

比赛板子

aet重载运算符

```
1 #include <bits/stdc++.h>
2
3 #define PII pair<int, int>
4 using namespace std;
5
6 const int N = 2e5 + 10;
7 int a[N], b[N], p[N], c[N], ans[N];
8 int n, m;
9 struct cmp
10 {
11     bool operator()(int a, int b) const
12     {
13         return p[a] < p[b];
14     }
15 };
16 set<int, cmp> s[4];
17 bool st[N];
18
19 int main()
20 {
21     ios::sync_with_stdio(false);
22     cin.tie(nullptr);
23     cin >> n;
24     for (int i = 0; i < n; i++)
25     {
26         cin >> p[i];
27     }
28     for (int i = 0; i < n; i++)
29     {
30         cin >> a[i];
31         s[a[i]].insert(i);
32     }
33     for (int i = 0; i < n; i++)
34     {
35         cin >> b[i];
36         s[b[i]].insert(i);
37     }
38 }
```

```

39     cin >> m;
40     for (int i = 0; i < m; i++)
41         cin >> c[i];
42     for (int i = 0; i < m; i++)
43     {
44         int x = c[i];
45         bool flag = false;
46         while (1)
47         {
48             if (s[x].empty())
49                 break;
50             int res = *s[x].begin();
51             s[x].erase(res);
52             if (st[res])
53                 continue;
54             else
55             {
56                 st[res] = true;
57                 cout << p[res] << ' ';
58                 flag = true;
59                 break;
60             }
61         }
62         if (!flag)
63             cout << "-1 ";
64     }
65     return 0;
66 }
67

```

anti-SG



Anti-SG

☞ 对于任意一个Anti-SG游戏[†]，如果我们规定当局面中所有的单一游戏的SG值为0时，游戏结束(操作的人失败)，则先手必胜当且仅当：

- ☞ (1) 游戏的SG函数不为0且游戏中某个单一游戏的SG函数大于1。
- ☞ (2) 游戏的SG函数为0且游戏中没有单一游戏的SG函数大于1。

sg翻硬币模型



翻硬币问题

- 1. n 枚硬币排成一排，有的正面朝上，有的反面朝上。我们从左开始对硬币按1~ n 编号。
- 2. 两个人轮流根据某些约束翻硬币（如：每次只能翻一或两枚，或者每次只能翻连续的几枚），但他所翻动的硬币中，最右边的必须是从正面翻到反面。
- 3. 谁不能翻谁输。

所有1的位置，其余的位置为0的sg值异或和

10101

答案=sg(1)^sg(001)^sg(00001)

dinic(最大流)

```
1 #include <bits/stdc++.h>
2 using namespace std;
3 int read()
4 {
5     int num = 0;
6     bool flag = 1;
7     char c = getchar();
8     for (; c < '0' || c > '9'; c = getchar())
9         if (c == '-')
10             flag = 0;
11     for (; c >= '0' && c <= '9'; c = getchar())
12         num = (num << 1) + (num << 3) + c - '0';
13     return flag ? num : -num;
14 }
15 const int N = 110;
16 const int M = 5010;
17 template <typename T>
18 struct FlowGraph
19 {
20     int s, t, vtot;
21     int head[N], etot;
22     int dis[N], cur[N];
23     struct edge
24     {
25         int v, nxt;
```

```

26     T f;
27 } e[M << 1];
28 void addedge(int u, int v, T f, T f2 = 0)
29 {
30     e[etot] = {v, head[u], f};
31     head[u] = etot++;
32     e[etot] = {u, head[v], f2};
33     head[v] = etot++;
34 }
35 bool bfs()
36 {
37     for (int i = 1; i <= vtot; i++)
38     {
39         dis[i] = 0;
40         cur[i] = head[i];
41     }
42     queue<int> q;
43     q.push(s);
44     dis[s] = 1;
45     while (!q.empty())
46     {
47         int u = q.front();
48         q.pop();
49         for (int i = head[u]; ~i; i = e[i].nxt)
50         {
51             if (e[i].f && !dis[e[i].v])
52             {
53                 int v = e[i].v;
54                 dis[v] = dis[u] + 1;
55                 if (v == t)
56                     return 1;
57                 q.push(v);
58             }
59         }
60     }
61     return 0;
62 }
63 T dfs(int u, T m)
64 {
65     if (u == t)
66         return m;
67     T flow = 0;
68     for (int i = cur[u]; ~i; cur[u] = i = e[i].nxt)
69         if (e[i].f && dis[e[i].v] == dis[u] + 1)
70         {
71             T f = dfs(e[i].v, min(m, e[i].f));
72             e[i].f -= f;

```

```

73         e[i ^ 1].f += f;
74         m -= f;
75         flow += f;
76         if (!m)
77             break;
78     }
79     if (!flow)
80         dis[u] = -1;
81     return flow;
82 }
83 T dinic()
84 {
85     T flow = 0;
86     while (bfs())
87         flow += dfs(s, numeric_limits<T>::max());
88     return flow;
89 }
90 void init(int s_, int t_, int vtot_)
91 {
92     s = s_;
93     t = t_;
94     vtot = vtot_;
95     etot = 0;
96     for (int i = 1; i <= vtot; i++)
97         head[i] = -1;
98 }
99 };
100 FlowGraph<long long> g;
101 int n, m, s, t;
102 int main()
103 {
104     n = read();
105     m = read();
106     s = read();
107     t = read();
108     g.init(s, t, n);
109     for (int i = 1; i <= m; i++)
110     {
111         int x = read(), y = read(), z = read();
112         g.addedge(x, y, z);
113     }
114     printf("%lld\n", g.dinic());
115     return 0;
116 }
117

```

d阶nim



d阶nim

单击此处添加文本

I

N 阶 Nim 游戏：有 k 堆石子，各包含 x_1, x_2, \dots, x_k 颗石子。双方玩家轮流操作，每次操作选择其中非空的若干堆，至少一堆但不超过 N 堆，在这若干堆中的每堆各取走其中的若干颗石子（1 颗，2 颗...甚至整堆），数目可以不同，取走最后一颗石子的玩家获胜。

结论：当且仅当在每一个不同的二进制位上， x_1, x_2, \dots, x_k 中在该位上 1 的个数是 $N+1$ 的倍数时，后手方有必胜策略，否则先手必胜。

阶梯nim



阶梯nim

有 n 堆石子。两个人轮流取，每次可以在第 i 堆石子里面选取若干石头放到 $i-1$ 堆里面。谁不能操作算输。

变种：移棋子问题。

所有偶数堆的sg值异或

ex_gcd

```
1 #include <bits/stdc++.h>
2 using namespace std;
3 int ex_gcd(int a, int &x, int b, int &y, int c)
4 {
5     if (b == 0)
6     {
7         x = c / a;
8         y = 0;
9         return a;
10    }
11    int d = ex_gcd(b, y, a % b, x, c);
12    y = (y - (a / b) * x) % Mod;
```

```

13     return d;
14 }
15 signed main()
16 {
17     return 0;
18 }
19

```

KM算法(完美最大权匹配 n^3)

```

1 #include <bits/stdc++.h>
2 using namespace std;
3 #define int long long
4 const int N = 1010;
5 const int inf = 1e18;
6 struct _KM
7 {
8     int n, Map[N][N], matched[N];
9     int slack[N], pre[N], ex[N], ey[N];
10    bool visx[N], visy[N];
11    void match(int u)
12    {
13        int x, y = 0, yy = 0, delta;
14        memset(pre, 0, sizeof(pre));
15        for (int i = 1; i <= n; i++)
16            slack[i] = inf;
17        matched[y] = u;
18        while (1)
19        {
20            x = matched[y];
21            delta = inf;
22            visy[y] = 1;
23            for (int i = 1; i <= n; i++)
24            {
25                if (visy[i])
26                    continue;
27                if (slack[i] > ex[x] + ey[i] - Map[x][i])
28                {
29                    slack[i] = ex[x] + ey[i] - Map[x][i];
30                    pre[i] = y;
31                }
32                if (slack[i] < delta)
33                {
34                    delta = slack[i];
35                    yy = i;

