```
/*
    Don't forget to close:
        ios::sync_with_stdio(false);
        cin.tie(nullptr);
*/
inline __int128 read() {
    __int128 x = 0, f = 1;
```

```cpp
    char ch = getchar();
    while (ch < '0' || ch > '9') {if (ch == '-') f = -1; ch = getchar(); }
    while (ch >= '0' && ch <= '9') {x = x * 10 + ch - '0'; ch = getchar(); }
    return x * f;
}

inline void print(__int128 x) {
    if (x < 0) putchar('-'), x = -x;
    if (x > 9) print(x / 10);
    putchar(x % 10 + '0');
}

template<typename T> inline void read(T &x) {
    x = 0;
    T f = 1;
    char ch = getchar();
    while (!isdigit(ch)) {if (ch == '-') f = -1; ch = getchar(); }
    while (isdigit(ch)) {x = x * 10 + ch - '0'; ch = getchar(); }
    x *= f;
}

template<typename T> inline void write(T x) {
    if (x < 0) putchar('-'), x = -x;
    if (x > 9) write(x / 10);
    putchar(x % 10 + '0');
}





namespace FastIO {
    const int SIZE = 1 << 16;
    char buf[SIZE], str[64];
    int l = SIZE, r = SIZE;
    int read(char *s) {
        while (r) {
            for (; l < r && buf[l] <= ' '; l++);
            if (l < r) break;
            l = 0, r = int(fread(buf, 1, SIZE, stdin));
        }
        int cur = 0;
        while (r) {
            for (; l < r && buf[l] > ' '; l++) s[cur++] = buf[l];
            if (l < r) break;
            l = 0, r = int(fread(buf, 1, SIZE, stdin));
        }
        s[cur] = '\0';
        return cur;
    }
    template<typename type>
    bool read(type &x, int len = 0, int cur = 0, bool flag = false) {
        if (!(len = read(str))) return false;
        if (str[cur] == '-') flag = true, cur++;
        for (x = 0; cur < len; cur++) x = x * 10 + str[cur] - '0';
        if (flag) x = -x;
        return true;
    }
    template <typename type>
    type read(int len = 0, int cur = 0, bool flag = false, type x = 0) {
        if (!(len = read(str))) return false;
        if (str[cur] == '-') flag = true, cur++;
        for (x = 0; cur < len; cur++) x = x * 10 + str[cur] - '0';
        return flag ? -x : x;
    }
} using FastIO::read;
```

```
/*
    Don't forget to init()!
*/
struct Trie {
    static constexpr int MAXBIT = 30;
    int next[N][2], cnt, num[N], sz[N][2];

    void init() {
        memset(next, 0, sizeof(next));
        memset(num, 0, sizeof(num));
        memset(sz, 0, sizeof(sz));
        cnt = 0;
    }

    void insert(int x) {
        int cur = 0;
        for (int i = MAXBIT; ~i; i--) {
            int bit = (x >> i) & 1;
            if (!next[cur][bit]) next[cur][bit] = ++cnt;
            sz[cur][bit]++;
            cur = next[cur][bit];
        }
        num[cur] = x;
    }

    int query(int x) {
        int cur = 0, ans = 0;
        for (int i = MAXBIT; ~i; i--) {
            int bit = (x >> i) & 1;
            if (sz[cur][bit] < (1 << i)) {
                cur = next[cur][bit];
            } else {
                cur = next[cur][bit ^ 1];
                ans |= 1 << i;
            }
            if (!cur) break;
        }
        return ans;
    }
} trie;
```

```
/*
  O(NloglogN)
*/
vector<bool> isprime;
vector<int> prime;

void getPrime(int n) {
    isprime.assign(n + 1, true);
    isprime[0] = isprime[1] = false;
    prime.clear();
    for (int i = 2; i <= n; i++) {
        if (!isprime[i]) continue;
        prime.push_back(i)
        for (int j = i * i; j <= n; j += i) isprime[j] = false;
    }
}

```

```
vector<int> primes;
vector<bool> isprime;

void getPrime(int n){
    isprime.assign(n + 1, true);
    isprime[0] = isprime[1] = false;
    for (int i = 2; i <= n; ++i) {
        if (isprime[i]) primes.emplace_back(i);
        for (auto p : primes) {
            if (i * p > n) break;
            isprime[i * p] = false;
            if (i % p == 0) break;
        }
    }
}
```

```
class dsu {
```

```
public:
    vector<int> p;
    int n;

    dsu(int _n) : n(_n) {
        p.resize(n);
        iota(p.begin(), p.end(), 0);
    }

    inline int get(int x) {
        return (x == p[x] ? x : (p[x] = get(p[x])));
    }

    inline bool unite(int x, int y) {
        x = get(x);
        y = get(y);
        if (x != y) {
            p[x] = y;
            return true;
        }
        return false;
    }
};




struct DSU {
    std::vector<int> f, siz;
    std::vector<bool> g;
    DSU(int n) : f(n), siz(n, 1), g(n) { std::iota(f.begin(), f.end(), 0); }
    int leader(int x) {
        while (x != f[x]) x = f[x] = f[f[x]];
        return x;
    }
    bool same(int x, int y) { return leader(x) == leader(y); }
    bool merge(int x, int y) {
        x = leader(x);
        y = leader(y);
        if (x == y) return false;
        if (g[x] && g[y]) {
            return false;
        }
        siz[x] += siz[y];
        g[x] = g[x] || g[y];
        f[y] = x;
        return true;
    }
    int size(int x) { return siz[leader(x)]; }
};




int f[N], deep[N];

void init(int x) { for (int i = 0; i <= x; i++) f[i] = i; }

int find(int x) { return f[x] == x ? x : f[x] = find(f[x]); }//路径压缩

inline void merge(int x, int y) {
    int u = find(x), v = find(y);
    if (u != v) {
        if (deep[u] > deep[v]) swap(u, v);
        f[u] = v;
        if (deep[u] == deep[v]) deep[y]++;
    }
}
```

```
int max(int a, int b) { return a>b?a:b; }
struct bign {
    int len, s[numlen];
    bign() {
        memset(s, 0, sizeof(s));
        len = 1;
    }
    bign(int num) { *this = num; }
    bign(const char *num) { *this = num; }
    bign operator = (const int num) {
        char s[numlen];
        sprintf(s, "%d", num);
        *this = s;
        return *this;
    }
    bign operator = (const char *num) {
        len = strlen(num);
        while(len > 1 && num[0] == '0') num++, len--;
        for(int i = 0;i < len; i++) s[i] = num[len-i-1] - '0';
        return *this;
    }

    void deal() {
        while(len > 1 && !s[len-1]) len--;
    }

    bign operator + (const bign &a) const {
        bign ret;
        ret.len = 0;
        int top = max(len, a.len) , add = 0;
        for(int i = 0;add || i < top; i++) {
            int now = add;
            if(i < len) now += s[i];
            if(i < a.len)   now += a.s[i];
            ret.s[ret.len++] = now%10;
            add = now/10;
        }
        return ret;
    }
    bign operator - (const bign &a) const {
        bign ret;
        ret.len = 0;
        int cal = 0;
        for(int i = 0;i < len; i++) {
            int now = s[i] - cal;
            if(i < a.len)   now -= a.s[i];
            if(now >= 0)    cal = 0;
            else {
                cal = 1; now += 10;
            }
            ret.s[ret.len++] = now;
        }
        ret.deal();
        return ret;
    }
    bign operator * (const bign &a) const {
        bign ret;
        ret.len = len + a.len;
        for(int i = 0;i < len; i++) {
            for(int j = 0;j < a.len; j++)
                ret.s[i+j] += s[i]*a.s[j];
        }
        for(int i = 0;i < ret.len; i++) {
            ret.s[i+1] += ret.s[i]/10;
            ret.s[i] %= 10;
        }
        ret.deal();
        return ret;
    }

    //乘以小数，直接乘快点
    bign operator * (const int num) {
        bign ret;
        ret.len = 0;
        int bb = 0;
        for(int i = 0;i < len; i++) {
```

```cpp
            int now = bb + s[i]*num;
            ret.s[ret.len++] = now%10;
            bb = now/10;
        }
        while(bb) {
            ret.s[ret.len++] = bb % 10;
            bb /= 10;
        }
        ret.deal();
        return ret;
    }

    bign operator / (const bign &a) const {
        bign ret, cur = 0;
        ret.len = len;
        for(int i = len-1;i >= 0; i--) {
            cur = cur*10;
            cur.s[0] = s[i];
            while(cur >= a) {
                cur -= a;
                ret.s[i]++;
            }
        }
        ret.deal();
        return ret;
    }

    bign operator % (const bign &a) const {
        bign b = *this / a;
        return *this - b*a;
    }

    bign operator += (const bign &a) { *this = *this + a; return *this; }
    bign operator -= (const bign &a) { *this = *this - a; return *this; }
    bign operator *= (const bign &a) { *this = *this * a; return *this; }
    bign operator /= (const bign &a) { *this = *this / a; return *this; }
    bign operator %= (const bign &a) { *this = *this % a; return *this; }

    bool operator < (const bign &a) const {
        if(len != a.len)    return len < a.len;
        for(int i = len-1;i >= 0; i--) if(s[i] != a.s[i])
            return s[i] < a.s[i];
        return false;
    }
    bool operator > (const bign &a) const  { return a < *this; }
    bool operator <= (const bign &a) const { return !(*this > a); }
    bool operator >= (const bign &a) const { return !(*this < a); }
    bool operator == (const bign &a) const { return !(*this > a || *this < a); }
    bool operator != (const bign &a) const { return *this > a || *this < a; }

    string str() const {
        string ret = "";
        for(int i = 0;i < len; i++) ret = char(s[i] + '0') + ret;
        return ret;
    }
};
istream& operator >> (istream &in, bign &x) {
    string s;
    in >> s;
    x = s.c_str();
    return in;
}
ostream& operator << (ostream &out, const bign &x) {
    out << x.str();
    return out;
}
// 大数开平方
bign Sqrt(bign x) {
    int a[numlen/2];
    int top = 0;
    for(int i = 0;i < x.len; i += 2) {
        if(i == x.len-1) {
            a[top++] = x.s[i];
        }
        else
            a[top++] = x.s[i] + x.s[i+1]*10;
    }
    bign ret = (int)sqrt((double)a[top-1]);
    int xx = (int)sqrt((double)a[top-1]);
    bign pre = a[top-1] - xx*xx;
    bign cc;
    for(int i = top-2;i >= 0; i--) {
        pre = pre*100 + a[i];
        cc = ret*20;
        for(int j = 9;j >= 0; j--) {
            bign now = (cc + j)*j;
            if(now <= pre) {
                ret = ret*10 + j;
                pre -= now;
                break;
            }
        }
    }
    return ret;
}
```

```
/*
    小根堆_笛卡尔树 O(n)
    1. 结点——对应于数列元素。即数列中的每个元素都对应于树中某个唯一结点，树结点也对应于数列中的某个唯一元素
    2. 中序遍历（in-order traverse）笛卡尔树即可得到原数列。即任意树结点的左子树结点所对应的数列元素下标比该结点所对应元素的下标小，右子树结点所对应数列元素下标比该结点所对应元素下标大。
    3. 树结构存在堆序性质，即任意树结点所对应数值大 / 小于其左、右子树内任意结点对应数值
*/

int stk[N];

for (int i = 1, top = 0, k; i <= n; i++) {
    k = top;
    while (k > 0 && p[stk[k]] > p[i]) k--;
    if (k) rs[stk[k]] = i;
    if (k < top) ls[i] = stk[k + 1];
    stk[++k] = i;
    top = k;
}
```

```
/*
  Overflow?
