















# 桂林更新 2023. 10. 26

桂林更新 2023. 10. 26

1. BIT\_with\_seg
2. cdq
3. cosf\_dinic
4. 点分治
5. dinic
6. geo
7. gsa2
8. int\_fenkuai
9. lct3
10. lct4
11. modui
12. SA2
13. scan
14. 组合

名称	修改日期
 BIT_with_seg.cpp	2023/10/19 21:13
 cdq.cpp	2023/10/19 23:54
 cosf_dinic.cpp	2023/10/4 23:27
 dian_fenzhi.cpp	2023/10/22 16:27
 dinic.cpp	2023/10/1 20:04
 geo.cpp	2023/8/26 0:19
 gsa2.cpp	2023/10/13 23:41
 int_fenkuai.cpp	2023/1/26 14:08
 lct3.cpp	2023/9/4 0:01
 lct4.cpp	2023/9/4 19:30
 modui.cpp	2023/10/26 16:53
 SA2.cpp	2023/10/26 16:57
 scan.cpp	2023/8/9 23:40
 zhu_he.cpp	2023/1/25 10:04

## 1. BIT\_with\_seg

```

1 #include<bits/stdc++.h>
2 using namespace std;
3 const int maxn=2e5+5;
```

```

4 struct segtr {
5     static const int maxp=maxn*200;
6     int ls[maxp], rs[maxp], sum[maxp], pcnt;
7     vector<int>ok;
8     void addnode(int &p) {
9         if(ok.empty()) {p++;pcnt;return ;}
10        p=ok.back();ok.pop_back();
11    }
12    void del(int &p) {ok.push_back(p);p=0;}
13    void init() {
14        ok.clear();
15        memset(ls,0,sizeof(ls));
16        memset(rs,0,sizeof(rs));
17        memset(sum,0,sizeof(sum));
18        pcnt=0;
19    }
20    void pushup(int p) {sum[p]=sum[ls[p]]+sum[rs[p]];}
21    void updata(int &u, int l, int r, int q, int k) {
22        if(!u) addnode(u);
23        sum[u]+=k;
24        if(l<r) {
25            int mid=(l+r)>>1;
26            if(q<=mid) updata(ls[u], l, mid, q, k);
27            else updata(rs[u], mid+1, r, q, k);
28        }
29        if(!sum[u]) del(u);
30    }
31    int query(int p, int l, int r, int ql, int qr) {
32        if(!p) return 0;
33        if(ql<=l&&r<=qr) return sum[p];
34        int mid=(l+r)>>1;
35        int res=0;
36        if(ql<=mid) res+=query(ls[p], l, mid, ql, qr);
37        if(mid<qr) res+=query(rs[p], mid+1, r, ql, qr);
38        return res;
39    }
40 };
41 struct BIT {
42     int n;
43     segtr seg;
44     int rt[maxn];
45     int lowbit(int p) {return p&-p;}
46     void init(int n=0) {this->n=n;seg.init();}
47     void updata(int p, int q, int k) {
48         for(;p<=n;p+=lowbit(p)) seg.updata(rt[p], 1, n, q, k);
49     }
50     int query(int p, int ql, int qr) {
51         int res=0;
52         for(;p; p-=lowbit(p)) res+=seg.query(rt[p], 1, n, ql, qr);
53         return res;
54     }
55     int query(int l, int r, int ql, int qr) {return query(r, ql, qr)-query(l-1, ql, qr);}
56 }bit;
57 void solve() {
58     int n, m;

```

```

59     cin>>n>>m;
60     vector<int>a(n+1), b(n+1), pos(n+1), c(n+1);
61     for(int i=1; i<=n; ++i) cin>>a[i], pos[a[i]]=i;
62     for(int i=1; i<=n; ++i) cin>>b[i], b[i]=pos[b[i]];
63     bit.init(n);
64     for(int i=1; i<=n; ++i) bit.updata(i, b[i], 1);
65     while(m--) {
66         int op, x, y;
67         cin>>op>>x>>y;
68         if(op==1) {
69             int l, r;
70             cin>>l>>r;
71             cout<<bit.query(l, r, x, y)<<"\n";
72         } else {
73             bit.updata(x, b[x], -1);
74             bit.updata(y, b[y], -1);
75             swap(b[x], b[y]);
76             bit.updata(x, b[x], 1);
77             bit.updata(y, b[y], 1);
78         }
79     }
80 }
81 signed main() {
82     ios::sync_with_stdio(0); cin.tie(0);
83     solve();
84 }