```

```

36         }
37     }
38     for (int i = 0; i <= n; i++)
39     {
40         if (visy[i])
41             ex[matched[i]] -= delta, ey[i] += delta;
42         else
43             slack[i] -= delta;
44     }
45     y = yy;
46     if (matched[y] == -1)
47         break;
48 }
49 while (y)
50 {
51     matched[y] = matched[pre[y]];
52     y = pre[y];
53 }
54 }
55 int KM()
56 {
57     memset(matched, -1, sizeof(matched));
58     memset(ex, 0, sizeof(ex));
59     memset(ey, 0, sizeof(ey));
60     for (int i = 1; i <= n; i++)
61     {
62         memset(visy, 0, sizeof(visy));
63         match(i);
64     }
65     int res = 0;
66     for (int i = 1; i <= n; i++)
67         if (matched[i] != -1)
68             res += Map[i][matched[i]];
69     return res / 2;
70 }
71 void init(int _n)
72 {
73     n = _n;
74     for (int i = 1; i <= n; i++)
75         for (int j = 1; j <= n; j++)
76             Map[i][j] = -inf;
77 }
78 } g;
79 int n, a[N], p[N], b[N], c[N];
80 signed main()
81 {
82     scanf("%lld", &n);

```



```

83     for (int i = 1; i <= n; i++)
84         scanf("%lld", &a[i]);
85     for (int i = 1; i <= n; i++)
86         scanf("%lld", &p[i]);
87     for (int i = 1; i <= n; i++)
88         scanf("%lld", &b[i]);
89     for (int i = 1; i <= n; i++)
90         scanf("%lld", &c[i]);
91     g.init(n * 2);
92     for (int i = 1; i <= n; i++)
93     {
94         for (int j = 1; j <= n; j++)
95         {
96             int now = b[i] + c[j], w = 0;
97             for (int k = 1; k <= n; k++)
98                 if (a[k] < now)
99                     w += p[k];
100             g.Map[i][j + n] = g.Map[j + n][i] = w;
101         }
102     }
103     printf("%lld", g.KM());
104     return 0;
105 }
106

```

Manacher

```

1  #include <bits/stdc++.h>
2  using namespace std;
3  const int N = 11e6 + 10;
4  char s[N], b[N << 1];
5  int p[N << 1];
6  int Manacher()
7  {
8      int n = strlen(s + 1);
9      for (int i = 1; i <= n; i++)
10         b[i * 2] = s[i], b[i * 2 - 1] = '$';
11     int len = n << 1;
12     b[0] = '%';
13     b[++len] = '$';
14     int pos, R = 0, Max = 0;
15     for (int i = 1; i <= len; i++)
16     {
17         if (i < R)
18             p[i] = min(R - i + 1, p[2 * pos - i]);
19
20         while (b[i + p[i]] == b[i - p[i]])
21             p[i]++;
22         if (i + p[i] > R)
23             R = i + p[i], pos = i;
24     }
25     return p[1];
26 }

```

```

19     else
20         p[i] = 1;
21     while (b[i + p[i]] == b[i - p[i]])
22         p[i]++;
23     if (i + p[i] - 1 > R)
24     {
25         R = i + p[i] - 1;
26         pos = i;
27     }
28     Max = max(Max, p[i] - 1);
29 }
30 return Max;
31 }
32 signed main()
33 {
34     scanf("%s", s + 1);
35     printf("%d", Manacher());
36     return 0;
37 }
38

```

差分约束

```

1 #include <bits/stdc++.h>
2 using namespace std;
3 const int N = 5e3 + 10;
4 int read()
5 {
6     int num = 0;
7     bool flag = 1;
8     char c = getchar();
9     for (; c < '0' || c > '9'; c = getchar())
10         if (c == '-')
11             flag = 0;
12     for (; c >= '0' && c <= '9'; c = getchar())
13         num = (num << 1) + (num << 3) + c - '0';
14     return flag ? num : -num;
15 }
16 int n, m, d[N];
17 vector<array<int, 3>> e;
18 signed main()
19 {
20     n = read();
21     m = read();
22     for (int i = 1; i <= n; i++)

```

```

23     {
24         int x = read(), y = read(), z = read();
25         //      e.push_back({y,x,z});
26         // d[x]-d[y]<=z -> 答案的最大值
27         e.push_back({x, y, z});
28         // 设di'=-di(答案是di),求di'的最大值,就知道了di的最小值
29         //(-dx')-(-dy')<=z -> dy'-dx'<=z 边反向,最后求答案
30     }
31     for (int i = 1; i <= n; i++)
32         e.push_back({0, i, 0});
33     // d[0]-d[i]<=0 -> d[0]<=d[i], 求的最小值,所以每个点其实是d0' di',
34     //(-d0')<=(-di') di'-d0'<=0
35     for (int i = 0; i <= n; i++)
36         for (auto k : e)
37             d[k[1]] = min(d[k[1]], d[k[0]] + k[2]);
38     for (auto k : e)
39         if (d[k[1]] > d[k[0]] + k[2])
40         {
41             printf("-1");
42             return 0;
43         }
44     for (int i = 1; i <= n; i++)
45         printf("%d ", -d[i] + d[0]);
46     // 求最小值,每个取负就是这个答案
47     return 0;
48 }
49

```

笛卡尔树

```

1 // 一个区间取出最小值为根
2 // 分成左右区间在左右儿子
3 // 两边分别重复这些操作
4 #include <bits/stdc++.h>
5 using namespace std;
6 // 性质1: 区间最小值为两个端点的lca
7 // 性质2: 中序遍历就是原数组
8 /*性质3: 一个点的祖先节点,如果这个祖先节点到这个点的路径是第一个向 左/右 的路径
9 ,那么这个祖先节点就是 左/右 边第一个小于等于它的数*/
10 const int N = 1e5 + 10;
11 int n, a[N], l[N], r[N];
12 void build()
13 {
14     stack<int> st;
15     int root = 0;

```

```

16  for (int i = 1; i <= n; i++)
17  {
18      int last = 0;
19      while (!st.empty() && a[st.top()] > a[i])
20      {
21          last = st.top();
22          st.pop();
23      }
24      if (!st.empty())
25          r[st.top()] = i;
26      else
27          root = i;
28      l[i] = last;
29      st.push(i);
30  }
31 }
32 signed main()
33 {
34     scanf("%d", &n);
35     for (int i = 1; i <= n; i++)
36         scanf("%d", &a[i]);
37     return 0;
38 }
39

```

点, 线段, 极角排序

```

1  #include <bits/stdc++.h>
2  using namespace std;
3  #define ubit(a) (64 - __builtin_clzll(a))
4  // 获得最高的1的位置
5  #define popcount(a) __builtin_popcountll(a)
6  // 第一个1的个数
7  typedef double db;
8  const db EPS = 1e-9;
9  // 点(x,y)之间的操作 }
10
11 // TOP1
12 // 点
13 inline int sign(db a) { return a < -EPS ? -1 : a > EPS; }
14 // 1)a<0 -> -1 2)a>0 ->1 3)a=0->0
15 inline int cmp(db a, db b) { return sign(a - b); }
16 struct P
17 {
18     db x, y;

