if(i + p[i] > r) r = i + p[i], m = i;
    }
15    int k = 0;
    string ans = "";
17    for(int i = 0; i < n; ++i)
        if(p[i] > p[k]) k = i;
19    for(int r = k + p[k], l = k - p[k]; ++l < r;)
        if(s[l] != '#') ans += s[l];
21    return ans;
}
23 // manacher(s) 給出 s
25 // 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

```

1 class Z {
    public:
        vector<int> z;
        Z(string s) {
5             z = vector<int>(s.size());
            for(int l = 0, i = 1; i < n; ++i) {
7                 if(l + z[l] >= i)
                    z[i] = min(z[l] + l - i, z[i - l]);
9                 while(i + z[i] < n && s[z[i]] == s[i + z[i]])
                    ++z[i];
11                if(i + z[i] > l + z[l]) l = i;
            }
13        }
};
15 // 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

```

1 // cses Shortest Routes I
2
3 using namespace std;
    #define N 100005
5     #define LL long long
    #define pii pair<int, int>
7     #define pil pair<LL, LL>
    #define F first
9     #define S second
    #define pb push_back
11    #define DE if(1)
    #define INF (LL)1e16
13    vector<pil> adj[N];
    LL d[N];
15    bitset<N> vis;
    int main() {
17        int n, m, u, v;
        LL c;
19        priority_queue<pil, vector<pil>, greater<pil>> q;
        for(cin >> n >> m; m--;)
21            cin >> u >> v >> c, adj[u].pb({v, c});
        q.push({0, 1});
23        d[1] = 0;
        for(u = 2; u <= n; ++u) d[u] = INF;
25        for(; !q.empty(); q.pop()) {
            if(vis[q.top().S]) continue;
27            vis[q.top().S] = 1;
            for(auto &e : adj[q.top().S]) {
29                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});
31                }
            }
33        }
        for(u = 1; u <= n; ++u) printf("%lld ", d[u]);
35    }
}

```

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

```

12.3. cses 有向圖基環樹森林.cpp

```

1 // cses Planets Queries II 基環樹森林模板
2 using namespace std;
3 #define N 200005
4 #define pb push_back
5 // int cyc[i]=1~n 代表 i 屬於哪顆樹
6 // bitset incyc[i]=0/1 代表 i 是否在環上
7 // int len[k]=1~n 代表第 k 棵樹的環長度
8 // int num[i]=1~n 如果 incyc[i] 代表的是在環上的編號
9 // 否則代表的是環上最近的點的編號 int dis[i]=0~n-1
10 // 代表到環上最近點的距離 若 i 在環上則為 0
11 int tag = 1, cyc[N], len[N], num[N], dis[N], nxt[N][19];
12 bitset<N> vis, incyc;
13 vector<int> path;
14 void dfs(int now) {
15     if(vis[now]) {
16         int i = 1;
17         for(int k; k = path.back(), path.pop_back(),
18             k != now && !path.empty(); ++i) {
19             cyc[k] = tag;
20             incyc[k] = 1;
21             num[k] = i;
22         }
23     }
24     cyc[now] = tag;
25     incyc[now] = 1;
26     num[now] = i;
27     len[tag] = i;
28     ++tag;
29     return;
30 }
31 vis[now] = 1;
32 path.pb(now);
33 if(!cyc[nxt[now][0]]) dfs(nxt[now][0]);
34 if(cyc[now]) return;
35 cyc[now] = cyc[nxt[now][0]];
36 num[now] = num[nxt[now][0]];
37 dis[now] = dis[nxt[now][0]] + 1;
38 }
39 int jmp(int a, int x) {
40     for(int k = 19; k--;)
41         for(; 1 <= k <= x; x -= 1 <= k, a = nxt[a][k]);
42     return a;
43 }

```

```

45 }
46 int main() {
47     ios::sync_with_stdio(0);
48     cin.tie(0);
49     cout.tie(0);
50     int n, q, i = 1, u, v;
51     for(cin >> n >> q; i <= n; ++i) cin >> nxt[i][0];
52     for(int k = 1; k < 19; ++k)
53         for(i = 1; i <= n; ++i)
54             nxt[i][k] = nxt[nxt[i][k-1]][k-1];
55     for(i = 1; i <= n; ++i)
56         if(!cyc[i]) path.clear(), dfs(i);
57     for(; q--;) {
58         cin >> u >> v;
59         if(cyc[u] == cyc[v]) {
60             if(incyc[v])
61                 cout << (!incyc[u] ? dis[u] : 0) +
62                     (num[u] - num[v] + len[cyc[u]]) %
63                     len[cyc[u]]
64                     << "\n";
65             else if(num[u] == num[v] && dis[u] >= dis[v] &&
66                 jmp(u, dis[u] - dis[v]) == v)
67                 cout << dis[u] - dis[v] << "\n";
68             else
69                 cout << "-1\n";
70         } else
71             cout << "-1\n";
72     }
73 }