*/
ll pre[N][N];

void pre(int n, int m) {
    for (int i = 1; i <= n; i++)
        for (int j = 1; j <= m; j++)
            pre[i][j] = pre[i - 1][j] + pre[i][j - 1] - pre[i - 1][j - 1] + a[i][j];
}

ll calc(int x, int y, int u, int v) {
    return pre[u][v] + pre[x - 1][y - 1] - pre[x - 1][v] - pre[u][y - 1];
}
```

```
#include <bits/stdc++.h>

#define fi first
#define se second
#define mp make_pair
#define pb push_back
#define ls (p << 1)
#define rs (ls | 1)
```

```cpp
#define tm ((tl + tr) >> 1)
#define lowbit(x) ((x) & -(x))

using namespace std;
using ll = long long;
using ull = unsigned long long;
using pii = pair<int, int>;

constexpr double eps = 1e-8;
constexpr int NINF = 0xc0c0c0c0;
constexpr int INF = 0x3f3f3f3f;
constexpr ll LNINF = 0xc0c0c0c0c0c0c0c0;
constexpr ll LINF = 0x3f3f3f3f3f3f3f3f;
constexpr ll mod = 1e9 + 7;
constexpr ll N = 1e6 + 5;

int n;
double a[N][N];

void Gauss_Jordan() {
    for (int i = 1; i <= n; i++) {
        int mx = i;
        for (int j = i + 1; j <= n; j++) if (fabs(a[j][i]) > fabs(a[mx][i])) mx = j;
        if (i != mx) for (int j = 1; j <= n + 1; j++) swap(a[i][j], a[mx][j]);
        if (!a[i][i]) {
            cout << "No Solution\n";
            exit(0);
        }
        for (int j = 1; j <= n; j++) {
            if (j != i) {
                double tmp = a[j][i] / a[i][i];
                for (int k = i + 1; k <= n + 1; k++) a[j][k] -= a[i][k] * tmp;
            }
        }
    }
    for (int i = 1; i <= n; i++) a[i][n + 1] = a[i][n + 1] / a[i][i];
}

int main() {
    ios::sync_with_stdio(0);
    cin.tie(0);
    cin >> n;
    for (int i = 1; i <= n; i++) {
        for (int j = 1; j <= n + 1; j++) {
            cin >> a[i][j];
        }
    }
    Gauss_Jordan();
    for (int i = 1; i <= n; i++)
        cout << fixed << setprecision(2) << a[i][n + 1] << '\n';
    return 0;
}
```

```cpp
#include <bits/stdc++.h>

#define fi first
#define se second
#define mp make_pair
#define pb push_back
#define ls (p << 1)
#define rs (ls | 1)
#define tm ((tl + tr) >> 1)
#define lowbit(x) ((x) & -(x))

using namespace std;
using ll = long long;
using ull = unsigned long long;
using pii = pair<int, int>;

constexpr double eps = 1e-8;
constexpr int NINF = 0xc0c0c0c0;
constexpr int INF = 0x3f3f3f3f;
constexpr ll LNINF = 0xc0c0c0c0c0c0c0c0;
constexpr ll LINF = 0x3f3f3f3f3f3f3f3f;
constexpr ll mod = 1e9 + 7;
constexpr ll N = 1e5 + 5;

ll n, t[N], sum[N], len[N], sz[N], ans[N], T;
vector<pii> G[N];

void dfs(int u, int p) {
    sum[u] = t[u];
    sz[u] = 1;
    for (auto [v, w] : G[u]) {
        if (v == p) continue;
        dfs(v, u);
        sum[u] += sum[v];
        sz[u] += sz[v];
        len[1] += sz[v] * w;
        ans[1] += (sum[v] + sz[v] * t[1]) * w;
    }
}

void dp(int u, int p) {
    for (auto [v, w] : G[u]) {
        if (v == p) continue;
        len[v] = len[u] - w * sz[v] + w * (n - sz[v]);
        ans[v] = ans[u] - w * sum[v] + w * (T - sum[v]) - len[u] * t[u] + len[v] * t[v];
        dp(v, u);
    }
}

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);

    cin >> n;
    for (int i = 1; i <= n; i++) cin >> t[i];
    T = accumulate(t + 1, t + 1 + n, 0);
    for (int i = 1; i < n; i++) {
        int u, v, w;
        cin >> u >> v >> w;
        G[u].push_back({v, w});
        G[v].push_back({u, w});
    }
    dfs(1, 0);
    dp(1, 0);
    for (int i = 1; i <= n; i++) cout << ans[i] << '\n';

    return 0;
}
```

```
#include <bits/stdc++.h>

using namespace std;
#define rep(i, a, n) for (int i=a;i<n;i++)
#define per(i, a, n) for (int i=n-1;i>=a;i--)
#define pb push_back
#define mp make_pair
#define all(x) (x).begin(),(x).end()
#define fi first
#define se second
#define SZ(x) ((int)(x).size())
typedef vector<int> VI;
typedef long long ll;
typedef pair<int, int> PII;
typedef double db;
mt19937 mrand(random_device{}());
const ll mod = 1000000007;

int rnd(int x) { return mrand() % x; }
```

```

ll powmod(ll a, ll b) {
    ll res = 1;
    a %= mod;
    assert(b >= 0);
    for (; b; b >>= 1) {
        if (b & 1)res = res * a % mod;
        a = a * a % mod;
    }
    return res;
}

ll gcd(ll a, ll b) { return b ? gcd(b, a % b) : a; }
// head

typedef double db;
const db EPS = 1e-9;

inline int sign(db a) { return a < -EPS ? -1 : a > EPS; }

inline int cmp(db a, db b) { return sign(a - b); }

struct P {
    db x, y;

    P() {}

    P(db _x, db _y) : x(_x), y(_y) {}

    P operator+(P p) { return {x + p.x, y + p.y}; }

    P operator-(P p) { return {x - p.x, y - p.y}; }

    P operator*(db d) { return {x * d, y * d}; }

    P operator/(db d) { return {x / d, y / d}; }

    bool operator<(P p) const {
        int c = cmp(x, p.x);
        if (c) return c == -1;
        return cmp(y, p.y) == -1;
    }

    bool operator==(P o) const {
        return cmp(x, o.x) == 0 && cmp(y, o.y) == 0;
    }

    db dot(P p) { return x * p.x + y * p.y; }

```

```cpp
    db det(P p) { return x * p.y - y * p.x; }

    db distTo(P p) { return (*this - p).abs(); }

    db alpha() { return atan2(y, x); }

    void read() { cin >> x >> y; }

    void write() { cout << "(" << x << "," << y << ")" << endl; }

    db abs() { return sqrt(abs2()); }

    db abs2() { return x * x + y * y; }

    P rot90() { return P(-y, x); }

    P unit() { return *this / abs(); }

    int quad() const { return sign(y) == 1 || (sign(y) == 0 && sign(x) >= 0); }

    P rot(db an) { return {x * cos(an) - y * sin(an), x * sin(an) + y * cos(an)}; }
};

struct L { //ps[0] -> ps[1]
    P ps[2];

    P &operator[](int i) { return ps[i]; }

    P dir() { return ps[1] - ps[0]; }

    L(P a, P b) {
        ps[0] = a;
        ps[1] = b;
    }

    bool include(P p) { return sign((ps[1] - ps[0]).det(p - ps[0])) > 0; }

    L push() { // push eps outward
        const double eps = 1e-8;
        P delta = (ps[1] - ps[0]).rot90().unit() * eps;
        return {ps[0] + delta, ps[1] + delta};
    }
};

#define cross(p1, p2, p3) ((p2.x-p1.x)*(p3.y-p1.y)-(p3.x-p1.x)*(p2.y-p1.y))
#define crossOp(p1, p2, p3) sign(cross(p1,p2,p3))
```

```cpp

bool chkLL(P p1, P p2, P q1, P q2) {
    db a1 = cross(q1, q2, p1), a2 = -cross(q1, q2, p2);
    return sign(a1 + a2) != 0;
}

P isLL(P p1, P p2, P q1, P q2) {
    db a1 = cross(q1, q2, p1), a2 = -cross(q1, q2, p2);
    return (p1 * a2 + p2 * a1) / (a1 + a2);
}

P isLL(L l1, L l2) { return isLL(l1[0], l1[1], l2[0], l2[1]); }

bool intersect(db l1, db r1, db l2, db r2) {
    if (l1 > r1) swap(l1, r1);
    if (l2 > r2) swap(l2, r2);
    return !(cmp(r1, l2) == -1 || cmp(r2, l1) == -1);
}

bool isSS(P p1, P p2, P q1, P q2) {
    return intersect(p1.x, p2.x, q1.x, q2.x) && intersect(p1.y, p2.y, q1.y, q2.y) &&
           crossOp(p1, p2, q1) * crossOp(p1, p2, q2) <= 0 && crossOp(q1, q2, p1)
                                                             * crossOp(q1, q2, p2) <= 0;
}

bool isSS_strict(P p1, P p2, P q1, P q2) {
    return crossOp(p1, p2, q1) * crossOp(p1, p2, q2) < 0 && crossOp(q1, q2, p1)
                                                            * crossOp(q1, q2, p2) < 0;
}

bool isMiddle(db a, db m, db b) {
    return sign(a - m) == 0 || sign(b - m) == 0 || (a < m != b < m);
}

bool isMiddle(P a, P m, P b) {
    return isMiddle(a.x, m.x, b.x) && isMiddle(a.y, m.y, b.y);
}

bool onSeg(P p1, P p2, P q) {
    return crossOp(p1, p2, q) == 0 && isMiddle(p1, q, p2);
}

bool onSeg_strict(P p1, P p2, P q) {
    return crossOp(p1, p2, q) == 0 && sign((q - p1).dot(p1 - p2)) * sign((q - p2).dot(p1 - p2)) < 0;
}

P proj(P p1, P p2, P q) {
    P dir = p2 - p1;
    return p1 + dir * (dir.dot(q - p1) / dir.abs2());
}

P reflect(P p1, P p2, P q) {
    return proj(p1, p2, q) * 2 - q;
}

db nearest(P p1, P p2, P q) {
    P h = proj(p1, p2, q);
    if (isMiddle(p1, h, p2))
        return q.distTo(h);
    return min(p1.distTo(q), p2.distTo(q));
}

db disSS(P p1, P p2, P q1, P q2) {
    if (isSS(p1, p2, q1, q2)) return 0;
    return min(min(nearest(p1, p2, q1), nearest(p1, p2, q2)), min(nearest(q1, q2, p1), nearest(q1, q2, p2)));
}

db rad(P p1, P p2) {
    return atan2l(p1.det(p2), p1.dot(p2));
}

db incircle(P p1, P p2, P p3) {
    db A = p1.distTo(p2);
    db B = p2.distTo(p3);
    db C = p3.distTo(p1);
    return sqrtl(A * B * C / (A + B + C));
}

//polygon

db area(vector<P> ps) {
    db ret = 0;
    rep(i, 0, ps.size()) ret += ps[i].det(ps[(i + 1) % ps.size()]);
    return ret / 2;
}

int contain(vector<P> ps, P p) { //2:inside,1:on_seg,0:outside
    int n = ps.size(), ret = 0;
    rep(i, 0, n) {
        P u = ps[i], v = ps[(i + 1) % n];
        if (onSeg(u, v, p)) return 1;
        if (cmp(u.y, v.y) <= 0) swap(u, v);
        if (cmp(p.y, u.y) > 0 || cmp(p.y, v.y) <= 0) continue;
        ret ^= crossOp(p, u, v) > 0;
    }
    return ret * 2;
}

vector<P> convexHull(vector<P> ps) {
    int n = ps.size();
    if (n <= 1) return ps;
    sort(ps.begin(), ps.end());
    vector<P> qs(n * 2);
    int k = 0;
    for (int i = 0; i < n; qs[k++] = ps[i++])
        while (k > 1 && crossOp(qs[k - 2], qs[k - 1], ps[i]) <= 0) --k;
    for (int i = n - 2, t = k; i >= 0; qs[k++] = ps[i--])
        while (k > t && crossOp(qs[k - 2], qs[k - 1], ps[i]) <= 0) --k;
    qs.resize(k - 1);
    return qs;
}

vector<P> convexHullNonStrict(vector<P> ps) {
    //caution: need to unique the Ps first
    int n = ps.size();
    if (n <= 1) return ps;
    sort(ps.begin(), ps.end());
    vector<P> qs(n * 2);
    int k = 0;
    for (int i = 0; i < n; qs[k++] = ps[i++])
        while (k > 1 && crossOp(qs[k - 2], qs[k - 1], ps[i]) < 0) --k;
    for (int i = n - 2, t = k; i >= 0; qs[k++] = ps[i--])
        while (k > t && crossOp(qs[k - 2], qs[k - 1], ps[i]) < 0) --k;
    qs.resize(k - 1);
    return qs;
}

db convexDiameter(vector<P> ps) {
    int n = ps.size();
    if (n <= 1) return 0;
    int is = 0, js = 0;
    rep(k, 1, n) is = ps[k] < ps[is] ? k : is, js = ps[js] < ps[k] ? k : js;
    int i = is, j = js;
    db ret = ps[i].distTo(ps[j]);
    do {
        if ((ps[(i + 1) % n] - ps[i]).det(ps[(j + 1) % n] - ps[j]) >= 0)
            (++j) %= n;
        else
```

```
            (++i) %= n;
        ret = max(ret, ps[i].distTo(ps[j]));
    } while (i != is || j != js);
    return ret;
}

vector<P> convexCut(const vector<P> &ps, P q1, P q2) {
    vector<P> qs;
    int n = ps.size();
    rep(i, 0, n) {
        P p1 = ps[i], p2 = ps[(i + 1) % n];
        int d1 = crossOp(q1, q2, p1), d2 = crossOp(q1, q2, p2);
        if (d1 >= 0) qs.pb(p1);
        if (d1 * d2 < 0) qs.pb(isLL(p1, p2, q1, q2));
    }
    return qs;
}

//min_dist

db min_dist(vector<P> &ps, int l, int r) {
    if (r - l <= 5) {
        db ret = 1e100;
        rep(i, l, r) rep(j, l, i) ret = min(ret, ps[i].distTo(ps[j]));
        return ret;
    }
    int m = (l + r) >> 1;
    db ret = min(min_dist(ps, l, m), min_dist(ps, m, r));
    vector<P> qs;