```

```

19     P() {}
20     P(db _x, db _y) : x(_x), y(_y) {}
21     P operator+(P p) { return {x + p.x, y + p.y}; }
22     P operator-(P p) { return {x - p.x, y - p.y}; }
23     P operator*(db d) { return {x * d, y * d}; }
24     P operator/(db d) { return {x / d, y / d}; }
25     // 点的加减乘除基本运算
26     bool operator<(P p) const
27     { // 两者谁更小
28         int c = cmp(x, p.x);
29         if (c)
30             return c == -1;
31         return cmp(y, p.y) == -1;
32     }
33     bool operator==(P o) const
34     { // 是否相等
35         return cmp(x, o.x) == 0 && cmp(y, o.y) == 0;
36     }
37     db dot(P p) { return x * p.x + y * p.y; } // 点积
38     db det(P p) { return x * p.y - y * p.x; } // 叉积 = |a||b|sin(o) o为a到b逆
    时针的那个夹角
39     db distTo(P p) { return (*this - p).abs(); } // 求点到点的距离
40     db alpha() { return atan2(y, x); } // 求极角(-pai,pai)(x负半轴为-
    pai逆时针到pai)
41     db abs() { return sqrt(abs2()); } // 求两点距离
42     db abs2() { return x * x + y * y; } // 两点距离平方
43     P rot90() { return P(-y, x); } // 逆时针旋转90度
44     P unit() { return *this / abs(); }
45     int quad() const { return sign(y) == 1 || (sign(y) == 0 && sign(x) >= 0); }
46     P rot(db an) { return {x * cos(an) - y * sin(an), x * sin(an) + y *
    cos(an)}; } // 逆时针转an的角度(an弧度制)
47 };
48
49 // Top2
50 // 线，随便两个点表示一段线，要用点的包装
51 #define cross(p1, p2, p3) ((p2.x - p1.x) * (p3.y - p1.y) - (p3.x - p1.x) *
    (p2.y - p1.y)) // (p1p2 叉乘 p1p3)
52 #define crossOp(p1, p2, p3) sign(cross(p1, p2, p3))
    // 0是三点共线 1是p1->p2->p3 是逆时针，-1则是顺时针
53
54 // 直线 p1p2, q1q2 是否恰有一个交点 1表示有交点 0表示无交点
55 bool chkLL(P p1, P p2, P q1, P q2)
56 {
57     db a1 = cross(q1, q2, p1), a2 = -cross(q1, q2, p2);
58     return sign(a1 + a2) != 0;
59 }
60

```

```

61 // 求直线 p1p2, q1q2 的交点 (要先用上面的判断是否有交点)
62 P isLL(P p1, P p2, P q1, P q2)
63 {
64     db a1 = cross(q1, q2, p1), a2 = -cross(q1, q2, p2);
65     return (p1 * a2 + p2 * a1) / (a1 + a2);
66 }
67
68 // 判断区间 [l1, r1], [l2, r2] 是否相交
69 bool intersect(db l1, db r1, db l2, db r2)
70 {
71     if (l1 > r1)
72         swap(l1, r1);
73     if (l2 > r2)
74         swap(l2, r2);
75     return !(cmp(r1, l2) == -1 || cmp(r2, l1) == -1);
76 }
77
78 // 线段 p1p2, q1q2 是否相交
79 bool isSS(P p1, P p2, P q1, P q2)
80 {
81     return intersect(p1.x, p2.x, q1.x, q2.x) && intersect(p1.y, p2.y, q1.y,
82         q2.y) &&
83         crossOp(p1, p2, q1) * crossOp(p1, p2, q2) <= 0 && crossOp(q1, q2,
84         p1) * crossOp(q1, q2, p2) <= 0;
85 }
86
87 // 线段 p1p2, q1q2 严格相交 (不交在端点)
88 bool isSS_strict(P p1, P p2, P q1, P q2)
89 {
90     return crossOp(p1, p2, q1) * crossOp(p1, p2, q2) < 0 && crossOp(q1, q2,
91         p1) * crossOp(q1, q2, p2) < 0;
92 }
93
94 // m 是否在 a 和 b 线段之间
95 bool isMiddle(db a, db m, db b)
96 {
97     return sign(a - m) == 0 || sign(b - m) == 0 || (a < m != b < m);
98 }
99
100 // 点m是否在这个ab这个矩形内
101 bool isMiddle(P a, P m, P b)
102 {
103     return isMiddle(a.x, m.x, b.x) && isMiddle(a.y, m.y, b.y);
104 }
105
106 // 点 p 是否在线段 p1p2 上
107 bool onSeg(P p1, P p2, P q)
108 { // 可能有精度问题

```

```

105     return crossOp(p1, p2, q) == 0 && isMiddle(p1, q, p2);
106 }
107 // q1q2 和 p1p2 的交点 在 p1p2 上? 确定的时候不需要crossOp(p1,p2,q) == 0
108
109 // 点 p 严格在 p1p2 上
110 bool onSeg_strict(P p1, P p2, P q)
111 {
112     return crossOp(p1, p2, q) == 0 && sign((q - p1).dot(p1 - p2)) * sign((q -
        p2).dot(p1 - p2)) < 0;
113 }
114
115 // 求 q 到 直线p1p2 的投影 (垂足)  p1 != p2
116 P proj(P p1, P p2, P q)
117 {
118     P dir = p2 - p1;
119     return p1 + dir * (dir.dot(q - p1) / dir.abs2());
120 }
121
122 // 求 q 以 直线p1p2 为轴的反射  p1 != p2
123 P reflect(P p1, P p2, P q)
124 {
125     return proj(p1, p2, q) * 2 - q;
126 }
127
128 // 求 q 到 线段p1p2 的最小距离
129 db nearest(P p1, P p2, P q)
130 {
131     if (p1 == p2)
132         return p1.distTo(q);
133     P h = proj(p1, p2, q);
134     if (isMiddle(p1, h, p2))
135         return q.distTo(h);
136     return min(p1.distTo(q), p2.distTo(q));
137 }
138
139 // 求 线段p1p2 与 线段q1q2 的距离
140 db disSS(P p1, P p2, P q1, P q2)
141 {
142     if (isSS(p1, p2, q1, q2))
143         return 0;
144     return min(min(nearest(p1, p2, q1), nearest(p1, p2, q2)), min(nearest(q1,
        q2, p1), nearest(q1, q2, p2)));
145 }
146
147 // TOP3
148 // 极角排序 (x负半轴到逆时针排序)
149 const int N = 1e5 + 10;

```

```

150 P p[N];
151 int n;
152 void Sort()
153 {
154     sort(p + 1, p + n + 1, [&](P &a, P &b)
155         {
156             int qa=a.quad(),qb=b.quad();
157             if(qa!=qb)return qa<qb;
158             else return sign(a.det(b))>0; });
159 }
160 signed main()
161 {
162     return 0;
163 }
164

```

动态开点线段树

```

1  #include <bits/stdc++.h>
2  using namespace std;
3  int read()
4  {
5      int num = 0;
6      bool flag = 1;
7      char c = getchar();
8      for (; c < '0' || c > '9'; c = getchar())
9          if (c == '-')
10             flag = 0;
11     for (; c >= '0' && c <= '9'; c = getchar())
12         num = (num << 1) + (num << 3) + c - '0';
13     return flag ? num : -num;
14 }
15 const int N = 1e5 + 10;
16 int n, q;
17 #define ll long long
18 struct Tree
19 {
20     struct cow
21     {
22         int ls, rs;
23         ll sum, lazy;
24     } tree[N << 2];
25     int tot = 1;
26     #define ls(p) tree[p].ls
27     #define rs(p) tree[p].rs