```

## 2. cdq

```

1  #include<bits/stdc++.h>
2  using namespace std;
3  const int maxn=2e5+5;
4  struct BIT{
5      vector<int>tr;
6      int n;
7      BIT(int n):n(n), tr(n+12) {};
8      int lowbit(int p){return p&-p;}
9      void updata(int p, int x){
10         for(; p<=n; p+=lowbit(p)) tr[p]+=x;
11     }
12     int query(int p){
13         int res=0;
14         for(; p; p-=lowbit(p)) res+=tr[p];
15         return res;
16     }
17     int query(int l, int r){return query(r)-query(l-1);}
18     void clear(){for(auto &x:tr)x=0;}
19 };
20 struct CDQ{
21     struct event{
22         int l, r, y, val, id;
23         event(int l=0, int r=0, int y=0, int val=0, int id=0):l(l), r(r), y(y), val(val), id(id) {}
24         const bool isupdata(){return id<0;}
25         const bool isquery(){return id>0;}

```

```

26     bool const operator<(const event& o) const {return y<o.y;}
27 };
28 vector<event>events,buffer;
29 vector<int>res;
30 int n;BIT bit;
31 CDQ(int n):n(n),bit(n){init();}
32 int mx,mxid;
33 void init(){mx=mxid=0;}// BIT clear()???
34 void addupdate(int x,int y,int val){
35     events.push_back({x,x,y,val,-1});
36     mx=max(mx,x);
37 }
38 void addquery(int l,int r,int y,int val,int id){
39     events.push_back({l,r,y,val,id});
40     mx=max(mx,r);
41     mxid=max(mxid,id);
42 }
43 void solve(int l,int r){
44     if(r-l+1<=100){
45         for(int i=l;i<=r;++i)if(events[i].isquery()){
46             auto now=events[i];
47             for(int j=l;j<i;++j)if(events[j].isupdate()){
48                 if(events[j].y<=now.y&&now.l<=events[j].l&&events[j].l<=now.r)
49                     res[now.id]+=now.val*events[j].val;
50             }
51         }
52         sort(events.begin()+l,events.begin()+r+1);
53         return ;
54     }
55     int mid=(l+r)>>1;
56     solve(l,mid),solve(mid+1,r);
57     int cur=l;
58     for(int i=l,j=mid+1;i<=mid||j<=r;){
59         if(j>r||i<=mid&&events[i].y<=events[j].y){
60             if(events[i].isupdate())bit.update(events[i].l,events[i].val);
61             buffer[cur++]=events[i++];
62         }else{
63             if(events[j].isquery())res[events[j].id]+=events[j].val*bit.query(events[j].l,events[j].r);
64             buffer[cur++]=events[j++];
65         }
66     }
67     if(mid-l+1>(bit.n>>9))bit.clear();
68     else for(int i=l;i<=mid;++i)if(events[i].isupdate())bit.update(events[i].l,-events[i].val);
69     for(int i=l;i<=r;++i)events[i]=buffer[i];
70 }
71 void solve(){
72     res.assign(mxid+50,0);
73     bit.n=mx;bit.tr.assign(mx+50,0);
74     buffer.resize(events.size()+50);
75     solve(0,events.size()-1);
76 }
77 };
78 void solve(){
79     int n,m;

```

```

80     cin>>n>>m;
81     CDQ cdq(n);
82     vector<int>a(n+1), b(n+1), pos(n+1);
83     for(int i=1; i<=n; ++i) cin>>a[i], pos[a[i]]=i;
84     for(int i=1; i<=n; ++i) cin>>b[i], b[i]=pos[b[i]], cdq.addupdata(i, b[i], 1);
85     int cnt=0;
86     while(m--) {
87         int op, x, y;
88         cin>>op>>x>>y;
89         if(op==1) {
90             int l, r;
91             cin>>l>>r;
92             cdq.addquery(l, r, y, 1, cnt);
93             cdq.addquery(l, r, x-1, -1, cnt);
94             ++cnt;
95         } else {
96             cdq.addupdata(x, b[x], -1);
97             cdq.addupdata(y, b[y], -1);
98             swap(b[x], b[y]);
99             cdq.addupdata(x, b[x], 1);
100            cdq.addupdata(y, b[y], 1);
101        }
102    }
103    cdq.solve();
104    for(int i=0; i<cnt; ++i) cout<<cdq.res[i]<<"\n";
105 }
106 signed main() {
107     ios::sync_with_stdio(0); cin.tie(0);
108     solve();
109 }