    rep(i, l, r) if (abs(ps[i].x - ps[m].x) <= ret) qs.pb(ps[i]);
    sort(qs.begin(), qs.end(), [](P a, P b) -> bool { return a.y < b.y; });
    rep(i, 1, qs.size()) for (int j = i - 1; j >= 0 && qs[j].y >= qs[i].y - ret; --j)
            ret = min(ret, qs[i].distTo(qs[j]));
    return ret;
}

int type(P o1, db r1, P o2, db r2) {
    db d = o1.distTo(o2);
    if (cmp(d, r1 + r2) == 1) return 4;
    if (cmp(d, r1 + r2) == 0) return 3;
    if (cmp(d, abs(r1 - r2)) == 1) return 2;
    if (cmp(d, abs(r1 - r2)) == 0) return 1;
    return 0;
}

vector<P> isCL(P o, db r, P p1, P p2) {
    if (cmp(abs((o - p1).det(p2 - p1) / p1.distTo(p2)), r) > 0) return {};
```

```cpp
    db x = (p1 - o).dot(p2 - p1), y = (p2 - p1).abs2(), d = x * x - y * ((p1 - o).abs2() - r * r);
    d = max(d, 0.0);
    P m = p1 - (p2 - p1) * (x / y), dr = (p2 - p1) * (sqrt(d) / y);
    return {m - dr, m + dr}; //along dir: p1->p2
}

vector<P> isCC(P o1, db r1, P o2, db r2) { //need to check whether two circles are the same
    db d = o1.distTo(o2);
    if (cmp(d, r1 + r2) == 1) return {};
    if (cmp(d, abs(r1 - r2)) == -1) return {};
    d = min(d, r1 + r2);
    db y = (r1 * r1 + d * d - r2 * r2) / (2 * d), x = sqrt(r1 * r1 - y * y);
    P dr = (o2 - o1).unit();
    P q1 = o1 + dr * y, q2 = dr.rot90() * x;
    return {q1 - q2, q1 + q2};//along circle 1
}

vector<P> tanCP(P o, db r, P p) {
    db x = (p - o).abs2(), d = x - r * r;
    if (sign(d) <= 0) return {}; // on circle => no tangent
    P q1 = o + (p - o) * (r * r / x);
    P q2 = (p - o).rot90() * (r * sqrt(d) / x);
    return {q1 - q2, q1 + q2}; //counter clock-wise
}


vector<L> extanCC(P o1, db r1, P o2, db r2) {
    vector<L> ret;
    if (cmp(r1, r2) == 0) {
        P dr = (o2 - o1).unit().rot90() * r1;
        ret.pb(L(o1 + dr, o2 + dr)), ret.pb(L(o1 - dr, o2 - dr));
    } else {
        P p = (o2 * r1 - o1 * r2) / (r1 - r2);
        vector<P> ps = tanCP(o1, r1, p), qs = tanCP(o2, r2, p);
        rep(i, 0, min(ps.size(), qs.size())) ret.pb(L(ps[i], qs[i])); //c1 counter-clock wise
    }
    return ret;
}

vector<L> intanCC(P o1, db r1, P o2, db r2) {
    vector<L> ret;
    P p = (o1 * r2 + o2 * r1) / (r1 + r2);
    vector<P> ps = tanCP(o1, r1, p), qs = tanCP(o2, r2, p);
    rep(i, 0, min(ps.size(), qs.size())) ret.pb(L(ps[i], qs[i])); //c1 counter-clock wise
    return ret;
}

db areaCT(db r, P p1, P p2) {
    vector<P> is = isCL(P(0, 0), r, p1, p2);
    if (is.empty()) return r * r * rad(p1, p2) / 2;
    bool b1 = cmp(p1.abs2(), r * r) == 1, b2 = cmp(p2.abs2(), r * r) == 1;
    if (b1 && b2) {
        if (sign((p1 - is[0]).dot(p2 - is[0])) <= 0 &&
            sign((p1 - is[0]).dot(p2 - is[0])) <= 0)
            return r * r * (rad(p1, is[0]) + rad(is[1], p2)) / 2 + is[0].det(is[1]) / 2;
        else return r * r * rad(p1, p2) / 2;
    }
    if (b1) return (r * r * rad(p1, is[0]) + is[0].det(p2)) / 2;
    if (b2) return (p1.det(is[1]) + r * r * rad(is[1], p2)) / 2;
    return p1.det(p2) / 2;
}

bool parallel(L l0, L l1) { return sign(l0.dir().det(l1.dir())) == 0; }

bool sameDir(L l0, L l1) { return parallel(l0, l1) && sign(l0.dir().dot(l1.dir())) == 1; }

bool cmp(P a, P b) {
    if (a.quad() != b.quad()) {
        return a.quad() < b.quad();
    } else {
        return sign(a.det(b)) > 0;
    }
}

bool operator<(L l0, L l1) {
    if (sameDir(l0, l1)) {
        return l1.include(l0[0]);
    } else {
        return cmp(l0.dir(), l1.dir());
    }
}

bool check(L u, L v, L w) {
    return w.include(isLL(u, v));
}

vector<P> halfPlaneIS(vector<L> &l) {
    sort(l.begin(), l.end());
    deque<L> q;
    for (int i = 0; i < (int) l.size(); ++i) {
        if (i && sameDir(l[i], l[i - 1])) continue;
```

```cpp
        while (q.size() > 1 && !check(q[q.size() - 2], q[q.size() - 1], l[i])) q.pop_back();
        while (q.size() > 1 && !check(q[1], q[0], l[i])) q.pop_front();
        q.push_back(l[i]);
    }
    while (q.size() > 2 && !check(q[q.size() - 2], q[q.size() - 1], q[0])) q.pop_back();
    while (q.size() > 2 && !check(q[1], q[0], q[q.size() - 1])) q.pop_front();
    vector<P> ret;
    for (int i = 0; i < (int) q.size(); ++i) ret.push_back(isLL(q[i], q[(i + 1) % q.size()]));
    return ret;
}

P inCenter(P A, P B, P C) {
    double a = (B - C).abs(), b = (C - A).abs(), c = (A - B).abs();
    return (A * a + B * b + C * c) / (a + b + c);
}

P circumCenter(P a, P b, P c) {
    P bb = b - a, cc = c - a;
    double db = bb.abs2(), dc = cc.abs2(), d = 2 * bb.det(cc);
    return a - P(bb.y * dc - cc.y * db, cc.x * db - bb.x * dc) / d;
}

P othroCenter(P a, P b, P c) {
    P ba = b - a, ca = c - a, bc = b - c;
    double Y = ba.y * ca.y * bc.y,
            A = ca.x * ba.y - ba.x * ca.y,
            x0 = (Y + ca.x * ba.y * b.x - ba.x * ca.y * c.x) / A,
            y0 = -ba.x * (x0 - c.x) / ba.y + ca.y;
    return {x0, y0};
}

const int N = 201000;
int n, H;
int x[N], y[N];
P p[N];
db ans;

int main() {
    scanf("%d%d", &n, &H);
    rep(i, 1, n + 1) {
        scanf("%d%d", x + i, y + i);
        p[i] = P(x[i], y[i]);
    }
    P eye(x[n], y[n] + H);
    P block(x[n], y[n]);
    per(i, 1, n) {
        bool canseeR = crossOp(eye, block, p[i + 1]) <= 0;
        bool canseeL = crossOp(eye, block, p[i]) <= 0;
        if (canseeL) {
            if (canseeR) ans += p[i].distTo(p[i + 1]);
            else {
                P inter = isLL(eye, block, p[i], p[i + 1]);
                ans += p[i].distTo(inter);
            }
        }
        if (canseeL) block = p[i];
    }
    printf("%.10f\n", ans);
}
```

```
inline ll qpow(ll a, ll b, ll p = mod){
    ll ans = 1;
    a %= p;
    while(b>0){
        if(b & 1) ans = (ans * a) % p;
        a = (a * a) % p;
        b >>= 1;
    }
    return ans;
}
```

```
#include <bits/stdc++.h>

#define fi first
#define se second
#define mp make_pair

using namespace std;
typedef long long ll;
typedef pair<int, int> pii;
const double eps = 1e-8;
const int NINF = 0xc0c0c0c0;
const int INF = 0x3f3f3f3f;
const ll mod = 1e9 + 7;
const ll N = 1e6 + 5;

int n, q, a[N], x, y, l = 1, r, p[N], ans[N], sq, belong[N], num;

struct node {
    int l, r, bel;

    bool operator<(const node &T) const {
        return (belong[l] ^ belong[T.l]) ? belong[l] < belong[T.l] : ((belong[l] & 1) ? r < T.r : r > T.r);
    }
} querry[N];

int cmp(node a, node b) {
    if (a.l / sq != b.l / sq)return a.l / sq < b.l / sq;
    return a.r / sq < b.r / sq;
}

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);
    cin >> n >> q;
    for (int i = 1; i <= n; i++) cin >> a[i];
    for (int i = 1; i <= q; i++) cin >> x >> y, querry[i] = {x, y, i};
    sq = sqrt(n);
    for (int i = 1; i <= n; i++) belong[i] = min(sq, (i - 1) / sq + 1);
    sort(querry + 1, querry + 1 + q);
    for (int i = 1; i <= q; i++) {
        while (querry[i].l < l) num += !p[a[--l]]++;
        while (querry[i].l > l) num -= !--p[a[l++]];
        while (querry[i].r < r) num -= !--p[a[r--]];
        while (querry[i].r > r) num += !p[a[++r]]++;
        ans[querry[i].bel] = num;
    }
    for (int i = 1; i <= q; i++) cout << ans[i] << '\n';
    return 0;
}
```

```
using db = double;

pair<db, db> solve(db X1, db Y1, db X2, db Y2, db X3, db Y3) {
    double X = ((Y2 - Y1) * (Y3 * Y3 - Y1 * Y1 + X3 * X3 - X1 * X1) - (Y3 - Y1) * (Y2 * Y2 - Y1 * Y1 + X2 * X2 - X1 * X1)) / (2.0 * ((X3 - X1) * (Y2 - Y1) - (X2 - X1) * (Y3 - Y1)));
    double Y = ((X2 - X1) * (X3 * X3 - X1 * X1 + Y3 * Y3 - Y1 * Y1) - (X3 - X1) * (X2 * X2 - X1 * X1 + Y2 * Y2 - Y1 * Y1)) / (2.0 * ((Y3 - Y1) * (X2 - X1) - (Y2 - Y1) * (X3 - X1)));
    return make_pair(X, Y);
}
```

```
#include <bits/stdc++.h>

#define fi first
#define se second
#define mp make_pair
#define pb push_back
```

```cpp
#define ls (p << 1)
#define rs (ls | 1)
#define tm ((tl + tr) >> 1)
#define lowbit(x) ((x) & -(x))

using namespace std;
using ll = long long;
using ull = unsigned long long;
using pii = pair<int, int>;

constexpr double eps = 1e-8;
constexpr int NINF = 0xc0c0c0c0;
constexpr int INF = 0x3f3f3f3f;
constexpr ll LNINF = 0xc0c0c0c0c0c0c0c0;
constexpr ll LINF = 0x3f3f3f3f3f3f3f3f;
constexpr ll mod = 1e9 + 7;
constexpr ll N = 1e6 + 5;

int n, a, b, da, db;
int diam, d[N], f[N];
vector<int> G[N];

void dfs(int u, int p) {
    f[u] = 0;
    for (auto v:G[u]) {
        if (v != p) {
            d[v] = d[u] + 1;
            dfs(v, u);
            diam = max(diam, f[u] + f[v] + 1);
            f[u] = max(f[u], f[v] + 1);
        }
    }
}

void solve() {
    cin >> n >> a >> b >> da >> db;
    for (int i = 1; i <= n; i++) G[i].clear();
    for (int i = 1; i < n; i++) {
        int u, v;
        cin >> u >> v;
        G[u].emplace_back(v);
        G[v].emplace_back(u);
    }
    diam = d[a] = 0;
    dfs(a, -1);
    cout << (2 * da >= min(diam, db) || d[b] <= da ? "Alice" : "Bob") << '\n';
}

```

```cpp
int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);
    int T;
    cin >> T;
    while (T--) solve();
    return 0;
}
```

```cpp
#include <bits/stdc++.h>

#define fi first
#define se second
#define mp make_pair
#define pb push_back
#define ls (p << 1)
#define rs (ls | 1)
#define tm ((tl + tr) >> 1)
#define lowbit(x) ((x) & -(x))

using namespace std;
using ll = long long;
using ull = unsigned long long;
using pii = pair<int, int>;

constexpr double eps = 1e-8;
constexpr int NINF = 0xc0c0c0c0;
constexpr int INF = 0x3f3f3f3f;
constexpr ll LNINF = 0xc0c0c0c0c0c0c0c0;
constexpr ll LINF = 0x3f3f3f3f3f3f3f3f;
constexpr ll mod = 1e9 + 7;
constexpr ll N = 1e6 + 5;

int n, a, b, da, db;
int diam, d[N];
vector<int> G[N];

int dfs(int u, int p) {
    int len = 0;
    for (auto v:G[u]) {
        if (v != p) {
            d[v] = d[u] + 1;
            int cur = dfs(v, u) + 1;
            diam = max(diam, cur + len);
            len = max(len, cur);
        }
    }
    return len;
}

void solve() {
    cin >> n >> a >> b >> da >> db;
    for (int i = 1; i <= n; i++) G[i].clear();
    for (int i = 1; i < n; i++) {
        int u, v;
        cin >> u >> v;
        G[u].emplace_back(v);
        G[v].emplace_back(u);
    }
    diam = d[a] = 0;
    dfs(a, -1);
    cout << (2 * da >= min(diam, db) || d[b] <= da ? "Alice" : "Bob") << '\n';
}

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);
    int T;
    cin >> T;
    while (T--) solve();
    return 0;
}
```

```cpp
template <typename T> struct fenwick_tree {
    int n;
    vector<T> t;
    fenwick_tree() : n(0) {}
    fenwick_tree(int n) : n(n), t(n + 1) {}

    void add(int p, T x) {
        while (p <= n) {
            t[p] += x;
            p += lowbit(p);
        }
    }

    T sum(int x) {
        T s = 0;
        while (x > 0) {
            s += t[x];
            x -= lowbit(x);
        }
        return s;
    }

    T sum(int l, int r) {
        return sum(r) - sum(l - 1);
    }
};
```

```
#include <bits/stdc++.h>