```



```

28 #define sum(p) tree[p].sum
29 #define lazy(p) tree[p].lazy
30 void chck_new(int p)
31 {
32     if (!ls(p))
33         ls(p) = ++tot;
34     if (!rs(p))
35         rs(p) = ++tot;
36 }
37 void lazytime(int p, int l, int r)
38 {
39     int mid = l + r - 1 >> 1;
40     sum(ls(p)) += lazy(p) * (mid - l + 1);
41     sum(rs(p)) += lazy(p) * (r - mid);
42     lazy(ls(p)) += lazy(p);
43     lazy(rs(p)) += lazy(p);
44     lazy(p) = 0;
45 }
46 void add(int p, int l, int r, int L, int R, ll d)
47 {
48     if (l >= L && r <= R)
49     {
50         sum(p) += d * (r - l + 1);
51         lazy(p) += d;
52         return;
53     }
54     chck_new(p);
55     lazytime(p, l, r);
56     int mid = l + r - 1 >> 1;
57     if (L <= mid)
58         add(ls(p), l, mid, L, R, d);
59     if (R > mid)
60         add(rs(p), mid + 1, r, L, R, d);
61     sum(p) = sum(ls(p)) + sum(rs(p));
62 }
63 ll ask(int p, int l, int r, int L, int R)
64 {
65     if (l >= L && r <= R)
66         return sum(p);
67     chck_new(p);
68     lazytime(p, l, r);
69     int mid = l + r - 1 >> 1;
70     ll ans = 0;
71     if (L <= mid)
72         ans += ask(ls(p), l, mid, L, R);
73     if (R > mid)
74         ans += ask(rs(p), mid + 1, r, L, R);

```

```

75         return ans;
76     }
77 } T;
78 int main()
79 {
80     n = read();
81     q = read();
82     for (int i = 1; i <= n; i++)
83         T.add(1, 1, n, i, i, read());
84     for (int i = 1; i <= q; i++)
85     {
86         int op = read();
87         if (op == 1)
88         {
89             int x = read(), y = read(), k = read();
90             T.add(1, 1, n, x, y, k);
91         }
92         else
93         {
94             int x = read(), y = read();
95             printf("%lld\n", T.ask(1, 1, n, x, y));
96         }
97     }
98     return 0;
99 }

```

对拍

```

1  #include <bits/stdc++.h>
2  using namespace std;
3  int main()
4  {
5      int T = 0;
6      while (1)
7      {
8          T++;
9          system("C:\\randsequence.exe"); // rand
10         double stb = clock();
11         system("C:\\1.exe"); // 暴力
12         double edb = clock();
13         double stz = clock();
14         system("C:\\2.exe"); // 正解
15         double edz = clock();
16         if (system("fc C:\\1.out C:\\2.out"))
17         {

```

```

18         cout << "WA";
19         return 0;
20     }
21     else
22     {
23         printf("测试点: %d \n 暴力: %.0lfms\n 正解: %.0lfms\n", T, edb -
stb, edz - stz);
24     }
25 }
26 }
27

```

多边形

```

1 #include <bits/stdc++.h>
2 using namespace std;
3 typedef double db;
4 const db EPS = 1e-9;
5
6 inline int sign(db a) { return a < -EPS ? -1 : a > EPS; }
7
8 inline int cmp(db a, db b) { return sign(a - b); }
9
10 struct P
11 {
12     db x, y;
13     P() {}
14     P(db _x, db _y) : x(_x), y(_y) {} // 点的加减乘除基本运算
15     P operator+(P p) { return {x + p.x, y + p.y}; }
16     P operator-(P p) { return {x - p.x, y - p.y}; }
17     P operator*(db d) { return {x * d, y * d}; }
18     P operator/(db d) { return {x / d, y / d}; }
19
20     bool operator<(P p) const
21     { // 先比较x大小, 再比较y 返回小的
22         int c = cmp(x, p.x);
23         if (c)
24             return c == -1;
25         return cmp(y, p.y) == -1;
26     }
27
28     bool operator==(P o) const
29     {
30         return cmp(x, o.x) == 0 && cmp(y, o.y) == 0;
31     }

```

```

32
33     db dot(P p) { return x * p.x + y * p.y; }
34     db det(P p) { return x * p.y - y * p.x; } // 叉积 正为逆时针方向
35
36     db distTo(P p) { return (*this - p).abs(); } // 求点到点的距离
37     db alpha() { return atan2(y, x); } // 求极角
38     void read() { cin >> x >> y; }
39     void write() { cout << "(" << x << "," << y << ")" << endl; }
40     db abs() { return sqrt(abs2()); } // 求两点距离
41     db abs2() { return x * x + y * y; }
42     P rot90() { return P(-y, x); } // 旋转 90度
43     P unit() { return *this / abs(); } //
44     int quad() const { return sign(y) == 1 || (sign(y) == 0 && sign(x) >= 0); }
45     P rot(db an) { return {x * cos(an) - y * sin(an), x * sin(an) + y *
cos(an)}; }
46 };
47 #define cross(p1, p2, p3) ((p2.x - p1.x) * (p3.y - p1.y) - (p3.x - p1.x) *
(p2.y - p1.y))
48 #define crossOp(p1, p2, p3) sign(cross(p1, p2, p3))
49 bool isMiddle(db a, db m, db b)
50 {
51     return sign(a - m) == 0 || sign(b - m) == 0 || (a < m != b < m);
52 }
53
54 bool isMiddle(P a, P m, P b)
55 {
56     return isMiddle(a.x, m.x, b.x) && isMiddle(a.y, m.y, b.y);
57 }
58 // 点 p 在线段 p1p2 上
59 bool onSeg(P p1, P p2, P q)
60 {
61     return crossOp(p1, p2, q) == 0 && isMiddle(p1, q, p2);
62 }
63 // 求直线 p1p2, q1q2 的交点
64 P isLL(P p1, P p2, P q1, P q2)
65 {
66     db a1 = cross(q1, q2, p1), a2 = -cross(q1, q2, p2);
67     return (p1 * a2 + p2 * a1) / (a1 + a2);
68 }
69 //-----这里开始是多边形, 前面都是点线段的代码-----
70
71 db area(vector<P> ps)
72 {
73     db res = 0;
74     for (int i = 0; i < ps.size(); i++)
75         res += ps[i].det(ps[(i + 1) % ps.size()]);
76     return abs(res) / 2;

```

```

77 } // 面积, 只有点按照逆时针或顺时针才能这么算
78
79 int contain(vector<P> ps, P p)
80 { // 2:inside, 1:on_seg, 0:outside
81     int n = ps.size(), ret = 0;
82     for (int i = 0; i < n; i++)
83     {
84         P u = ps[i], v = ps[(i + 1) % n];
85         if (onSeg(u, v, p))
86             return 1;
87         if (cmp(u.y, v.y) <= 0)
88             swap(u, v);
89         if (cmp(p.y, u.y) > 0 || cmp(p.y, v.y) <= 0)
90             continue;
91         ret ^= crossOp(p, u, v) > 0;
92     }
93     return ret * 2;
94 } // 点包含
95
96 // 凸包: 包含所有点的一个凸多边形, 且顶点都是给定点
97 vector<P> convexHull(vector<P> ps)
98 {
99     int n = ps.size();
100     if (n <= 1)
101         return ps;
102     sort(ps.begin(), ps.end());
103     vector<P> qs(n * 2);
104     int k = 0;
105     for (int i = 0; i < n; qs[k++] = ps[i++])
106         while (k > 1 && crossOp(qs[k - 2], qs[k - 1], ps[i]) <= 0)
107             --k;
108     for (int i = n - 2, t = k; i >= 0; qs[k++] = ps[i--])
109         while (k > t && crossOp(qs[k - 2], qs[k - 1], ps[i]) <= 0)
110             --k;
111     qs.resize(k - 1);
112     return qs;
113 } // 严格的凸包 (不含180°角)
114 vector<P> convexHullNonStrict(vector<P> ps)
115 {
116     int n = ps.size();
117     if (n <= 1)
118         return ps;
119     sort(ps.begin(), ps.end());
120     vector<P> qs(n * 2);
121     int k = 0;
122     for (int i = 0; i < n; qs[k++] = ps[i++])
123         while (k > 1 && crossOp(qs[k - 2], qs[k - 1], ps[i]) < 0)