```

### 3. cosf\_dinic

```

1  #include<bits/stdc++.h>
2  using namespace std;
3  const int V = 20100;
4  const int E = 201000;
5  template<typename T>
6  struct MinCostGraph {
7      int s, t, vtot;
8      int head[V], etot;
9      T dis[V], flow, cost;
10     int pre[V];
11     bool vis[V];
12     struct edge {
13         int v, nxt;
14         T f, c;
15     } e[E * 2];
16     void addedge(int u, int v, T f, T c, T f2 = 0) {
17         e[etot] = {v, head[u], f, c};
18         head[u] = etot++;
19         e[etot] = {u, head[v], f2, -c};
20         head[v] = etot++;
21     }

```

```

22 bool spfa() {
23     T inf = numeric_limits<T>::max() / 2;
24     //vector<T>incf(vtot, 0); incf[s]=inf;
25     for(int i = 0; i <= vtot; ++i) {
26         dis[i] = inf;
27         vis[i] = false;
28         pre[i] = -1;
29     }
30     dis[s] = 0;
31     vis[s] = true;
32     queue<int> q;
33     q.push(s);
34     while(!q.empty()) {
35         int u = q.front();
36         for(int i = head[u]; ~i; i = e[i].nxt) {
37             int v = e[i].v;
38             if(e[i].f && dis[v] > dis[u] + e[i].c) { //反
39                 dis[v] = dis[u] + e[i].c;
40                 pre[v] = i;
41                 //incf[v]=min(incf[u], e[i].f);
42                 if(!vis[v]) {
43                     vis[v] = true;
44                     q.push(v);
45                 }
46             }
47         }
48         q.pop();
49         vis[u] = false;
50     }
51     return dis[t] != inf; //incf[t]>0;
52 }
53 void augment() {
54     int u = t;
55     T f = numeric_limits<T>::max();
56     while(~pre[u]) {
57         f = min(f, e[pre[u]].f);
58         u = e[pre[u] ^ 1].v;
59     }
60
61     flow += f;
62     cost += f * dis[t];
63     u = t;
64     while(~pre[u]) {
65         e[pre[u]].f -= f;
66         e[pre[u] ^ 1].f += f;
67         u = e[pre[u] ^ 1].v;
68     }
69 }
70 pair<T, T> solve() {
71     flow = 0;
72     cost = 0;
73     while(spfa()) augment();
74     return {flow, cost};
75 }
76 void init(int s_, int t_, int vtot_) {

```

```

77     s = s_;
78     t = t_;
79     vtot = vtot_;
80     etot = 0;
81     //如果要用0这个点的话, i要从0开始
82     for(int i = 0; i <= vtot; ++i) head[i] = -1;
83 }
84 };
85 MinCostGraph<int> g;
86 void solve() {
87     int n, m, S, T;
88     cin >> n >> m >> S >> T;
89     g.init(S, T, n + 10);
90     for(int i = 1; i <= m; ++i) {
91         int u, v, f, c;
92         cin >> u >> v >> f >> c;
93         g.addedge(u, v, f, c);
94     }
95     auto [flow, cost] = g.solve();
96     cout << flow << " " << cost;
97 }

```

## 4. 点分治

```

1  #include<bits/stdc++.h>
2  #define int long long
3  using namespace std;
4  vector<int>g[200000+50];
5  int ctr=-1, n, k, sz[200000+50], del[200000+50];
6  void dfs(int u, int f=0) {
7      sz[u]=1;
8      int mx=0;
9      for(auto v:g[u]) {
10         if(del[v] || v==f) continue;
11         dfs(v, u);
12         if(ctr!=-1) return ;
13         mx=max(mx, sz[v]);
14         sz[u]+=sz[v];
15     }
16     mx=max(mx, n-sz[u]);
17     if(mx<=n/2) {
18         ctr=u;
19         sz[f]=n-sz[u];
20     }
21 }
22 int tmp[200000+50], tot, cnt, lens[200000+50];
23 void dfs2(int u, int f, int len) {
24     if(len>k) return ;
25     cnt+=lens[k-len]+(len==k);
26     tmp[tot++]=len;
27     for(auto v:g[u]) {
28         if(del[v] || v==f) continue;
29         dfs2(v, u, len+1);
30     }