```

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) {

```



```

19     if(dep[a] > dep[b]) swap(a, b);
20     b = jmp(b, dep[b] - dep[a]);
21     if(a == b) return a;
22     for(int i = LG; i--;)
23         for(; par[a][i] != par[b][i]; b = par[b][i])
24             a = par[a][i];
25     return par[a][0];
26 }
27 int main() {
28     int n;
29     cin >> n;
30     for(int i = n, u, v; --i;)
31         cin >> u >> v, g[u].pb(v), g[v].pb(u);
32     dfs();
33     for(int i = 1; i < LG; ++i)
34         for(int j = 1; j <= n; ++j)
35             par[j][i] = par[par[j][i - 1]][i - 1];
36     int k = lca(1, n);
37 }
38 //點編號 1~n, 建的無向圖但改 dfs
39 //就能變有向, 改有向記得邊要反著建 dep[n] 代表 n 的深度 (1
40 // base), par[i][j] 代表 i 往上 1<<j 步的祖先是誰, 不存在則是
41 // 0, sub[i] 代表 i 的子樹大小 jmp(i, j) 代表 i 往上 j
42 // 步的祖先是誰
43 #pragma GCC optimize(
44     "Ofast,fast-math,unroll-loops,no-stack-protector")
45 using namespace std;
46 #define ll long long
47 #define pb push_back
48 #define N 200005
49 #define pii pair<int, int>
50 #define V vector
51 #define inf 1000000000
52 #define M 200005
53 #define LG 18
54 #define pii pair<int, int>
55 #define ppp pair<pii, pii>
56 char buf[1 << 22], *p1, *p2;
57 int p[12];
58 #define gc()
59     (p1 == p2 &&
60         (p2 = (p1 = buf) + fread(buf, 1, 1 << 22, stdin),
61             p1 == p2)
62         ? EOF
63         : *p1++)
64 inline int gi() {
65     int x = 0;
66     for(char c; '0' <= (c = gc()) && c <= '9'; x += c - '0')
67         x *= 10;
68     return x;
69 }
70 inline void pi(int x, char c = ' ') {
71     if(!x) putchar('0');
72     int i = 0;
73     for(; x; x /= 10) p[i++] = x % 10;
74     for(; i--;) putchar(p[i] + '0');
75     putchar(c);
76 }
77 int main() {
78     cin.tie(0) -> sync_with_stdio(0);
79     int n, m, q;
80     cin >> n >> m >> q;
81     vector<ppp> g(m);
82     bitset<M> ans;
83     vector<vector<pii>> adj(n + 1, vector<pii>());
84     for(int i = 0; i < m; ++i) {
85         auto &p1, p2 = g[i];
86         auto &w, idx = p1;
87         auto &u, v = p2;
88         cin >> u >> v >> w;
89         idx = i;
90     }
91     sort(g.begin(), g.end());
92     vector<ll> dsu(n + 1, -1);
93     auto qry = [&dsu](auto qry, int x) -> int {
94         return dsu[x] < 0 ? x : dsu[x] = qry(qry, dsu[x]);
95     };
96     auto upd = [&dsu, &qry](int u, int v) -> void {
97         if(dsu[u] = qry(qry, u)) > dsu[v] = qry(qry, v))
98             swap(u, v);
99         dsu[u] += dsu[v];
100         dsu[v] = u;
101     };
102     for(auto &p1, p2 : g) {
103         auto &w, idx = p1;
104         auto &u, v = p2;
105         if(qry(qry, u) != qry(qry, v))
106             upd(u, v), adj[u].pb({v, w}), adj[v].pb({u, w});
107     }
108     vector<vector<int>> par(n + 1, vector<int>(LG)),
109         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];
130         if(u == v) return u;
131         for(int i = LG; i--;)
132             if(par[u][i] != par[v][i])
133                 u = par[u][i], v = par[v][i];
134         return par[u][0];
135     };
136     auto path = [&par, &mx, &dep](int k, int x) -> int {
137         int ans = 0;
138         for(int i = LG; i--;)
139             if((1 << i) & (dep[x] - dep[k]))
140                 ans = max(ans, mx[x][i]), x = par[x][i];
141         return ans;
142     };
143     for(auto &p1, p2 : g) {
144         auto &w, idx = p1;
145         auto &u, v = p2;
146         int k = lca(u, v);
147         ans[idx] = max(path(k, u), path(k, v)) >= w;
148     }
149     for(int i = 0; i < m; ++i)
150         cout << i << " "
151         << (const char[2][5]){ "NO\n", "YES\n" }[ans[i]];
152     cout << "\n";
153     for(int k; q--;) {
154         cin >> k;
155         int flag = 1;
156         for(int x; k--;) {
157             cin >> x;
158             if(!ans[x - 1]) flag = 0;
159         }
160         cout << (const char[2][5]){ "NO\n", "YES\n" }[flag];
161     }
162 }

```