```

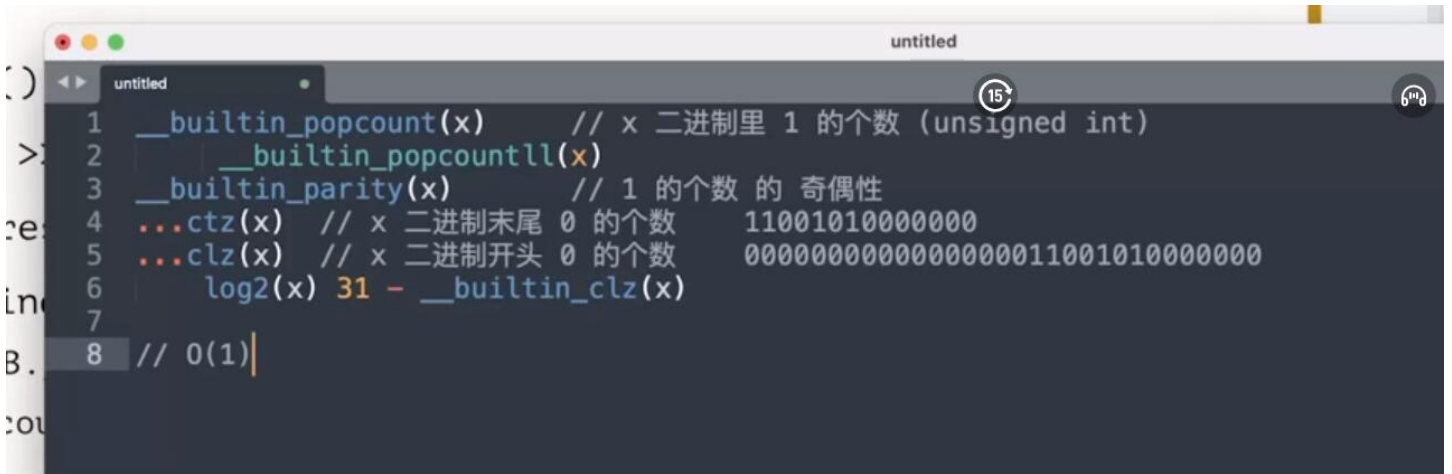
```

124         --k;
125         for (int i = n - 2, t = k; i >= 0; qs[k++] = ps[i--])
126             while (k > t && crossOp(qs[k - 2], qs[k - 1], ps[i]) < 0)
127                 --k;
128         qs.resize(k - 1);
129         return qs;
130 } // 不严格的凸包 (含180°角)
131 // 注意这个一定要去重
132
133 vector<P> convexCut(const vector<P> &ps, P q1, P q2)
134 {
135     vector<P> qs;
136     int n = ps.size();
137     for (int i = 0; i < n; i++)
138     {
139         P p1 = ps[i], p2 = ps[(i + 1) % n];
140         int d1 = crossOp(q1, q2, p1), d2 = crossOp(q1, q2, p2);
141         if (d1 >= 0)
142             qs.push_back(p1);
143         if (d1 * d2 < 0)
144             qs.push_back(isLL(p1, p2, q1, q2));
145     }
146     return qs;
147 } // q1 q2分割凸多边形ps后变成的新凸多边形
148
149 db convexDiameter(vector<P> ps)
150 {
151     int n = ps.size();
152     if (n <= 1)
153         return 0;
154     int is = 0, js = 0;
155     for (int k = 1; k < n; k++)
156         is = ps[k] < ps[is] ? k : is, js = ps[js] < ps[k] ? k : js;
157     int i = is, j = js;
158     db ret = ps[i].distTo(ps[j]);
159     do
160     {
161         if ((ps[(i + 1) % n] - ps[i]).det(ps[(j + 1) % n] - ps[j]) >= 0)
162             (++j) %= n;
163         else
164             (++i) %= n;
165         ret = max(ret, ps[i].distTo(ps[j]));
166     } while (i != is || j != js);
167     return ret;
168 } // 这些点中最远的两个点的点距离
169 signed main()
170 {

```

```
171     return 0;
172 }
173
```

二进制



The screenshot shows a code editor window titled 'untitled' with a dark theme. It contains C++ code demonstrating various built-in functions for bit manipulation. The code is as follows:

```
1  __builtin_popcount(x)      // x 二进制里 1 的个数 (unsigned int)
2  __builtin_popcountll(x)
3  __builtin_parity(x)        // 1 的个数 的 奇偶性
4  ...ctz(x) // x 二进制末尾 0 的个数    11001010000000
5  ...clz(x) // x 二进制开头 0 的个数    00000000000000000011001010000000
6  log2(x) 31 - __builtin_clz(x)
7
8  // 0(1)
```

割边

```
1  #include <bits/stdc++.h>
2  using namespace std;
3  const int N = 1e5 + 10;
4  const int inf = 1e9;
5  int read()
6  {
7      int num = 0;
8      bool flag = 1;
9      char c = getchar();
10     for (; c < '0' || c > '9'; c = getchar())
11         if (c == '-')
12             flag = 0;
13     for (; c >= '0' && c <= '9'; c = getchar())
14         num = (num << 1) + (num << 3) + c - '0';
15     return flag ? num : -num;
16 }
17 vector<pair<int, int>> e[N];
18 int n, m, dfn[N], low[N], idx;
19 vector<int> bridge;
20 void Tarjan(int x, int id)
21 {
22     dfn[x] = low[x] = ++idx;
23     for (auto it : e[x])
24     {
25         int y = it.first, id2 = it.second;
```

```

26         if (!dfn[y])
27         {
28             Tarjan(y, id2);
29             low[x] = min(low[x], low[y]);
30         }
31         else if (id != id2)
32             low[x] = min(low[x], dfn[y]);
33     }
34     if (low[x] == dfn[x] && id)
35         bridge.push_back(id);
36 }
37 signed main()
38 {
39     n = read();
40     m = read();
41     for (int i = 1; i <= m; i++)
42     {
43         int x = read(), y = read();
44         e[x].push_back({y, i});
45         e[y].push_back({x, i});
46     }
47     for (int i = 1; i <= n; i++)
48         if (!dfn[i])
49             Tarjan(i, 0);
50     sort(bridge.begin(), bridge.end());
51     printf("%d\n", bridge.size());
52     for (auto num : bridge)
53         printf("%d ", num);
54     return 0;
55 }
56

```

割点

```

1  #include <bits/stdc++.h>
2  using namespace std;
3  const int N = 1e5 + 10;
4  const int inf = 1e9;
5  int read()
6  {
7      int num = 0;
8      bool flag = 1;
9      char c = getchar();
10     for (; c < '0' || c > '9'; c = getchar())
11         if (c == '-')

```



```

12         flag = 0;
13         for (; c >= '0' && c <= '9'; c = getchar())
14             num = (num << 1) + (num << 3) + c - '0';
15         return flag ? num : -num;
16     }
17     vector<int> e[N];
18     int n, m, dfn[N], low[N], idx, cut[N], rt;
19     void Tarjan(int x, int fa)
20     {
21         dfn[x] = low[x] = ++idx;
22         int ct = 0;
23         for (auto y : e[x])
24         {
25             if (!dfn[y])
26             {
27                 Tarjan(y, fa);
28                 ct++;
29                 if (low[y] >= dfn[x])
30                 {
31                     cut[x] = 1;
32                 }
33                 low[x] = min(low[x], low[y]);
34             }
35             else if (y != fa)
36                 low[x] = min(low[x], dfn[y]);
37         }
38         if (x == rt && ct <= 1)
39             cut[x] = 0;
40     }
41     signed main()
42     {
43         n = read();
44         m = read();
45         for (int i = 1; i <= m; i++)
46         {
47             int x = read(), y = read();
48             e[x].push_back(y);
49             e[y].push_back(x);
50         }
51         for (int i = 1; i <= n; i++)
52             if (!dfn[i])
53                 rt = i, Tarjan(i, 0);
54         int sum = 0;
55         for (int i = 1; i <= n; i++)
56             sum += cut[i];
57         printf("%d\n", sum);
58         for (int i = 1; i <= n; i++)