```

```

31 }
32 void run(int u) {
33     for(auto v:g[u]) {
34         if(del[v])continue;
35         dfs2(v,u,1);
36         for(int i=0;i<tot;++i) lens[tmp[i]]++;
37         tot=0;
38     }
39     for(int i=0;i<=k;++i) lens[i]=0;
40     del[u]=1;
41     for(auto v:g[u]) {
42         if(del[v])continue;
43         n=sz[v];
44         ctr=-1;
45         dfs(v);
46         run(ctr);
47     }
48 }
49 void solve() {
50     cin>>n>>k;
51     for(int i=1;i<n;++i) {
52         int u,v;
53         cin>>u>>v;
54         g[u].push_back(v);
55         g[v].push_back(u);
56     }
57     dfs(1);
58     run(ctr);
59     cout<<cnt<<"\n";
60 }
61 signed main() {
62     ios::sync_with_stdio(0);
63     cin.tie(0);
64     solve();
65 }

```

## 5. dinic

```

1 #include<bits/stdc++.h>
2 using namespace std;//sqrt(n)*m, E*V*V
3 #define int long long
4 const int V = 1e5+11;
5 const int E = V*30;
6 template<typename T>
7 struct FlowGraph {
8     int s, t, vtot;
9     int head[V], etot;
10    int dis[V], cur[V];
11    struct edge {
12        int v, nxt;
13        T f;
14    } e[E * 2];
15
16    void addedge(int u, int v, T f) {

```



```

17     e[etot] = {v, head[u], f};
18     head[u] = etot++;
19     e[etot] = {u, head[v], 0};
20     head[v] = etot++;
21 }
22
23 bool bfs() {
24     //如果要0这个点的话, i要从0开始
25     for(int i = 0; i <= vtot; i++) {
26         dis[i] = 0;
27         cur[i] = head[i];
28     }
29     queue<int> q;
30     q.push(s);
31     dis[s] = 1;
32     while(!q.empty()) {
33         int u = q.front();
34         q.pop();
35         for(int i = head[u]; ~i; i = e[i].nxt) {
36             if(e[i].f && !dis[e[i].v]) {
37                 int v = e[i].v;
38                 dis[v] = dis[u] + 1;
39                 if(v == t) return true;
40                 q.push(v);
41             }
42         }
43     }
44     return false;
45 }
46
47 T dfs(int u, T m) {
48     if(u == t) return m;
49     T flow = 0;
50     for(int i = cur[u]; ~i; cur[u] = i = e[i].nxt) {
51         if(e[i].f && dis[e[i].v] == dis[u] + 1) {
52             T f = dfs(e[i].v, min(m, e[i].f));
53             e[i].f -= f;
54             e[i ^ 1].f += f;
55             m -= f;
56             flow += f;
57             if(!m) break;
58         }
59     }
60     if(!flow) dis[u] = -1;
61     return flow;
62 }
63
64 T dinic() {
65     T flow = 0;
66     while(bfs()) flow += dfs(s, numeric_limits<T>::max());
67     return flow;
68 }
69 void init(int s_, int t_, int vtot_) {
70     s = s_, t = t_, vtot = vtot_;
71     etot = 0;

```

```

72     //如果要用0这个点的话，i要从0开始
73     for(int i = 0; i <= vtot; i++) head[i] = -1;
74 }
75 };
76 FlowGraph<int> g;
77 void solve() {
78
79 }
80 signed main() {
81     ios::sync_with_stdio(0);cin.tie(0);
82     solve();
83 }