#define fi first
#define se second
#define mp make_pair
#define pb push_back
#define ls (p << 1)
#define rs (ls | 1)
#define tm ((tl + tr) >> 1)
#define lowbit(x) ((x) & -(x))

using namespace std;
using ll = long long;
using ull = unsigned long long;
using pii = pair<int, int>;

constexpr double eps = 1e-8;
constexpr int NINF = 0xc0c0c0c0;
constexpr int INF = 0x3f3f3f3f;
constexpr ll LNINF = 0xc0c0c0c0c0c0c0c0;
constexpr ll LINF = 0x3f3f3f3f3f3f3f3f;
constexpr ll mod = 1e9 + 7;
constexpr ll N = 1e6 + 5;

string s;
int f[N][17], n, K;

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);

    cin >> s >> K;
    n = s.size();
    for (auto &c : s) {
        if (isdigit(c)) c -= '0';
        else c -= 'A' - 10;
    }
    int mask = 0;
    for (int i = 0; i < n; i++) {
        for (int j = 1; j <= 16; j++) {
            f[i + 1][j] = (f[i + 1][j] + 1ll * f[i][j] * j % mod) % mod;
            if (j < 16) f[i + 1][j + 1] = (f[i + 1][j + 1] + 1ll * f[i][j] * (16 - j) % mod) % mod;
        }
        f[i + 1][1] = (f[i + 1][1] + 1ll * f[i][0] * 15 % mod) % mod;
        f[i + 1][0] = (f[i + 1][0] + f[i][0]) % mod;
        for (int j = 0; j < s[i]; j++) {
            int MASK = mask;
            if (i || j) MASK |= 1 << j;
            f[i + 1][__builtin_popcount(MASK)]++;
        }
        mask |= 1 << s[i];
    }
    int ans = (f[n][K] + (__builtin_popcount(mask) == K)) % mod;
    cout << ans << '\n';

    return 0;
}
```

```
#include <bits/stdc++.h>

#define fi first
#define se second
#define mp make_pair
#define lowbit(x) ((x) & -(x))

using namespace std;
using ll = long long;
using pii = pair<int, int>;

constexpr double eps = 1e-8;
constexpr int NINF = 0xc0c0c0c0;
constexpr int INF = 0x3f3f3f3f;
constexpr ll mod = 1e9 + 7;
constexpr ll N = 1e6 + 5;

ll f[15][15][2][2], digit;
string s;

ll dfs(int pos, int sum, bool limit, bool zero) {
    ll ans = 0;
    if (pos == s.size()) return sum;
    if (f[pos][sum][limit][zero] != -1) return f[pos][sum][limit][zero];
    for (int i = 0; i <= (limit ? s[pos] - '0' : 9); i++) {
        if (zero && i == 0) {
            ans += dfs(pos + 1, sum, limit && i == s[pos] - '0', true);
        } else {
            ans += dfs(pos + 1, sum + (i == digit), limit && i == s[pos] - '0', false);
        }
    }
    return f[pos][sum][limit][zero] = ans;
}

ll calc(ll x) {
    memset(f, -1, sizeof(f));
    s = to_string(x);
    return dfs(0, 0, true, true);
}

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);

    ll l, r;
    cin >> l >> r;
    for (int i = 0; i <= 9; i++) {
        digit = i;
        cout << calc(r) - calc(l - 1) << " \n"[i == 9];
    }

    return 0;
}
```

```
#include<bits/stdc++.h>

#define fi first
#define se second
#define mp make_pair
using namespace std;
typedef long long ll;
typedef pair<int, int> P;
const double eps = 1e-8;
const int NINF = 0xc0c0c0c0;
const int INF = 0x3f3f3f3f;
const ll mod = 1e9 + 7;
const ll N = 1e5 + 5;

int n, m;
vector<int> G[N];
int a[N], cnt, deep[N], deg[N];

void topsort() {
    queue<int> q;
    for (int i = 1; i <= n; i++) if (!deg[i]) q.push(i), deep[i] = 1;
    while (q.size()) {
        int p = q.front();
        q.pop();
        a[++cnt] = p;
```

```cpp
        for (auto c:G[p]) {
            deep[c] = deep[p] + 1;
            if (!--deg[c]) q.push(c);
        }
    }
}

int main() {
    ios::sync_with_stdio(false);
    cin.tie(0);
    cin >> n >> m;
    for (int i = 1; i <= m; i++) {
        int u, v;
        cin >> u >> v;
        G[u].push_back(v);
        ++deg[v];
    }
    topsort();
    for (int i = 1; i <= n; i++) {
        cout << deep[i] << '\n';
    }
    return 0;
}
```

```cpp
int deg[N];
vector<int> top_idx;

bool topsort(int n) {
    queue<int> q;
    for (int i = 1; i <= n; i++) if (!deg[i]) q.push(i);
    while (!q.empty()) {
        int u = q.front();
        q.pop();
        top_idx.push_back(u);
        for (auto v : G[u]) if (!--deg[v]) q.push(v);
    }
    return top_idx.size() == n;
}
```

```cpp
template<typename T>
T exgcd(T a, T b, T &x, T &y) {
    if (!b) { x = 1, y = 0; return a; }
    T d = exgcd(b, a % b, x, y);
    T z = x;
    x = y, y = z - y * (a / b);
    return d;
}

```

```
template<typename T>
T inv(T a, T p) {
    T x, y;
    exgcd(a, p, x, y);
    return (x % p + p) % p;
}

/*
    线性求逆元
*/

vector<int> inverse(int n, int p) {
    vector<int> inv(n + 1);
    inv[1] = 1;
    for (int i = 2; i <= n; i++) {
        inv[i] = 1ll * (p - p / i) * inv[p % i] % p;
    }
    return inv;
}
```

```
ll fac[N + 5], ifac[N + 5];

ll qpow(ll a, ll b) {
    ll res = 1;
    while (b > 0) {
        if (b & 1) res = res * a % mod;
        a = a * a % mod;
        b >>= 1;
    }
    return res;
}

inline ll perm(ll x, ll y) { return y > x || y < 0 ? 0 : fac[x] * ifac[x - y] % mod; }

inline ll comb(ll x, ll y) { return y > x || y < 0 ? 0 : perm(x, y) * ifac[y] % mod; }

void init() {
    fac[0] = 1;
    for (int i = 1; i <= N; i++) fac[i] = 1ll * i * fac[i - 1] % mod;
    ifac[N] = qpow(fac[N], mod - 2);
    for (int i = N; i; i--) ifac[i - 1] = 1ll * i * ifac[i] % mod;
}
```

```
#include<cstdio>
#include<cmath>
#include<algorithm>
```

```cpp

using namespace std;
const double eps = 1e-3;

int n;

struct Point {
    double x;
    double y;
    double z;

    Point(double _x = 0.0, double _y = 0.0, double _z = 0.0) {
        x = _x;
        y = _y;
        z = _z;
    }
};

Point points[105];

inline double dis(Point a, Point b) {
    double norm = (a.x - b.x) * (a.x - b.x) + (a.y - b.y) * (a.y - b.y) + (a.z - b.z) * (a.z - b.z);
    return sqrt(norm);
}

double solve() {
    double T = 1000.0; //初始温度
    double rate = 0.99999; //温度下降系数
    Point ans_p;    //初始圆心
    int cur;
    Point max_p; //距离圆心最远的点
    double ans = 1e99;
    while (T > eps) //模拟降温
    {
        double max_dis = 0.0; //选定圆心到最远点距离
        for (int i = 1; i <= n; ++i) { //最远的点即points[cur]
            if (dis(ans_p, points[i]) > max_dis) {
                max_dis = dis(ans_p, points[i]);
                cur = i;
            }
        }
        ans = min(ans, max_dis);
        ans_p.x += (points[cur].x - ans_p.x) * (T / 1000.0);
        ans_p.y += (points[cur].y - ans_p.y) * (T / 1000.0);
        ans_p.z += (points[cur].z - ans_p.z) * (T / 1000.0);
        T *= rate;
    }
    return ans;
}

int main() {
    scanf("%d", &n);
    for (int i = 1; i <= n; ++i) {
        scanf("%lf%lf%lf", &points[i].x, &points[i].y, &points[i].z);
    }
    printf("%.15f\n", solve());
    return 0;
}
```

```cpp
#include <bits/stdc++.h>
#include <algorithm>

using namespace std;
constexpr int N = 5000 + 5;

struct node {
    double x, y, z;
} a[N];

struct vec {
    double x, y;

    vec(const double &x0 = 0, const double &y0 = 0) : x(x0), y(y0) {}

    vec operator+(const vec &t) const { return vec(x + t.x, y + t.y); }

    vec operator-(const vec &t) const { return vec(x - t.x, y - t.y); }

    vec operator*(const double &t) const { return vec(x * t, y * t); }

    vec operator/(const double &t) const { return vec(x / t, y / t); }

    const double len2() const { return x * x + y * y; }

    const double len() const { return sqrt(len2()); }

    vec norm() const { return *this / len(); }

    vec rotate_90_c() { return vec(y, -x); }
};

double dot(const vec &a, const vec &b) { return a.x * b.x + a.y * b.y; }

double crs(const vec &a, const vec &b) { return a.x * b.y - a.y * b.x; }

```

```
vec lin_lin_int(const vec &p0, const vec &v0, const vec &p1, const vec &v1) {
    double t = crs(p1 - p0, v1) / crs(v0, v1);
    return p0 + v0 * t;
}

vec circle(const vec &a, const vec &b, const vec &c) {
    return lin_lin_int((a + b) / 2, (b - a).rotate_90_c(), (a + c) / 2, (c - a).rotate_90_c());
}

int n;
vec pot[100005];

double work() {
    random_shuffle(pot + 1, pot + n + 1);
    vec o;
    double r2 = 0;
    for (int i = 1; i <= n; i++) {
        if ((pot[i] - o).len2() > r2) {
            o = pot[i], r2 = 0;
            for (int j = 1; j < i; j++) {
                if ((pot[j] - o).len2() > r2) {
                    o = (pot[i] + pot[j]) / 2, r2 = (pot[j] - o).len2();
                    for (int k = 1; k < j; k++) {
                        if ((pot[k] - o).len2() > r2) {
                            o = circle(pot[i], pot[j], pot[k]), r2 = (pot[k] - o).len2();
                        }
                    }
                }
            }
        }
    }
    return 2.0 * sqrt(r2);
}

int main() {
    scanf("%d", &n);
    for (int i = 1; i <= n; i++) cin >> a[i].x >> a[i].y >> a[i].z;
    for (int i = 1; i <= n; i++) {
        pot[i].x = a[i].x;
        pot[i].y = a[i].y;
    }
    double ans = work();
    for (int i = 1; i <= n; i++) {
        pot[i].x = a[i].x;
        pot[i].y = a[i].z;
    }
```

```cpp
    ans = min(ans, work());
    for (int i = 1; i <= n; i++) {
        pot[i].x = a[i].y;
        pot[i].y = a[i].z;
    }
    ans = min(ans, work());
    printf("%.10lf\n", ans);
    return 0;
}
```

```cpp
#include <bits/stdc++.h>

#define fi first
#define se second
#define mp make_pair

using namespace std;
typedef long long ll;
typedef pair<int, int> pii;
const double eps = 1e-8;
const int NINF = 0xc0c0c0c0;
const int INF = 0x3f3f3f3f;
const ll mod = 1e9 + 7;
const ll N = 1e6 + 5;

int n, A[N];

int Min_show() {
    int i = 0, j = 1, k = 0;
    while (i < n and j < n and k < n) {
        if (A[(i + k) % n] == A[(j + k) % n]) k++;
        else {
            A[(i + k) % n] > A[(j + k) % n] ? i += k + 1 : j + k + 1;
            if (i == j) i++;
            k = 0;
        }
    }
    return min(i, j);
}

int Max_show() {
    int i = 0, j = 1, k = 0;
    while (i < n and j < n and k < n) {
        if (A[(i + k) % n] == A[(j + k) % n]) k++;
        else {
            A[(i + k) % n] < A[(j + k) % n] ? i += k + 1 : j + k + 1;
            if (i == j) i++;
            k = 0;
        }
    }
    return min(i, j);
}

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);
    cin >> n;
    for (int i = 0; i < n; i++) {
        cin >> A[i];
    }
    int s = Min_show();
    for (int i = 0; i < n; i++) cout << A[(s + i) % n] << " \n"[i == n - 1];
    return 0;
}
```

```
#include <bits/stdc++.h>

#define fi first
#define se second
#define mp make_pair

using namespace std;
typedef long long ll;
typedef pair<int, int> pii;
const double eps = 1e-8;
const int NINF = 0xc0c0c0c0;
const int INF = 0x3f3f3f3f;
const ll mod = 1e9 + 7;
const ll N = 1e5 + 5;

int n, m;

struct BIT {
    int bit[N], mi[N], ma[N];

    void add(int x, int v) {
        bit[x] = v;
        for (; x <= n; x += x & -x) {
            mi[x] = ma[x] = v;
            for (int i = 1; i < (x & -x); i <<= 1)
                mi[x] = min(mi[x], mi[x - i]),
                        ma[x] = max(ma[x], ma[x - i]);
        }
    }

    int difference(int l, int r) {