```

```

59         if (cut[i])
60             printf("%d ", i);
61     return 0;
62 }
63

```

卢卡斯

二、卢卡斯定理

1、定义

卢卡斯定理如下：

$$C_n^m \equiv C_{n/p}^{m/p} C_{n \bmod p}^{m \bmod p} (\bmod p)$$

其中 p 为素数。

莫队(带时间修改)

```

1  #include <bits/stdc++.h>
2  using namespace std;
3  int read()
4  {
5      int num = 0;
6      bool flag = 1;
7      char c = getchar();
8      for (; c < '0' || c > '9'; c = getchar())
9          ;
10     for (; c >= '0' && c <= '9'; c = getchar())
11         num = (num << 1) + (num << 3) + c - '0';
12     return flag ? num : -num;
13 }
14 const int N = 1e5 + 10;
15 map<int, int> lsh;
16 struct Q
17 {
18     int l, r, id;
19 } Qu[N], Ch[N];

```

```

20 int v[N];
21 bool cmpQu(Q a, Q b)
22 {
23     if (v[a.l] ^ v[b.l])
24         return a.l < b.l;
25     if (v[a.r] ^ v[b.r])
26         if (v[a.l] & 1)
27             return a.r < b.r;
28         else
29             return a.r > b.r;
30     return a.id < b.id;
31 }
32 int cnt[N << 1], cnt_cnt[N], ans[N];
33 int n, q, a[N], tot, Qu_tot, Ch_tot;
34 void add(int x)
35 {
36     cnt_cnt[cnt[a[x]]]--;
37     cnt_cnt[++cnt[a[x]]]++;
38 }
39 void del(int x)
40 {
41     cnt_cnt[cnt[a[x]]--]--;
42     cnt_cnt[cnt[a[x]]]++;
43 }
44 void Time(int t, int i)
45 {
46     int l = Qu[i].l, r = Qu[i].r;
47     int pos = Ch[t].l, val = Ch[t].r;
48     if (pos >= l && pos <= r)
49     {
50         cnt_cnt[cnt[a[pos]]--]--;
51         cnt_cnt[cnt[a[pos]]]++;
52         cnt_cnt[cnt[val]]--;
53         cnt_cnt[++cnt[val]]++;
54     }
55     swap(Ch[t].r, a[pos]);
56 }
57 int main()
58 {
59     n = read();
60     q = read();
61     for (int i = 1; i <= n; i++)
62     {
63         a[i] = read();
64         if (!lsh[a[i]])
65             lsh[a[i]] = ++tot;
66         a[i] = lsh[a[i]];

```

```

67     }
68     for (int i = 1; i <= q; i++)
69     {
70         int t = read();
71         if (t == 1)
72         {
73             Qu[++Qu_tot].l = read();
74             Qu[Qu_tot].r = read();
75             Qu[Qu_tot].id = i;
76         }
77         else
78         {
79             Ch[++Ch_tot].l = read();
80             Ch[Ch_tot].r = read();
81             Ch[Ch_tot].id = i;
82             if (!lsh[Ch[Ch_tot].r])
83                 lsh[Ch[Ch_tot].r] = ++tot;
84             Ch[Ch_tot].r = lsh[Ch[Ch_tot].r];
85         }
86     }
87     int gh = pow(n, 2.0 / 3.0), l = 1, r = 0, t = 0;
88     for (int i = 1; i <= n; i++)
89         v[i] = (i - 1) / gh + 1;
90     sort(Qu + 1, Qu + Qu_tot + 1, cmpQu);
91     Ch[++Ch_tot].id = N;
92     //
93     for (int i = 1; i <= Qu_tot; i++)
94     {
95         int _l = Qu[i].l, _r = Qu[i].r,
96             _t = Qu[i].id, Ans = 1;
97         while (r < _r)
98             add(++r);
99         while (l > _l)
100             add(--l);
101         while (l < _l)
102             del(l++);
103         while (r > _r)
104             del(r--);
105         while (_t > Ch[t + 1].id)
106             Time(++t, i);
107         while (_t < Ch[t].id)
108             Time(t--, i);
109         while (cnt_cnt[Ans])
110             Ans++;
111         ans[Qu[i].id] = Ans;
112     }
113     for (int i = 1; i <= q; i++)

```

```

114         if (ans[i])
115             printf("%d\n", ans[i]);
116     return 0;
117 }
118

```

强连通分量

```

1  #include <bits/stdc++.h>
2  using namespace std;
3  const int N = 1e5 + 10;
4  const int inf = 1e9;
5  int read()
6  {
7      int num = 0;
8      bool flag = 1;
9      char c = getchar();
10     for (; c < '0' || c > '9'; c = getchar())
11         if (c == '-')
12             flag = 0;
13     for (; c >= '0' && c <= '9'; c = getchar())
14         num = (num << 1) + (num << 3) + c - '0';
15     return flag ? num : -num;
16 }
17 vector<int> e[N];
18 int n, m, dfn[N], low[N], v[N], idx;
19 vector<vector<int>> scc;
20 stack<int> st;
21 void Tarjan(int x)
22 {
23     dfn[x] = low[x] = ++idx;
24     st.push(x);
25     v[x] = 1;
26     for (auto y : e[x])
27     {
28         if (!dfn[y])
29         {
30             Tarjan(y);
31             low[x] = min(low[x], low[y]);
32         }
33         if (v[y])
34             low[x] = min(low[x], dfn[y]);
35     }
36     if (low[x] == dfn[x])
37     {

```

```

38     vector<int> c;
39     while (1)
40     {
41         int now = st.top();
42         st.pop();
43         v[now] = 0;
44         c.push_back(now);
45         if (now == x)
46             break;
47     }
48     sort(c.begin(), c.end());
49     scc.push_back(c);
50 }
51 }
52 signed main()
53 {
54     n = read();
55     m = read();
56     for (int i = 1; i <= m; i++)
57     {
58         int x = read(), y = read();
59         e[x].push_back(y);
60     }
61     for (int i = 1; i <= n; i++)
62         if (!dfn[i])
63             Tarjan(i);
64     sort(scc.begin(), scc.end());
65     for (auto now : scc)
66     {
67         for (auto x : now)
68             printf("%d ", x);
69         printf("\n");
70     }
71     return 0;
72 }
73

```

树链剖分

```

1  #include <bits/stdc++.h>
2  using namespace std;
3  int read()
4  {
5      int num = 0;
6      bool flag = 1;

```

```

7     char c = getchar();
8     for (; c < '0' || c > '9'; c = getchar())
9         if (c == '-')
10             flag = 0;
11     for (; c >= '0' && c <= '9'; c = getchar())
12         num = (num << 1) + (num << 3) + c - '0';
13     return flag ? num : -num;
14 }
15 const int N = 1e5 + 10;
16 int n;
17 struct Chain
18 {
19     struct Tr
20     {
21         int Max, sum, l, r;
22 #define l(p) tree[p].l
23 #define r(p) tree[p].r
24 #define Max(p) tree[p].Max
25 #define sum(p) tree[p].sum
26     } tree[N << 2];
27     void build(int p, int l, int r)
28     {
29         l(p) = l;
30         r(p) = r;
31         if (l == r)
32         {
33             sum(p) = Max(p) = w[id[l]];
34             return;
35         }
36         int mid = l + r >> 1;
37         build(p << 1, l, mid);
38         build(p << 1 | 1, mid + 1, r);
39         sum(p) = sum(p << 1) + sum(p << 1 | 1);
40         Max(p) = max(Max(p << 1), Max(p << 1 | 1));
41     }
42     void change(int p, int x, int z)
43     {
44         if (l(p) == r(p))
45         {
46             sum(p) = Max(p) = z;
47             return;
48         }
49         int mid = l(p) + r(p) >> 1;
50         if (x <= mid)
51             change(p << 1, x, z);
52         else
53             change(p << 1 | 1, x, z);