```

## 6. geo

```

1  #include <bits/stdc++.h>
2  using namespace std;
3
4  struct Point { double x, y; };          // 点
5  using Vec = Point;                     // 向量
6  struct Line { Point P; Vec v; };        // 直线（点向式）
7  struct Seg { Point A, B; };             // 线段（存两个端点）
8  struct Circle { Point O; double r; };   // 圆（存圆心和半径）
9
10 const Point O = {0, 0};                 // 原点
11 const Line Ox = {0, {1, 0}}, Oy = {0, {0, 1}}; // 坐标轴
12 const double PI = acos(-1), EPS = 1e-9;
13
14 bool eq(double a, double b) { return abs(a - b) < EPS; } // ==
15 bool gt(double a, double b) { return a - b > EPS; }      // >
16 bool lt(double a, double b) { return a - b < -EPS; }     // <
17 bool ge(double a, double b) { return a - b > -EPS; }     // >=
18 bool le(double a, double b) { return a - b < EPS; }      // <=
19
20 Vec r90a(Vec v) { return {-v.y, v.x}; }                  // 逆时针旋转90度的向量
21 Vec r90c(Vec v) { return {v.y, -v.x}; }                  // 顺时针旋转90度的向量
22 Vec operator+(Vec u, Vec v) { return {u.x + v.x, u.y + v.y}; } // 向量加向量
23 Vec operator-(Vec u, Vec v) { return {u.x - v.x, u.y - v.y}; } // 向量减向量
24 Vec operator*(double k, Vec v) { return {k * v.x, k * v.y}; } // 数乘
25 double operator*(Vec u, Vec v) { return u.x * v.x + u.y * v.y; } // 点乘
26 double operator^(Vec u, Vec v) { return u.x * v.y - u.y * v.x; } // 叉乘
27 double len(Vec v) { return sqrt(v.x * v.x + v.y * v.y); } // 向量长度
28 double slope(Vec v) { return v.y / v.x; }                // 斜率 // NOTE 不要用isinf判断斜率
    不存在，用后面的paral_y
29
30 // 两向量的夹角余弦
31 double cos_t(Vec u, Vec v) { return u * v / len(u) / len(v); } // DEPENDS len, V*V
32
33 // 归一化向量（与原向量方向相同的单位向量）
34 Vec norm(Vec v) { return {v.x / len(v), v.y / len(v)}; } // DEPENDS len
35
36 // 与原向量平行且横坐标大于等于0的单位向量
37 Vec pnorm(Vec v) { return (v.x < 0 ? -1 : 1) / len(v) * v; } // DEPENDS d*V, len
38

```

```

39 // 线段的方向向量
40 // NOTE 直线的方向向量直接访问属性v
41 Vec dvec(Seg l) { return l.B - l.A; } // DEPENDS V-V
42
43 // 两点式直线
44 Line line(Point A, Point B) { return {A, B - A}; }
45
46 // 斜截式直线
47 Line line(double k, double b) { return {{0, b}, {1, k}}; }
48
49 // 点斜式直线
50 Line line(Point P, double k) { return {P, {1, k}}; }
51
52 // 线段所在直线
53 Line line(Seg l) { return {l.A, l.B - l.A}; } // DEPENDS V-V
54
55 // 给定直线的横坐标求纵坐标
56 // NOTE 请确保直线不与y轴平行
57 double at_x(Line l, double x) { return l.P.y + (x - l.P.x) * l.v.y / l.v.x; }
58
59 // 给定直线的纵坐标求横坐标
60 // NOTE 请确保直线不与x轴平行
61 double at_y(Line l, double y) { return l.P.x - (y - l.P.y) * l.v.x / l.v.y; }
62
63 // 点到直线的垂足
64 // DEPENDS V-V, V*V, d*V
65 Point pedal(Point P, Line l) { return l.P - (l.P - P) * l.v / (l.v * l.v) * l.v; }
66
67 // 过某点作直线的垂线
68 Line perp(Line l, Point P) { return {P, r90c(l.v)}; } // DEPENDS r90c
69
70 // 角平分线
71 Line bisec(Point P, Vec u, Vec v) { return {P, norm(u) + norm(v)}; } // DEPENDS V+V, len, norm
72
73 // 线段的方向向量
74 // NOTE 直线的方向向量直接访问属性v
75 Vec dvec(Seg l) { return l.B - l.A; } // DEPENDS V-V
76
77 // 线段中点
78 Point midp(Seg l) { return {(l.A.x + l.B.x) / 2, (l.A.y + l.B.y) / 2}; }
79
80 // 线段中垂线
81 Line perp(Seg l) { return {midp(l), r90c(l.B - l.A)}; } // DEPENDS r90c, V-V, midp
82
83
84 // 向量是否互相垂直
85 bool verti(Vec u, Vec v) { return eq(u * v, 0); } // DEPENDS eq, V*V
86
87 // 向量是否互相平行
88 bool paral(Vec u, Vec v) { return eq(u ^ v, 0); } // DEPENDS eq, V^V
89
90 // 向量是否与x轴平行
91 bool paral_x(Vec v) { return eq(v.y, 0); } // DEPENDS eq V-V
92
93 // 向量是否与y轴平行