        int L = bit[r], R = bit[r];
        while (l <= r) {
            L = min(L, bit[r]);
            R = max(R, bit[r]);
            for (--r; r - l >= (r & -r); r -= (r & -r))
                L = min(L, mi[r]),
                        R = max(R, ma[r]);
        }
        return R - L + 1;
    }
};

BIT a;

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);
    cin >> n >> m;
    for (int i = 1; i <= n; i++) {
        int x;
        cin >> x;
        a.add(i, x);
    }
    while (m--) {
        int t, l, r;
        cin >> t >> l >> r;
        if (t == 1) a.add(l, r);
        else cout << (a.difference(l, r) == r - l + 1 ? "YES" : "NO") << '\n';
    }
    return 0;
}
```

```
typedef long long ll;
#define sz(x) ((ll)x.size())
#define pb push_back
const ll mod = 1e9 + 7;

ll fpow(ll a, ll b) {
    ll res = 1;
    a %= mod;
    assert(b >= 0);
    for (; b; b >>= 1) {
        if (b & 1)res = res * a % mod;
        a = a * a % mod;
    }
    return res;
```

```
}

namespace linear_seq {
    const ll maxn = 10010;
    ll res[maxn], base[maxn], c[maxn], md[maxn];
    vector<long long> v;

    void mul(ll *a, ll *b, ll k) {
        for (int i = 0; i < k + k; ++i) c[i] = 0;
        for (int i = 0; i < k; ++i)
            if (a[i])
                for (int j = 0; j < k; ++j)
                    c[i + j] = (c[i + j] + a[i] * b[j]) % mod;
        for (ll i = k + k - 1; i >= k; i--)
            if (c[i])
                for (int j = 0; j < v.size(); ++j)
                    c[i - k + v[j]] = (c[i - k + v[j]] - c[i] * md[v[j]]) % mod;
        for (int i = 0; i < k; ++i) a[i] = c[i];
    }

    ll solve(ll n, vector <ll> a, vector <ll> b) {
        ll ans = 0, cnt = 0, k = a.size();
        for (int i = 0; i < k; ++i) md[k - 1 - i] = -a[i];
        md[k] = 1;
        v.clear();
        for (int i = 0; i < k; ++i) if (md[i] != 0) v.push_back(i);
        for (int i = 0; i < k; ++i) res[i] = base[i] = 0;
        res[0] = 1;
        while ((1ll << cnt) <= n) cnt++;
        for (ll p = cnt; p >= 0; p--) {
            mul(res, res, k);
            if ((n >> p) & 1) {
                for (ll i = k - 1; i >= 0; --i) res[i + 1] = res[i];
                res[0] = 0;
                for (int j = 0; j < v.size(); ++j) res[v[j]] = (res[v[j]] - res[k] * md[v[j]]) % mod;
            }
        }
        for (int i = 0; i < k; ++i) ans = (ans + res[i] * b[i]) % mod;
        if (ans < 0) ans += mod;
        return ans;
    }

    vector <ll> bm(vector <ll> s) {
        vector <ll> C(1, 1), B(1, 1);
        ll k = 0, m = 1, b = 1;
        for (int n = 0; n < s.size(); ++n) {
            ll d = 0;
```

```cpp
            for (int i = 0; i <= k; ++i) d = (d + (ll) C[i] * s[n - i]) % mod;
            if (!d) ++m;
            else if (k + k <= n) {
                vector <ll> T = C;
                ll c = mod - d * fpow(b, mod - 2) % mod;
                while (sz(C) < sz(B) + m) C.pb(0);
                for (int i = 0; i < B.size(); ++i) C[i + m] = (C[i + m] + c * B[i]) % mod;
                k = n + 1 - k;
                B = T;
                b = d;
                m = 1;
            } else {
                ll c = mod - d * fpow(b, mod - 2) % mod;
                while (sz(C) < sz(B) + m) C.pb(0);
                for (int i = 0; i < B.size(); ++i) C[i + m] = (C[i + m] + c * B[i]) % mod;
                ++m;
            }
        }
        return C;
    }

    ll gao(vector <ll> a, ll n) {
        vector <ll> c = bm(a);
        c.erase(c.begin());
        for (int i = 0; i < c.size(); ++i) c[i] = (mod - c[i]) % mod;
        return solve(n, c, vector<ll>(a.begin(), a.begin() + sz(c)));
    }
};

class Solution {
public:
    /**
     * 返回c[n]%1000000007的值
     * @param n long长整型 即题目中的n
     * @return int整型
     */
    int Answerforcn(long long n) {
        vector <ll> a(100), b(100);
        a[0] = 2, a[1] = 6;
        b[0] = 7, b[1] = 35;
        for (int i = 2; i < 100; ++i) {
            a[i] = (a[i - 1] * 2 + 3 * a[i - 2]) % mod;
            b[i] = (b[i - 1] * 3 + 10 * b[i - 2]) % mod;
        }
        vector <ll> c(100);
        for (int i = 0; i < 100; ++i) c[i] = a[i] * b[i] % mod;
        return linear_seq::gao(c, n - 1);
    }
};
```

```
ll qpow(ll a, ll b, ll p = mod) {
    ll res = 1;
    a %= p;
    while (b > 0) {
        if (b & 1) res = res * a % p;
        a = a * a % p;
        b >>= 1;
    }
    return res;
}

struct ChothollyTree {
    map <int, ll> odt;

    auto split(int x) {
        auto it = prev(odt.upper_bound(x));
        if (it->first != x) {
            odt[x] = it->second;
            ++it;
        }
        return it;
    }

    void add(int l, int r, ll v) {
        auto end = split(r + 1);
        for (auto it = split(l); it != end; it++)
            it->second += v;
    }

    void assign(int l, int r, ll v) {
        auto end = split(r + 1), begin = split(l);
        odt.erase(begin, end);
        odt[l] = v;
    }

    ll kth(int l, int r, int k) {
        auto itr = split(r + 1), itl = split(l);
        vector<pair<ll,ll> > V;
        for (int R; itl != itr;) {
            R = (itr--)->first;
            V.emplace_back(itr->second, R - itr->first);
        }
        sort(V.begin(), V.end());
        for (auto [x, y] : V) {
            k -= y;
            if (k <= 0) return x;
        }
    }

    ll sum(int l, int r) {
        ll res = 0;
        auto itr = split(r + 1), itl = split(l);
        for (int R; itl != itr;) {
            R = (itr--)->first;
            res = (res + itr->second * (R - itr->first));
        }
        return res;
    }

    ll power(int l, int r, ll x, ll y) {
        ll res = 0;
        auto itr = split(r + 1), itl = split(l);
        for (int R; itl != itr;) {
            R = (itr--)->first;
            res = (res + qpow(itr->second, x, y) * (R - itr->first)) % y;
        }
        return res;
    }
};
```

```
struct node {
    int l, r;
    mutable ll v;
    node(const int &l, const int &r, const ll&v) : l(l), r(r), v(v) {}
    inline bool operator < (const node &T) const {
        return l < T.l;
    }
};

set<node> odt;

auto split(int x) {
    if (x > n) return odt.end();
    auto it = prev(odt.upper_bound({x, 0, 0}));
    if (it->l == x) return it;
    int l = it->l, r = it->r;
    ll v = it->v;
    odt.erase(it);
    odt.insert({l, x - 1, v});
    return odt.insert({x, r, v}).first;
```

```
}

void assign(int l, int r, ll v) {
    auto itr = split(r + 1), itl = split(l);
    odt.erase(itl, itr);
    odt.insert({l, r, v});
}

void add(int l, int r, ll v) {
    auto end = split(r + 1);
    for (auto it = split(l); it != end; it++) {
        it->v += v;
    }
}

ll kth(int l, int r, int k) {
    auto itr = split(r + 1);
    auto itl = split(l);
    vector<pair<ll, ll>> V;
    while (itl != itr) {
        V.push_back({itl->v, itl->r - itl->l + 1});
        itl++;
    }
    sort(V.begin(), V.end());
    for (auto [x, y] : V) {
        k -= y;
        if (k <= 0) return x;
    }
}

ll sum(int l, int r) {
    ll ans = 0;
    auto end = split(r + 1);
    for (auto it = split(l); it != end; it++) {
        ans += it->v;
    }
    return ans;
}

/*
i = l to r
ans = sum ai ^ x (mod y)
*/
ll power(int l, int r, ll x, ll y) {
    ll ans = 0;
    auto end = split(r + 1);
    for (auto it = split(l); it != end; it++) {
        ans = (ans + qpow(it->v, x, y) * (it->r - it->l + 1)) % y;
    }
```

```cpp
    return ans;
}
```

```cpp
template<typename T>
T exgcd(T a, T b, T &x, T &y) {
    if (!b) { x = 1, y = 0; return a; }
    T d = exgcd(b, a % b, x, y);
    T z = x;
    x = y, y = z - y * (a / b);
    return d;
}

template<typename T>
T inv(T a, T p) {
    T x, y;
    exgcd(a, p, x, y);
    return (x % p + p) % p;
}

/*
    arbitrarily i, j: gcd(m_i, m_j) = 1
*/

ll CRT(const vector<ll> &r, const vector<ll> &m) {
    assert(r.size() == m.size());
    int n = int(r.size());
    ll x = 0, p = 1;
    for (int i = 0; i < n; i++) p *= m[i];
    for (int i = 0; i < n; i++) {
        ll t = p / m[i];
        x = (x + r[i] * t * inv(t, m[i])) % p;
    }
    return x;
}
```

```cpp
struct Deque {
    int head, tail;
    bool ok;
    vector<int> q;

    /*
        init(size)
        space : O(2 * n)
    */

    Deque(int n): head(n), tail(n - 1), ok(false), q(2 * n + 1, 0) {}

    bool empty() {
        return tail < head;
    }

    int size() {
        return tail - head + 1;
    }

    int front() {
        return ok ? q[tail] : q[head];
    }

    int back() {
        return ok ? q[head] : q[tail];
    }

    void push_front(int x) {
        ok ? q[++tail] = x : q[--head] = x;
    }

    void push_back(int x) {
        ok ? q[--head] = x : q[++tail] = x;
    }

    void pop_front() {
        ok ? tail-- : head++;
    }

    void reverse() {
        ok ^= 1;
    }
};
```

```
/*
    O(N^2)
*/
int n, m, e[N][N];
ll d[N];
bool vis[N];

void dijkstra(int s) {
    memset(vis, 0, sizeof(vis));
    memset(d, 0x3f, sizeof(d));
    d[s] = 0;
    for (int i = 1; i <= n; i++) {
        int u = -1;
        for (int v = 1; v <= n; v++) if (!vis[v] && (u == -1 || d[v] < d[u])) u = v;
        for (int v = 1; v <= n; v++) if (d[v] > d[u] + e[u][v]) d[v] = d[u] + e[u][v];
    }
}
```

```
struct dijkstra {
    int n;
    vector<vector<pair<int, int>>> G;
    vector<bool> vis;
    vector<ll> d;

    dijkstra(int n, vector<vector<pair<int, int>>> G) : n(n), d(n + 1), vis(n + 1), G(G) {}

    void dij(int s) {
        d.assign(n + 1, INF);
        vis.assign(n + 1, false);
        priority_queue<pair<ll, int>, vector<pair<ll, int>>, greater<pair<ll, int>>> q;
        q.push({d[s] = 0, s});
        while (!q.empty()) {
            auto[_, u] = q.top();
            q.pop();
            if (vis[u]) continue;
            vis[u] = true;
            for (auto[v, w] : G[u]) {
                if (d[v] > d[u] + w) {
                    q.push({d[v] = d[u] + w, v});
                }
            }
        }
    }
};
```

```
struct Dijkstra {
    struct edge {
        int node = -1;
        int64_t weight = 0;

        edge() {}

        edge(int _node, int64_t _weight) : node(_node), weight(_weight) {}
    };

    struct state {
        int64_t dist;
        int node;

```

```cpp
        state() {}

        state(int64_t _dist, int _node) : dist(_dist), node(_node) {}

        bool operator<(const state &other) const {
            return dist > other.dist;
        }
    };

    int n;
    vector<vector<edge>> adj;
    vector<int64_t> dist;
    vector<int> parent;

    Dijkstra(int _n = 0) {
        init(_n);
    }

    void init(int _n) {
        n = _n;
        adj.assign(n, {});
    }

    void add_directional_edge(int a, int b, int64_t weight) {
        adj[a].emplace_back(b, weight);
    }

    void add_bidirectional_edge(int a, int b, int64_t weight) {
        add_directional_edge(a, b, weight);