```

```

54     sum(p) = sum(p << 1) + sum(p << 1 | 1);
55     Max(p) = max(Max(p << 1), Max(p << 1 | 1));
56 }
57 int ask_Max(int p, int l, int r)
58 {
59     if (l(p) >= l && r(p) <= r)
60         return Max(p);
61     int mid = l(p) + r(p) >> 1, Max = -INT_MAX;
62     if (l <= mid)
63         Max = max(Max, ask_Max(p << 1, l, r));
64     if (r > mid)
65         Max = max(Max, ask_Max(p << 1 | 1, l, r));
66     return Max;
67 }
68 int ask_sum(int p, int l, int r)
69 {
70     if (l(p) >= l && r(p) <= r)
71         return sum(p);
72     int mid = l(p) + r(p) >> 1, sum = 0;
73     if (l <= mid)
74         sum += ask_sum(p << 1, l, r);
75     if (r > mid)
76         sum += ask_sum(p << 1 | 1, l, r);
77     return sum;
78 }
79
80 struct cow
81 {
82     int x, y;
83 } e[N << 1];
84 int head[N], tot, w[N], id[N], dfn[N];
85 void inse(int xxxx, int yyyy)
86 {
87     e[++tot].x = head[xxxx];
88     head[xxxx] = tot;
89     e[tot].y = yyyy;
90 }
91 int fa[N], dep[N], sz[N], hv[N];
92 void dfs1(int x, int f)
93 {
94     dep[x] = dep[f] + 1;
95     sz[x] = 1;
96     fa[x] = f;
97     for (int i = head[x]; i; i = e[i].x)
98     {
99         int y = e[i].y;
100         if (y == f)

```



```

101         continue;
102         dfs1(y, x);
103         sz[x] += sz[y];
104         if (sz[y] > sz[hv[x]])
105             hv[x] = y;
106     }
107 }
108 int top[N];
109 void dfs2(int x, int t)
110 {
111     top[x] = t;
112     dfn[x] = ++tot;
113     id[tot] = x;
114     if (hv[x] == 0)
115         return;
116     dfs2(hv[x], t);
117     for (int i = head[x]; i; i = e[i].x)
118     {
119         int y = e[i].y;
120         if (y == fa[x] || y == hv[x])
121             continue;
122         dfs2(y, y);
123     }
124 }
125 void build_tree()
126 {
127     for (int i = 1; i < n; i++)
128     {
129         int x = read(), y = read();
130         inse(x, y);
131         inse(y, x);
132     }
133     for (int i = 1; i <= n; i++)
134         w[i] = read();
135     dfs1(1, 0);
136     tot = 0;
137     dfs2(1, 1);
138     build(1, 1, n);
139 }
140 int Ask_Max(int x, int y)
141 {
142     int Max = -INT_MAX;
143     while (top[x] != top[y])
144     {
145         if (dep[top[x]] < dep[top[y]])
146             swap(x, y);
147         Max = max(Max, ask_Max(1, dfn[top[x]], dfn[x]));

```

```

148         x = fa[top[x]];
149     }
150     if (dep[x] < dep[y])
151         swap(x, y);
152     Max = max(Max, ask_Max(1, dfn[y], dfn[x]));
153     return Max;
154 }
155 int Ask_sum(int x, int y)
156 {
157     int sum = 0;
158     while (top[x] != top[y])
159     {
160         if (dep[top[x]] < dep[top[y]])
161             swap(x, y);
162         sum += ask_sum(1, dfn[top[x]], dfn[x]);
163         x = fa[top[x]];
164     }
165     if (dep[x] < dep[y])
166         swap(x, y);
167     sum += ask_sum(1, dfn[y], dfn[x]);
168     return sum;
169 }
170 } T;
171 signed main()
172 {
173     n = read();
174     T.build_tree();
175     int q = read();
176     while (q--)
177     {
178         char op[10];
179         scanf("%s", op + 1);
180         if (op[1] == 'Q')
181         {
182             if (op[2] == 'M')
183                 printf("%d\n", T.Ask_Max(read(), read()));
184             else
185                 printf("%d\n", T.Ask_sum(read(), read()));
186         }
187         else
188         {
189             int l = T.dfn[read()], t = read();
190             T.change(1, l, t);
191         }
192     }
193     return 0;
194 }

```

线性基

```

1  #include <bits/stdc++.h>
2  using namespace std;
3  #define ll long long
4  ll read()
5  {
6      ll num = 0;
7      bool flag = 1;
8      char c = getchar();
9      for (; c < '0' || c > '9'; c = getchar())
10         if (c == '-')
11             flag = 0;
12     for (; c >= '0' && c <= '9'; c = getchar())
13         num = (num << 1) + (num << 3) + c - '0';
14     return flag ? num : -num;
15 }
16 const int B = 60;
17 struct linear_basis
18 {
19     ll num[B + 10];
20     int hv;
21     bool insert(ll x)
22     {
23         for (int i = B - 1; i >= 0; i--)
24         {
25             if (x >> i & 1)
26             {
27                 if (num[i] == 0)
28                 {
29                     num[i] = x;
30                     hv++;
31                     return 1;
32                 }
33                 x ^= num[i];
34             }
35         }
36         return 0;
37     }
38     ll querymin(ll x)
39     {
40         for (int i = B - 1; i >= 0; i--)
41             x = min(x, x ^ num[i]);

```

```

42     return x;
43 }
44 ll querymax(ll x)
45 {
46     for (int i = B - 1; i >= 0; i--)
47         x = max(x, x ^ num[i]);
48     return x;
49 }
50 // x 的 第 C 位
51 } ba;
52 ** L: u n          u          0          **//
53 //          insert      false g          0
54 int n;
55 ll k;
56 signed main()
57 {
58     n = read();
59     k = read();
60     ll now = 1;
61     for (int i = 1; i <= n; i++)
62         if (!ba.insert(read()))
63             now *= 2;
64     k = k / now;
65     ll ans = 0;
66     for (int i = B - 1; i >= 0; i--)
67     {
68         if (ba.num[i] == 0)
69             continue;
70         if (k >= (1ll << ba.hv - 1))
71         {
72             k -= (1ll << ba.hv - 1);
73             ans = max(ans, ans ^ ba.num[i]);
74         }
75         else
76             ans = min(ans, ans ^ ba.num[i]);
77         ba.hv--;
78     }
79     printf("%lld\n", ans);
80     return 0;
81 }
82

```

行列式计算

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

```

2
3 #define IOS ios::sync_with_stdio(false), cin.tie(0)
4 #define ll long long
5 using namespace std;
6 const int N = 205;
7 ll g[N][N];
8
9 ll Rainbow(int n, int mod)
10 {
11     ll ans = 1;
12     for (int i = 1; i <= n; i++)
13         for (int j = 1; j <= n; j++)
14             g[i][j] %= mod;
15     for (int i = 1; i <= n; i++)
16     {
17         for (int j = i + 1; j <= n; j++)
18         {
19             int x = i, y = j;
20             while (g[x][i])
21             {
22                 int t = g[y][i] / g[x][i];
23                 for (int k = i; k <= n; k++)
24                     g[y][k] = (g[y][k] - t * g[x][k]) % mod;
25                 swap(x, y);
26             }
27             if (x == i)
28             {
29                 for (int k = i; k <= n; k++)
30                     swap(g[j][k], g[i][k]);
31                 ans = -ans;
32             }
33         }
34         if (g[i][i] == 0)
35             return 0;
36         ans = ans * g[i][i] % mod;
37     }
38     if (ans < 0)
39         ans += mod;
40     return ans;
41     // 近似于 $n^3$ 
42 }
43
44 int main()
45 {
46     int n, m, mod;
47     cin >> n >> m >> mod;
48     for (int i = 0; i < m; i++)