        add_directional_edge(b, a, weight);
    }

    void dijkstra_check(priority_queue<state> &pq, int node, int from, int64_t new_dist) {
        if (new_dist < dist[node]) {
            dist[node] = new_dist;
            parent[node] = from;
            pq.emplace(new_dist, node);
        }
    }

    void dijkstra(const vector<int> &source) {
        if (n == 0) return;
        dist.assign(n, INF64);
        parent.assign(n, -1);
        priority_queue<state> pq;

        for (int src : source)
            dijkstra_check(pq, src, -1, 0);

        while (!pq.empty()) {
            state top = pq.top();
            pq.pop();

            if (top.dist > dist[top.node])
                continue;

            for (edge &e : adj[top.node])
                dijkstra_check(pq, e.node, top.node, top.dist + e.weight);
        }
    }
};
```

```
#include<bits/stdc++.h>

#define se second
#define fi first
using namespace std;
typedef long long ll;
typedef pair<int, int> P;
const double eps = 1e-8;
const int NINF = 0xc0c0c0c0;
const ll INF = 0x3f3f3f3f3f3f3f3f;
const ll mod = 1e9 + 7;
const ll maxn = 1e6 + 5;
const int N = 1e5 + 5;

int n, m, s;

struct edge {
    int to, cost;
};
vector<edge> G[N];
ll d[N];
int path[N];

void dijkstra(const int &s, const int &V) {
    priority_queue<P, vector<P>, greater<P> > q;
    memset(d, 0x3f, sizeof(d));
    memset(path, -1, sizeof(path));
    d[s] = 0;
    q.push({0, s});
    while (!q.empty()) {
        P t = q.top();
        q.pop();
        int v = t.se;
        if (d[v] < t.fi) continue;

        for (int i = 0; i < G[v].size(); i++) {
            edge e = G[v][i];
            if (d[e.to] > d[v] + e.cost) {
                d[e.to] = d[v] + e.cost;
                path[e.to] = v;
                q.push({d[e.to], e.to});
            }
        }
    }
}

void print(int e) {
    if (d[e] >= INF) {
        cout << -1 << '\n';
        return;
    }
    vector<int> prev;
    for (; e != -1; e = path[e]) prev.push_back(e);
    reverse(prev.begin(), prev.end());
    for (auto c:prev) cout << c << " ";
    cout << '\n';
}

int main() {
    ios::sync_with_stdio(false);
    cin.tie(0);
    cin >> n >> m;
    for (int i = 1; i <= m; i++) {
        int u, v, w;
        cin >> u >> v >> w;
        G[u].push_back({v, w});
        G[v].push_back({u, w});
    }
    dijkstra(1, n);
    print(n);
    return 0;
}
```

```
/*
    N = 点数 + 1
*/

template<typename T>
struct Dinic {
    struct Edge {
        int from, to;
        T cap, flow;
    };

    int n, m, s, t;
    vector<int> d, cur;
    vector <vector<int>> G;
    vector<bool> vis;
    vector <Edge> edges;

    Dinic() : n(0) {}

    Dinic(int n) : n(n), d(n, 0), cur(n, 0), G(n), vis(n, false) {}

    void init(int n) {
        for (int i = 0; i < n; i++) G[i].clear();
        edges.clear();
    }

    void add(int from, int to, T cap, T incap = 0) {
        edges.push_back({from, to, cap, 0});
        edges.push_back({to, from, incap, 0});
        m = edges.size();
        G[from].push_back(m - 2);
        G[to].push_back(m - 1);
    }

    bool bfs() {
        vis.assign(n, false);
        queue<int> Q;
        d[s] = 0;
        vis[s] = true;
        Q.push(s);
        while (!Q.empty()) {
            int u = Q.front();
            Q.pop();
            for (auto v : G[u]) {
                Edge &e = edges[v];
                if (!vis[e.to] && e.cap > e.flow) {
                    vis[e.to] = true;
                    d[e.to] = d[u] + 1;
                    Q.push(e.to);
                }
            }
        }
        return vis[t];
    }

    T dfs(int u, T w) {
        if (u == t || w == 0) return w;
        T flow = 0, f;
        for (int &i = cur[u]; i < G[u].size(); i++) {
            int v = G[u][i];
            Edge &e = edges[v];
            if (d[e.to] == d[u] + 1 && (f = dfs(e.to, min(w, e.cap - e.flow))) > 0) {
                e.flow += f;
                edges[v ^ 1].flow -= f;
                flow += f;
                w -= f;
                if (w == 0) break;
            }
        }
        return flow;
    }

    T Maxflow(int s, int t) {
        return Maxflow(s, t, numeric_limits<T>::max());
    }

    T Maxflow(int s, int t, T limit) {
        this->s = s;
        this->t = t;
        T maxflow = 0;
        while (bfs()) {
            cur.assign(n, 0);
            maxflow += dfs(s, limit);
        }
        return maxflow;
    }
};
```

```
#define maxn 250
#define INF 0x3f3f3f3f

struct Edge {
  int from, to, cap, flow;
  Edge(int u, int v, int c, int f) : from(u), to(v), cap(c), flow(f) {}
};

struct Dinic {
  int n, m, s, t;
  vector<Edge> edges;
  vector<int> G[maxn];
  int d[maxn], cur[maxn];
  bool vis[maxn];

  void init(int n) {
    for (int i = 0; i < n; i++) G[i].clear();
    edges.clear();
  }

  void AddEdge(int from, int to, int cap) {
    edges.push_back(Edge(from, to, cap, 0));
    edges.push_back(Edge(to, from, 0, 0));
    m = edges.size();
    G[from].push_back(m - 2);
    G[to].push_back(m - 1);
  }

  bool BFS() {
    memset(vis, 0, sizeof(vis));
    queue<int> Q;
    Q.push(s);
    d[s] = 0;
    vis[s] = 1;
    while (!Q.empty()) {
      int x = Q.front();
      Q.pop();
      for (int i = 0; i < G[x].size(); i++) {
        Edge& e = edges[G[x][i]];
        if (!vis[e.to] && e.cap > e.flow) {
          vis[e.to] = 1;
          d[e.to] = d[x] + 1;
          Q.push(e.to);
        }
      }
    }
    return vis[t];
  }

  int DFS(int x, int a) {
    if (x == t || a == 0) return a;
    int flow = 0, f;
    for (int& i = cur[x]; i < G[x].size(); i++) {
      Edge& e = edges[G[x][i]];
      if (d[x] + 1 == d[e.to] && (f = DFS(e.to, min(a, e.cap - e.flow))) > 0) {
        e.flow += f;
        edges[G[x][i] ^ 1].flow -= f;
        flow += f;
        a -= f;
        if (a == 0) break;
      }
    }
    return flow;
  }

  int Maxflow(int s, int t) {
    this->s = s;
    this->t = t;
    int flow = 0;
    while (BFS()) {
      memset(cur, 0, sizeof(cur));
      flow += DFS(s, INF);
    }
    return flow;
  }
};
```

```
/*
    Hash S(s);
    Sometimes we need to use a lot of primes to hash instead of overflow.

    if MLE, try:
        using ull = unisgned int;
*/

struct Hash_string{
    using ull = unsigned long long;
    vector<ull> H, P;
    ull base = 131;
    int n;

    Hash_string(string s):n((int)s.size()), H((int)s.size() + 1, 0), P((int)s.size() + 1, 0){
        P[0] = 1;
        s = " " + s;
        for (int i = 1; i <= n; i++) {
            H[i] = H[i - 1] * base + s[i] - 'a' + 1;
            P[i] = P[i - 1] * base;
        }
    }

    ull get(int L, int R) {
        return H[R] - H[L - 1] * P[R - L + 1];
    }
};



/*
    the only difference with last is get return pair to check.

    Hash S(s);
    Sometimes we need to use a lot of primes to hash instead of overflow.

    if MLE, try:
        using ull = unisgned int;
*/

struct Hash_string{
    using ull = unsigned long long;
    using ui = unsigned int;
    using PLI = pair<ull, ui>;
    vector<ull> H, P;
    ull base = 131;
    vector<ui> Hi, Pi;
    ull basei = 131;
    int n;

    Hash_string(string s):n((int)s.size()), H((int)s.size() + 1, 0), P((int)s.size() + 1, 0), Hi((int)s.size() + 1, 0), Pi((int)s.size() + 1, 0) {
        P[0] = 1;
        Pi[0] = 1;
        s = " " + s;
        for (int i = 1; i <= n; i++) {
            H[i] = H[i - 1] * base + s[i] - 'a' + 1;
            P[i] = P[i - 1] * base;
            Hi[i] = Hi[i - 1] * basei + s[i] - 'a' + 1;
            Pi[i] = Pi[i - 1] * basei;
        }
    }

    PLI get(int L, int R) {
        return make_pair(H[R] - H[L - 1] * P[R - L + 1], Hi[R] - Hi[L - 1] * Pi[R - L + 1]);
    }
};
```

```
namespace HASH {
    int to[N], cnt[N], next[N], head[mod], tot;

    inline void clear() {
        memset(head, tot = 0, sizeof(head));
        memset(cnt, 0, sizeof(cnt));
    }

    inline void insert(int x) {
        int k = x % mod;
        for (int i = head[k]; i; i = next[i])
            if (to[i] == x) {
                cnt[i]++;
                return;
            }
        to[++tot] = x;
        cnt[tot] = 1;
        next[tot] = head[k];
        head[k] = tot;
    }

    inline int query(int x) {
        int k = x % mod;
        for (int i = head[k]; i; i = next[i]) if (to[i] == x) return cnt[i];
        return 0;
    }
}

template<typename T> struct Hash {
    int n, p, N;
    vector<vector<T>> G;
    vector<T> cnt, value;

    Hash(int N, int p) : n(0), p(p), G(p), cnt(N), value(N) {}

    void clear() {
        n = 0;
        vector<vector<T>> X;
        swap(G, X);
        vector<T> x1, x2;
        swap(cnt, x1);
        swap(value, x2);
    }

    void insert(int x) {
        int k = x % p;
        for (auto i : G[k]) {
            if (value[i] == x) {
                cnt[i]++;
                return;
            }
        }
        value[n] = x;
        cnt[n] = 1;
        G[k].push_back(n++);
    }

    int query(int x) {
        int k = x % p;
        for (auto i : G[k]) {
            if (value[i] == x) {
                return cnt[i];
            }
        }
        return 0;
    }
};
```

```
#include <bits/stdc++.h>

#define fi first
#define se second
#define mp make_pair

using namespace std;
typedef long long ll;
typedef pair<int, int> pii;
const double eps = 1e-8;
const int NINF = 0xc0c0c0c0;
const int INF = 0x3f3f3f3f;
const ll mod = 1e9 + 7;
const ll N = 1e6 + 5;

string s, t;
int n, m, kmp[N], nx[N];

void get_nextval() {
    int i = 0, j = -1, len2 = t.size();
    kmp[0] = -1;
    while (i < len2) {
        if (j == -1 || t[i] == t[j]) {
            ++i, ++j;
            if (t[i] != t[j]) kmp[i] = j;
            else kmp[i] = kmp[j];
        } else j = kmp[j];
    }
}

void get_next() {
    int i = 0, j = -1, len2 = t.size();
    nx[0] = -1;
    while (i < len2) {
        if (j == -1 || t[i] == t[j]) nx[++i] = ++j;
        else j = nx[j];
    }
}

void get_kmp() {
    int i = 0, j = 0, len1 = s.size(), len2 = t.size();
    while (i < len1) {
```

```cpp
        if (j == -1 || s[i] == t[j]) i++, j++;
        else j = kmp[j];
        if (j == len2) cout << i - j + 1 << '\n', j = kmp[j];
    }
}

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);
    cin >> s >> t;
    n = s.size(), m = t.size();
    get_nextval();
    get_kmp();
    get_next();
    for (int i = 1; i <= m; i++) cout << nx[i] << " \n"[i == m];
    return 0;
}
```

```cpp
#include <bits/stdc++.h>

#define fi first
#define se second
#define mp make_pair

using namespace std;
typedef long long ll;
typedef pair<int, int> pii;
const double eps = 1e-8;
const int NINF = 0xc0c0c0c0;
const int INF = 0x3f3f3f3f;
const ll mod = 1e9 + 7;
const ll N = 5e5 + 5;

int n, m, s, deep[N], f[N][21], lg[N];
vector<int> G[N];

void dfs(int u, int fa) {
    f[u][0] = fa;