```

```

49     {
50         int u, v;
51         cin >> u >> v;
52         g[u][v]--, g[v][u]--;
53         g[u][u]++, g[v][v]++;
54     }
55     cout << Rainbow(n - 1, mod) << "\n";
56     return 0;
57 }

```

最小费用流

```

1  #include <bits/stdc++.h>
2  using namespace std;
3  int read()
4  {
5      int num = 0;
6      bool flag = 1;
7      char c = getchar();
8      for (; c < '0' || c > '9'; c = getchar())
9          if (c == '-')
10             flag = 0;
11     for (; c >= '0' && c <= '9'; c = getchar())
12         num = (num << 1) + (num << 3) + c - '0';
13     return flag ? num : -num;
14 }
15 const int N = 20100;
16 const int M = 201000;
17 template <typename T>
18 struct MinCostGraph
19 {
20     int s, t, vtot, head[N], etot;
21     T dis[N], flow, cost;
22     int pre[N];
23     bool vis[N];
24     struct edge
25     {
26         int v, nxt;
27         T f, c;
28     } e[M << 2];
29     void addedge(int u, int v, T f, T c, T f2 = 0)
30     {
31         e[etot] = {v, head[u], f, c};
32         head[u] = etot++;
33         e[etot] = {u, head[v], f2, -c};

```

```

34     head[v] = etot++;
35 }
36 bool spfa()
37 {
38     T inf = numeric_limits<T>::max() / 2;
39     for (int i = 1; i <= vtot; ++i)
40     {
41         dis[i] = inf;
42         vis[i] = 0;
43         pre[i] = -1;
44     }
45     dis[s] = 0;
46     vis[s] = 1;
47     queue<int> q;
48     q.push(s);
49     while (!q.empty())
50     {
51         int u = q.front();
52         for (int i = head[u]; ~i; i = e[i].nxt)
53         {
54             int v = e[i].v;
55             if (e[i].f && dis[v] > dis[u] + e[i].c)
56             {
57                 dis[v] = dis[u] + e[i].c;
58                 pre[v] = i;
59                 if (!vis[v])
60                 {
61                     vis[v] = 1;
62                     q.push(v);
63                 }
64             }
65         }
66         q.pop();
67         vis[u] = 0;
68     }
69     return dis[t] != inf;
70 }
71 void augment()
72 {
73     int u = t;
74     T f = numeric_limits<T>::max();
75     while (~pre[u])
76     {
77         f = min(f, e[pre[u]].f);
78         u = e[pre[u] ^ 1].v;
79     }
80     flow += f;

```

```

81     cost += f * dis[t];
82     u = t;
83     while (~pre[u])
84     {
85         e[pre[u]].f -= f;
86         e[pre[u] ^ 1].f += f;
87         u = e[pre[u] ^ 1].v;
88     }
89 }
90 array<int, 2> solve()
91 {
92     flow = 0;
93     cost = 0;
94     while (spfa())
95         augment();
96     return {flow, cost};
97 }
98
99 void init(int s_, int t_, int vtot_)
100 {
101     s = s_;
102     t = t_;
103     vtot = vtot_;
104     etot = 0;
105     for (int i = 1; i <= vtot; ++i)
106         head[i] = -1;
107 }
108 };
109 MinCostGraph<int> g;
110 int n, m;
111 int main()
112 {
113     n = read();
114     m = read();
115     g.init(1, n, n);
116     for (int i = 1; i <= m; i++)
117     {
118         int u = read(), v = read(), f = read(), c = read();
119         g.addedge(u, v, f, c);
120     }
121     array<int, 2> it = g.solve();
122     printf("%d %d", it[0], it[1]);
123     return 0;
124 }

```


写法一

```
1  string s, t;
2  cin >> s >> t;
3  ll n = s.length(), m = t.length();
4  s = " " + s;
5  t = " " + t;
6  ll nxt[m + 1] = {0}, f[n + 1] = {0};
7  ll j = 0;
8  rep(i, 2, m)
9  {
10     while (j > 0 && t[j + 1] != t[i])
11         j = nxt[j];
12     if (t[j + 1] == t[i])
13         j++;
14     nxt[i] = j;
15 }
16 j = 0;
17 rep(i, 1, n)
18 {
19     while ((j == m) || (j > 0 && t[j + 1] != s[i]))
20         j = nxt[j];
21     if (t[j + 1] == s[i])
22         j++;
23     f[i] = j;
24 }
```

写法二

```
1  string s, t;
2  cin >> s >> t;
3  ll n = s.length(), m = t.length();
4  t = " " + t + "#" + s;
5  ll nxt[n + m + 2] = {0};
6  ll j = 0;
7  rep(i, 2, n + m + 2)
8  {
9     while (j > 0 && t[j + 1] != t[i])
10         j = nxt[j];
11     if (t[j + 1] == t[i])
12         j++;
13     nxt[i] = j; // 从 m + 1 到 n + m + 1 遍历寻找 nxt[i] == m
14 }
```

```

//哈希
//H(x) = x % mod;
const int mod = 11; //注意是常量,或者写成int mod = 11;
vector<int> HashTable[11]; //vector<vector<int>> HashTable(mod); {
//计算哈希值
int Hash(int x)
{
    return x % mod;
}
//插入新值
void Insert(int x)
{
    int add = Hash(x);
    HashTable[add].push_back(x);
}
//查找元素
bool isExist(int x)
{
    int add = Hash(x);
    int len = HashTable[add].size();
    for(int i = 0; i < len; i++)
    {
        if(HashTable[add][i] == x)
            return true;
    }
    return false;
}

//字符串哈希
int p = 26;
int base = 37; //大于字符数量的素数:37 101 137等
int Hash(string s)
{
    int res = 0;
    int len = s.length();
    for(int i = 0; i < len; i++)
        res = (res * base + s[i] - 'a' + 1) % p;
    return res;
}
//计算子串
int a[N + 1]; //1-n下标依次
int SonHash(string s)
{
    int res = 0;
    int len = s.length();
    for(int i = 0; i < len; i++)
        res = (res * base + s[i] - 'a' + 1) % p, a[i + 1] = res;
    return res;
}
int CalcSubstringHash(int l, int r) //子串哈希值
{
    int t = a[l - 1] * (int)pow(base, r - l + 1) % p; //快速幂
    return (a[r] - t + p) % p;
}

```

```

//整数二分
bool check(int x) { /*Code*/ }

int calc(int l, int r)
{
    while(r - l > 1)
    {
        int mid = (l + r) / 2;
        if(check(mid))
            l = mid;
        else
            r = mid;
    }
    return l;
}

```

```

//浮点数二分
bool check(double x) { /*Code*/ }

double calc(double l, double r)
{
    const double eps = 1e-6; //题目给出的精度
    while(r - l > eps)
    {
        double mid = (l + r) / 2;
        if(check(mid))
            l = mid;
        else
            r = mid;
    }
    return l;
}

```

```

//逆元
const int mod = 1e9 + 7;
int inv(int a)
{
    int ret = 1;
    int b = mod - 2;
    while(b)
    {
        if(b & 1)
            ret = ret * a % mod;
        b /= 2;
        a = a * a % mod;
    }
    return ret;
}

```

```

//阶乘逆元
const int mod = 1e9 + 7;
const int N = 1e6 + 10;
int Fac[N];
int Inv[N];
int inv(int a){...}
void init(int n)
{
    Fac[0] = 1;
    for(int i = 1; i <= n; i++)
        Fac[i] = Fac[i - 1] * i % mod;
    Inv[n] = inv(Fac[n]);
    for(int i = n; i >= 0; i--)
        Inv[i] = Inv[i + 1] * (i + 1) % mod;
}

```