    deep[u] = deep[fa] + 1;
    for (auto v:G[u]) {
        if (v == fa) continue;
        dfs(v, u);
    }
}

void ST(int n) {
    for (int j = 1; (1 << j) <= n; j++)
        for (int i = 1; i <= n; i++)
```

```cpp
            f[i][j] = f[f[i][j - 1]][j - 1];
}

int LCA(int u, int v) {
    if (deep[u] < deep[v]) swap(u, v);
    int h = deep[u] - deep[v];
    for (int i = 0; i < 20; i++)
        if (h & (1 << i)) u = f[u][i];
    if (u == v) return u;
    for (int i = 19; i >= 0; i--) {
        if (f[u][i] != f[v][i]) {
            u = f[u][i];
            v = f[v][i];
        }
    }
    return f[u][0];
}

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

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);
    cin >> n >> m >> s;
    for (int i = 1; i < n; i++) {
        int u, v;
        cin >> u >> v;
        G[u].emplace_back(v);
        G[v].emplace_back(u);
    }
    dfs(s, 0);
    ST(n);
    while (m--) {
        int u, v;
        cin >> u >> v;
        cout << LCA(u, v) << '\n';
    }
    return 0;
}
```

```cpp
#include<bits/stdc++.h>

using namespace std;
typedef long long ll;
const int INF = 0x3f3f3f3f;
const ll mod = 1e9 + 7;
const ll MAXN = 1e6 + 5;

ll p = mod;

inline ll qpow(ll a, ll b) {
    ll base = a % p;
    ll ans = 1;
    while (b > 0) {
        if (b & 1) ans = (ans * base) % p;
        base = base * base % p;
        b >>= 1;
    }
    return ans;
}

inline ll C(ll n, ll m) {
    if (n < m) return 0;//组合数n<m特判
    if (m > n - m) m = n - m;//组合数性质
    ll a = 1, b = 1;
    for (int i = 0; i < m; i++) {
        a = (a * (n - i) % p);//组合数分子 a
        b = (b * (i + 1)) % p;//组合数分母 b
    }
    return a * qpow(b, p - 2) % p;//费马小定理 a/b=a*b^(p-2)
}

inline ll Lucas(ll n, ll m, ll p = mod) {
    return m == 0 ? 1 : Lucas(n / p, m / p) * C(n % p, m % p) % p;
}

int main() {
    ios::sync_with_stdio(false);
    cin.tie(0);
    ll n, a, b;
    cin >> n >> a >> b;
    cout << (mod + qpow(2, n) - 1 - (C(n, a) + C(n, b)) % mod) % mod;
    return 0;
}
```

```cpp
#include<bits/stdc++.h>

using namespace std;
typedef long long ll;
typedef pair<int, int> P;
const double eps = 1e-8;
const int NINF = 0xc0c0c0c0;
const int INF = 0x3f3f3f3f;
const ll mod = 1e9 + 7;
const ll maxn = 3e7 + 5;

char s[maxn], str[maxn];
int n, p[maxn], ans;

void manacher_init() {
    str[0] = str[1] = '#';
    for (int i = 0; i < n; i++) {
        str[(i << 1) + 2] = s[i];
        str[(i << 1) + 3] = '#';
    }
    n = (n << 1) + 2;
    str[n] = 0;
}

void manacher() {
    int mx = 0, mid;
    for (int i = 1; i < n; i++) {
        if (mx > i) p[i] = min(p[(mid << 1) - i], p[mid] + mid - i);
        else p[i] = 1;
        for (; str[i + p[i]] == str[i - p[i]]; p[i]++);
        if (p[i] + i > mx) mx = p[i] + i, mid = i;
    }
}

int main() {
    ios::sync_with_stdio(false);
    cin.tie(0);
    cin >> s;
    n = strlen(s);
    manacher_init();
    manacher();
    cout << *max_element(p, p + n) - 1 << '\n';
    return 0;
}
```

```cpp
#include <bits/stdc++.h>

#define fi first
#define se second
#define mp make_pair
#define pb push_back
#define ls (p << 1)
#define rs (ls | 1)
#define tm ((tl + tr) >> 1)
#define lowbit(x) ((x) & -(x))

```

```cpp
using namespace std;
using ll = long long;
using ull = unsigned long long;
using pii = pair<int, int>;

constexpr double eps = 1e-8;
constexpr int NINF = 0xc0c0c0c0;
constexpr int INF = 0x3f3f3f3f;
constexpr ll LNINF = 0xc0c0c0c0c0c0c0c0;
constexpr ll LINF = 0x3f3f3f3f3f3f3f3f;
constexpr ll mod = 1e9 + 7;
constexpr ll N = 1e6 + 5;

int n, a[] = {0, 9, 82, 5, 3, 9, 2, 1, 0}, b[N];

void merge_sort(int L,int R){
  if(L == R) return;
  int mid = (L + R) >> 1;
  merge_sort(L, mid);
  merge_sort(mid + 1, R);
  int i = L, j= mid + 1, k = L;
  while(i <= mid && j <= R){
    if(a[i] <= a[j]) b[k++] = a[i++];
    else b[k++] = a[j++];
  }
  while(i <= mid) b[k++] = a[i++];
  while(j <= R) b[k++] = a[j++];
  for(int i = L; i <= R; i++) a[i] = b[i];
}

int main(){
  ios::sync_with_stdio(0);
  cin.tie(0);
  int n = sizeof(a) / sizeof(a[0]);
  cout << "n = " << n << '\n';
  merge_sort(0, n - 1);
  for(int i = 0; i < n; i++) cout << a[i] << " \n"[i == n-1];
  return 0;
}
```

```cpp
ll qpow(ll a, ll b, ll p) {
    ll ans = 1;
    while (b > 0) {
        if (b & 1) ans = (__int128) ans * a % p;
        a = (__int128) a * a % p;
        b >>= 1;
    }
    return ans;
}

bool isPrime(ll x) {
    if (x < 3) return x == 2;
    if (x % 2 == 0) return false;
    ll a[] = {2, 325, 9375, 28178, 450775, 9780504, 1795265022}, d = x - 1, r = 0;
    while (d % 2 == 0) d /= 2, ++r;
    for (int i = 0; i < 7; i++) {
        ll v = qpow(a, d, x);
        if (v <= 1 || v == x - 1) continue;
        for (int i = 0; i < r; i++) {
            v = (__int128) v * v % x;
            if (v == x - 1) {
                v = 1;
                break;
            }
            if (v == 1) return false;
        }
        if (v != 1) return false;
    }
    return true;
}
```

```cpp
#include <bits/stdc++.h>

#define fi first
#define se second
#define mp make_pair
#define lowbit(x) ((x) & -(x))

using namespace std;
using ll = long long;
using pii = pair<int, int>;

constexpr double eps = 1e-8;
constexpr int NINF = 0xc0c0c0c0;
constexpr int INF = 0x3f3f3f3f;
constexpr ll mod = 1e9 + 7;
constexpr ll N = 2e5 + 5;

#define ls (p << 1)
#define rs (ls | 1)
#define tm ((tl + tr) >> 1)

int tag[N << 2], sum[N << 2], a[N];
string s, f;

```

```
void build(int p, int tl, int tr) {
    tag[p] = -1;
    if (tl == tr) sum[p] = a[tl];
    else {
        build(ls, tl, tm);
        build(rs, tm + 1, tr);
        sum[p] = sum[ls] + sum[rs];
    }
}

void modify(int p, int tl, int tr, int v) {
    sum[p] = (tr - tl + 1) * v;
    tag[p] = v;
}

void push_down(int p, int tl, int tr) {
    if (tag[p] != -1) {
        modify(ls, tl, tm, tag[p]);
        modify(rs, tm + 1, tr, tag[p]);
        tag[p] = -1;
    }
}

int rangesum(int p, int tl, int tr, int L, int R) {
    if (tl >= L && tr <= R) return sum[p];
    push_down(p, tl, tr);
    int res = 0;
    if (L <= tm) res += rangesum(ls, tl, tm, L, R);
    if (R > tm) res += rangesum(rs, tm + 1, tr, L, R);
    return res;
}

void modify(int p, int tl, int tr, int L, int R, int v) {
    if (tl >= L && tr <= R) return modify(p, tl, tr, v);
    push_down(p, tl, tr);
    if (L <= tm) modify(ls, tl, tm, L, R, v);
    if (R > tm) modify(rs, tm + 1, tr, L, R, v);
    sum[p] = sum[ls] + sum[rs];
}

bool check() {
    int n, q;
    cin >> n >> q;
    cin >> s >> f;
    s = " " + s, f = " " + f;
    for (int i = 1; i <= n; i++) a[i] = f[i] - '0';
    build(1, 1, n);
    vector<int> l(q), r(q);
    for (int i = 0; i < q; i++) {
```

```cpp
        cin >> l[i] >> r[i];
    }
    for (int i = q - 1; ~i; i--) {
        int s = rangesum(1, 1, n, l[i], r[i]);
        if (s == r[i] - l[i] + 1 - s) {
            return false;
        }
        modify(1, 1, n, l[i], r[i], s > r[i] - l[i] + 1 - s);
        s = rangesum(1, 1, n, l[i], r[i]);
    }
    for (int i = 1; i <= n; i++) {
        if (rangesum(1, 1, n, i, i) != s[i] - '0') {
            return false;
        }
    }
    return true;
}

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);

    int T;
    cin >> T;
    while (T--) {
        cout << (check() ? "YES" : "NO") << '\n';
    }

    return 0;
}
```

```cpp
#define ls (p << 1)
#define rs (ls | 1)
#define tm ((tl + tr) >> 1)

int mx[N << 2], a[N << 2];

void up(int p) {
    mx[p] = max(mx[ls], mx[rs]);
}

void build(int p, int tl, int tr) {
    if (tl == tr) {
        mx[p] = a[tl];
    } else {
        build(ls, tl, tm);
        build(rs, tm + 1, tr);
        up(p);
    }
}

void modify(int p, int tl, int tr, int pos, int v) {
    if (tl == tr) {
        mx[p] = v;
    } else {
        if (tm >= pos) modify(ls, tl, tm, pos, v);
        else modify(rs, tm + 1, tr, pos, v);
        up(p);
    }
}

int query(int p, int tl, int tr, int l, int r) {
    if (l <= tl && tr <= r) {
        return mx[p];
    } else {
        int ans = INT_MIN;
        if (tm >= l) ans = max(ans, query(ls, tl, tm, l, r));
        if (tm < r) ans = max(ans, query(rs, tm + 1, tr, l, r));
        return ans;
    }
}
```

```cpp
#define ls (p << 1)
#define rs (ls | 1)
#define tm ((tl + tr) >> 1)

int mi[N << 2], a[N << 2];

void up(int p) {
    mi[p] = min(mx[ls], mx[rs]);
}

void build(int p, int tl, int tr) {
    if (tl == tr) {
        mi[p] = a[tl];
    } else {
        build(ls, tl, tm);
        build(rs, tm + 1, tr);
        up(p);
    }
}

void modify(int p, int tl, int tr, int pos, int v) {
    if (tl == tr) {
        mi[p] = v;
    } else {
        if (tm >= pos) modify(ls, tl, tm, pos, v);
        else modify(rs, tm + 1, tr, pos, v);
        up(p);
    }
}

int query(int p, int tl, int tr, int l, int r) {
    if (l <= tl && tr <= r) {
        return mi[p];
    } else {
        int ans = INT_MAX;
        if (tm >= l) ans = min(ans, query(ls, tl, tm, l, r));
        if (tm < r) ans = min(ans, query(rs, tm + 1, tr, l, r));
        return ans;
    }
}
```

```cpp
#include <bits/stdc++.h>

#define fi first
#define se second
#define mp make_pair

using namespace std;
typedef long long ll;
typedef pair<int, int> pii;
const double eps = 1e-8;
const int NINF = 0xc0c0c0c0;
const int INF = 0x3f3f3f3f;
const ll mod = 1e9 + 7;
const ll N = 1e5 + 5;

ll n, m, a[N], f[N][25], Log2[N];

struct ST {
    vector<int> Log2;
    vector<vector<int>> f;
    int n;

    ST(int n): n(n), f(n + 1, vector<int> (25, 0)), Log2(n + 1) {}

    void init(vector<int> a) {
        for (int i = 2; i <= n; i++) Log2[i] = Log[i / 2] + 1;
        for (int i = 1; i <= n; i++) f[i][0] = a[i];
        for (int j = 1; (1 << j) <= n; j++)
            for (int i = 1; i + (1ll << j) - 1 <= n; i++)
                f[i][j] = max(f[i][j - 1], f[i + (1ll << (j - 1))][j - 1]);
    }

    inline ll query(int l, int r) {
        if (l > r) return 0;
        int k = Log2[r - l + 1];
        return max(f[l][k], f[r - (1ll << k) + 1][k]);
    }
};

void ST() {
    for (int i = 2; i <= n; i++) Log2[i] = Log[i / 2] + 1;
    for (int i = 1; i <= n; i++) f[i][0] = a[i];
    for (int j = 1; (1 << j) <= n; j++)
        for (int i = 1; i + (1ll << j) - 1 <= n; i++)
            f[i][j] = max(f[i][j - 1], f[i + (1ll << (j - 1))][j - 1]);
}

inline ll query(int l, int r) {
    if (l > r) return 0;
    int k = Log2[r - l + 1];
    return max(f[l][k], f[r - (1ll << k) + 1][k]);
}

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);
    cin >> n >> m;
    for (int i = 1; i <= n; i++) cin >> a[i];
    ST();
    while (m--) {
        int l, r;
        cin >> l >> r;
        cout << query(l, r) << '\n';
    }
    return 0;
}
```

```cpp
#include <bits/stdc++.h>

#define fi first
#define se second
#define mp make_pair

using namespace std;
typedef long long ll;
typedef pair<int, int> pii;
const double eps = 1e-8;
const int NINF = 0xc0c0c0c0;
const int INF = 0x3f3f3f3f;
const ll mod = 1e9 + 7;
const ll N = 1e6 + 5;

int low[N], dfn[N], bridge[N], cntBridge, dfsClock;
vector<int> G[N];

void tarjan(int u, int fa) {
    f[u] = fa;
    low[u] = dfn[u] = ++dfsClock;
    for (auto v:G[u]) {
        if (!dfn[v]) {
            tarjan(v, u);
            low[u] = min(low[u], low[v]);
            if (low[v] > dfn[u]) bridge[v] = true, cntBridge++;
        } else if (dfn[v] < dfn[u] && v != fa) low[u] = min(low[u], dfn[v]);
    }
}

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);

    return 0;
}
```

```cpp
#include<bits/stdc++.h>

#define fi first
#define se second
#define mp make_pair
using namespace std;
typedef long long ll;
typedef pair<int, int> P;
const double eps = 1e-8;
const int NINF = 0xc0c0c0c0;
const int INF = 0x3f3f3f3f;
const ll mod = 1e9 + 7;
const ll N = 2e4 + 5;

int n, m, cnt, low[N], dfn[N], cut[N];
vector<int> G[N];

void tarjan(int x, int fa) {
    low[x] = dfn[x] = ++cnt;
    int son = 0;
    for (auto c:G[x]) {
        if (!dfn[c]) {
            tarjan(c, fa);
            low[x] = min(low[x], low[c]);
            if (low[c] >= dfn[x] && x != fa) cut[x] = true;
            if (x == fa) son++;
        } else low[x] = min(low[x], dfn[c]);
    }
    if (son >= 2 && x == fa) cut[x] = true;
}

int main() {
    ios::sync_with_stdio(false);
    cin.tie(0);
    cin >> n >> m;
    for (int i = 1; i <= m; i++) {
        int u, v;
        cin >> u >> v;
        G[u].push_back(v);
        G[v].push_back(u);
    }
    for (int i = 1; i <= n; i++)
        if (!dfn[i])
            tarjan(i, i);
    int tot = 0;
    for (int i = 1; i <= n; i++)
        if (cut[i])
            tot++;
    cout << tot << '\n';
    for (int i = 1; i <= n; i++)
        if (cut[i])
            cout << i << " ";
    return 0;
}
```

```
#include <bits/stdc++.h>

#define fi first
#define se second
#define mp make_pair

using namespace std;
using ll = long long;
using pii = pair<int, int>;

constexpr double eps = 1e-8;
constexpr int NINF = 0xc0c0c0c0;
constexpr int INF = 0x3f3f3f3f;
constexpr ll mod = 1e9 + 7;
constexpr ll N = 2e5 + 5;
```

```cpp

int n, m, M, zkw_mx[N << 2], zkw_cnt[N << 2];

inline void update(int x) {
    int ls = x << 1;
    int rs = ls | 1;
    if (zkw_mx[ls] == zkw_mx[rs]) {
        zkw_mx[x] = zkw_mx[ls];
        zkw_cnt[x] = zkw_cnt[ls] + zkw_cnt[rs];
    } else if (zkw_mx[ls] < zkw_mx[rs]) {
        zkw_mx[x] = zkw_mx[rs];
        zkw_cnt[x] = zkw_cnt[rs];
    } else {
        zkw_mx[x] = zkw_mx[ls];
        zkw_cnt[x] = zkw_cnt[ls];
    }
}

inline void update(int x, int val) {
    zkw_mx[x += M] = val;
    for (x >>= 1; x; x >>= 1) update(x);
}

inline void query(int l, int r) {
    int x, mx = -1, cnt = 0;
    for (l += M - 1, r += M + 1; l ^ r ^ 1; l >>= 1, r >>= 1) {
        if (~l & 1) {
            x = l ^ 1;
            if (mx < zkw_mx[x]) {
                mx = zkw_mx[x];
                cnt = zkw_cnt[x];
            } else if (mx == zkw_mx[x]) cnt += zkw_cnt[x];
        }
        if (r & 1) {
            x = r ^ 1;
            if (mx < zkw_mx[x]) {
                mx = zkw_mx[x];
                cnt = zkw_cnt[x];
            } else if (mx == zkw_mx[x]) cnt += zkw_cnt[x];
        }
    }
    cout << mx << ' ' << cnt << '\n';
}

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);
    cin >> n >> m;
    for (M = 1; M <= n; M <<= 1);
    for (int i = 1; i <= n; i++) {
        cin >> zkw_mx[M + i];
        zkw_cnt[M + i] = 1;
    }
    for (int i = M - 1; i; i--) update(i);
    while (m--) {
        string op;
        int x, y;
        cin >> op >> x >> y;
        if (op == "Ask") query(x, y);
        else update(x, y);
    }
    return 0;
}
```

```cpp
#include <bits/stdc++.h>

#define fi first
#define se second
#define mp make_pair
#define lowbit(x) ((x) & -(x))
#define ls (p << 1)
#define rs (ls | 1)
#define tm ((tl + tr) >> 1)

using namespace std;
using ll = long long;
using ull = unsigned long long;
using pii = pair<int, int>;

constexpr double eps = 1e-8;
constexpr int NINF = 0xc0c0c0c0;
constexpr int INF = 0x3f3f3f3f;
constexpr ll LNINF = 0xc0c0c0c0c0c0c0c0;
constexpr ll LINF = 0x3f3f3f3f3f3f3f3f;
constexpr ll mod = 1e9 + 7;
constexpr ll N = 2e6 + 5;

struct line {
    int l, r, h, op;

    inline bool operator<(const line &T) const {
        return h == T.h ? op > T.op : h < T.h;
    }
} a[N];

int n, m, x[N], cnt[N << 2], len[N << 2];

```

```cpp
void push_up(int p, int tl, int tr) {
    if (cnt[p]) {
        len[p] = x[tr] - x[tl];
    } else {
        len[p] = len[ls] + len[rs];
    }
}

void modify(int p, int tl, int tr, int L, int R, int v) {
    if (x[tr] <= L || R <= x[tl]) return;
    if (L <= x[tl] && x[tr] <= R) {
        cnt[p] += v;
        push_up(p, tl, tr);
        return;
    }
    if (L <= x[tm]) modify(ls, tl, tm, L, R, v);
    if (R > x[tm]) modify(rs, tm, tr, L, R, v);
    push_up(p, tl, tr);
}

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);

    cin >> n;
    for (int i = 0; i < n; i++) {
        int x1, y1, x2, y2;
        cin >> x1 >> y1 >> x2 >> y2;
        a[++m] = {x1, x2, y1, 1};
        x[m] = x1;
        a[++m] = {x1, x2, y2, -1};
        x[m] = x2;
    }
    n = m;
    sort(a + 1, a + 1 + n);
    sort(x + 1, x + 1 + n);
    m = unique(x + 1, x + 1 + n) - (x + 1);
    ll ans = 0;
    for (int i = 1; i < n; i++) {
        modify(1, 1, m, a[i].l, a[i].r, a[i].op);
        ans += 1ll * len[1] * (a[i + 1].h - a[i].h);
    }
    cout << ans << '\n';

    return 0;
}
```

```cpp
#include <bits/stdc++.h>

#define fi first
#define se second
#define mp make_pair
#define pb push_back
#define ls (p << 1)
#define rs (ls | 1)
#define tm ((tl + tr) >> 1)
#define lowbit(x) ((x) & -(x))

using namespace std;
using db = double;
using ll = long long;
using ull = unsigned long long;
using pii = pair<int, int>;

const db PI = acos(-1.0);
constexpr db EPS = 1e-8;
constexpr int NINF = 0xc0c0c0c0;
constexpr int INF = 0x3f3f3f3f;
constexpr ll LNINF = 0xc0c0c0c0c0c0c0c0;
constexpr ll LINF = 0x3f3f3f3f3f3f3f3f;
constexpr ll MOD = 1e9 + 7;
constexpr ll N = 1e6 + 5;

int tot, rt[N];

struct node {
    int l, r;
    ll sum;
} t[N << 5];

void modify(int &x, int l, int r, int v) {
    t[++tot] = t[x];
    x = tot;
    t[x].sum += v;
    if (l == r) return;
    int mid = (l + r) >> 1;
    if (v <= mid) modify(t[x].l, l, mid, v);
    else modify(t[x].r, mid + 1, r, v);
}

ll query(int x, int y, int l, int r, ll v) {
    if (l == r) return t[y].sum - t[x].sum;
    int mid = (l + r) >> 1;
    if (v <= mid) return query(t[x].l, t[y].l, l, mid, v);
    else return query(t[x].r, t[y].r, mid + 1, r, v) + t[t[y].l].sum - t[t[x].l].sum;
}

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);
    cout << fixed << setprecision(20);

    int n, m;
    cin >> n >> m;
    for (int i = 1; i <= n; i++) {
        int x;
        cin >> x;
        modify(rt[i] = rt[i - 1], 1, 1000000000, x);
    }
    int l, r;
    ll ans = 0;
    while (m--) {
        cin >> l >> r;
        l = (l + ans) % n + 1;
        r = (r + ans) % n + 1;
        if (l > r) swap(l, r);
        ans = 0;
        while (true) {
            ll res = query(rt[l - 1], rt[r], 1, 1000000000, ans + 1);
            if (res == ans) break;
            ans = res;
        }
        cout << ++ans << '\n';
    }

    return 0;
}
```

```
#include <bits/stdc++.h>

#define fi first
#define se second
#define mp make_pair
#define pb push_back
#define ls (p << 1)
#define rs (ls | 1)
#define tm ((tl + tr) >> 1)
#define lowbit(x) ((x) & -(x))

using namespace std;
using db = double;
using ll = long long;
using ull = unsigned long long;
```

```cpp
using pii = pair<int, int>;

const db PI = acos(-1.0);
constexpr db EPS = 1e-8;
constexpr int NINF = 0xc0c0c0c0;
constexpr int INF = 1e9;
constexpr ll LNINF = 0xc0c0c0c0c0c0c0c0;
constexpr ll LINF = 0x3f3f3f3f3f3f3f3f;
constexpr ll MOD = 1e9 + 7;
constexpr ll N = 1e6 + 5;

int rt[N], a[N], tot;

struct node {
    int l, r, cnt;
} t[N << 5];

void modify(int &x, int l, int r, int v) {
    t[++tot] = t[x];
    x = tot;
    t[x].cnt++;
    if (l == r) return;
    int mid = (l + r) >> 1;
    if (v <= mid) modify(t[x].l, l, mid, v);
    else modify(t[x].r, mid + 1, r, v);
}

int query(int x, int y, int l, int r, int v) {
    if (l == r) return l;
    int mid = (l + r) >> 1, cnt = t[t[y].l].cnt - t[t[x].l].cnt;
    if (v <= cnt) return query(t[x].l, t[y].l, l, mid, v);
    else return query(t[x].r, t[y].r, mid + 1, r, v - cnt);
}

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);
    cout << fixed << setprecision(20);

    int n, m;
    cin >> n >> m;
    for (int i = 1; i <= n; i++) {
        cin >> a[i];
        modify(rt[i] = rt[i - 1], -INF, INF, a[i]);
    }
    while (m--) {
        int l, r, k;
        cin >> l >> r >> k;
        cout << query(rt[l - 1], rt[r], -INF, INF, k) << '\n';
    }

    return 0;
}
```

```cpp
#include <bits/stdc++.h>

#define fi first
#define se second
#define mp make_pair
#define pb push_back
#define ls (p << 1)
#define rs (ls | 1)
#define tm ((tl + tr) >> 1)
#define lowbit(x) ((x) & -(x))

using namespace std;
using db = double;
using ll = long long;
using ull = unsigned long long;
using pii = pair<int, int>;

const db PI = acos(-1.0);
constexpr db EPS = 1e-8;
constexpr int NINF = 0xc0c0c0c0;
constexpr int INF = 1000000;
constexpr ll LNINF = 0xc0c0c0c0c0c0c0c0;
constexpr ll LINF = 0x3f3f3f3f3f3f3f3f;
constexpr ll MOD = 1e9 + 7;
constexpr ll N = 1e5 + 5;

struct node {
    int l, r, cnt;
    ll sum;
} t[N << 5];

int tot, rt[N];

void modify(int &x, int l, int r, int v) {
    t[++tot] = t[x];
    x = tot;
    t[x].cnt++;
    t[x].sum += v;
    if (l == r) return;
    int mid = (l + r) >> 1;
    if (v <= mid) modify(t[x].l, l, mid, v);
    else modify(t[x].r, mid + 1, r, v);
}
```

```cpp

ll query(int x, int y, int l, int r, int v) {
    if (l == r) return 1ll * v * l;
    int mid = (l + r) >> 1, cnt = t[t[y].r].cnt - t[t[x].r].cnt;
    if (v <= cnt) return query(t[x].r, t[y].r, mid + 1, r, v);
    else return query(t[x].l, t[y].l, l, mid, v - cnt) + t[t[y].r].sum - t[t[x].r].sum;
}

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);
    cout << fixed << setprecision(20);

    int T;
    cin >> T;
    while (T--) {
        tot = 0;
        int n;
        cin >> n;
        for (int i = 1; i <= n; i++) {
            int x;
            cin >> x;
            modify(rt[i] = rt[i - 1], 1, 1000000, x);
        }
        int m;
        cin >> m;
        while (m--) {
            int l, r, k;
            cin >> l >> r >> k;
            int x = r - l + 1;
            cout << 1ll * x * (x + 1) * (2 * x + 1) / 6 + query(rt[l - 1], rt[r], 1, 1000000, k) << '\n';
        }
    }

    return 0;
}
```