```

```

94 bool paral_y(Vec v) { return eq(v.x, 0); } // DEPENDS eq
95
96 // 点是否在直线上
97 bool on(Point P, Line l) { return eq((P.x - l.P.x) * l.v.y, (P.y - l.P.y) * l.v.x); } // DEPENDS eq
98
99 // 点是否在线段上
100 bool on(Point P, Seg l) { return eq(len(P - l.A) + len(P - l.B), len(l.A - l.B)); } // DEPENDS eq,
    len, V-V
101
102 // 两个点是否重合
103 bool operator==(Point A, Point B) { return eq(A.x, B.x) && eq(A.y, B.y); } // DEPENDS eq
104
105 // 两条直线是否重合
106 bool operator==(Line a, Line b) { return on(a.P, b) && on(a.P + a.v, b); } // DEPENDS eq, on(L)
107
108 // 两条线段是否重合
109 bool operator==(Seg a, Seg b) { return (a.A == b.A && a.B == b.B) || (a.A == b.B && a.B == b.A); } //
    DEPENDS eq, P==P
110
111 // 以横坐标为第一关键词、纵坐标为第二关键词比较两个点
112 bool operator<(Point A, Point B) { return lt(A.x, B.x) || (eq(A.x, B.x) && lt(A.y, B.y)); } // DEPENDS
    eq, lt
113
114 // 直线与圆是否相切
115 bool tangency(Line l, Circle C) { return eq(abs((C.O ^ l.v) - (l.P ^ l.v)), C.r * len(l.v)); } //
    DEPENDS eq, V^V, len
116
117 // 圆与圆是否相切
118 bool tangency(Circle C1, Circle C2) { return eq(len(C1.O - C2.O), C1.r + C2.r); } // DEPENDS eq, V-V,
    len
119
120 // 两点间的距离
121 double dis(Point A, Point B) { return len(A - B); } // DEPENDS len, V-V
122
123 // 点到直线的距离
124 double dis(Point P, Line l) { return abs((P ^ l.v) - (l.P ^ l.v)) / len(l.v); } // DEPENDS V^V, len
125
126 // 平行直线间的距离
127 // NOTE 请确保两直线是平行的
128 double dis(Line a, Line b) { return abs((a.P ^ pnorm(a.v)) - (b.P ^ pnorm(b.v))); } // DEPENDS d*V,
    V^V, len, pnorm
129
130
131 // 平移
132 Line operator+(Line l, Vec v) { return {l.P + v, l.v}; } // DEPENDS V+V
133 Seg operator+(Seg l, Vec v) { return {l.A + v, l.B + v}; }
134
135 // 旋转
136 Point rotate(Point P, double rad) { return {cos(rad) * P.x - sin(rad) * P.y, sin(rad) * P.x +
    cos(rad) * P.y}; } // DEPENDS V+V, V-V
137 Point rotate(Point P, double rad, Point C) { return C + rotate(P - C, rad); } //
    DEPENDS ^1
138 Line rotate(Line l, double rad, Point C = 0) { return {rotate(l.P, rad, C), rotate(l.v, rad)}; } //
    DEPENDS ^1, ^2

```

```

139 Seg rotate(Seg l, double rad, Point C = 0) { return {rotate(l.A, rad, C), rotate(l.B, rad, C)}; } //
DEPENDS ^1, ^2
140
141 // 对称
142 // 关于点对称
143 Point reflect(Point A, Point P) { return {P.x * 2 - A.x, P.y * 2 - A.y}; }
144 Line reflect(Line l, Point P) { return {reflect(l.P, P), l.v}; } // DEPENDS ^1
145 Seg reflect(Seg l, Point P) { return {reflect(l.A, P), reflect(l.B, P)}; } // DEPENDS ^1
146 // 关于直线对称
147
148 // NOTE 向量和点在这里的表现不同, 求向量关于某直线的对称向量需要用reflect_v
149 Point reflect(Point A, Line ax) { return reflect(A, pedal(A, ax)); } // DEPENDS ^1
DEPENDS V-V, V*V, d*V, pedal
150 Vec reflect_v(Vec v, Line ax) { return reflect(v, ax) - reflect(0, ax); } // DEPENDS ^1, ^4
151 Line reflect(Line l, Line ax) { return {reflect(l.P, ax), reflect_v(l.v, ax)}; } // DEPENDS ^1, ^4,
^5
152 Seg reflect(Seg l, Line ax) { return {reflect(l.A, ax), reflect(l.B, ax)}; } // DEPENDS ^1, ^4
153
154 // 直线与直线交点
155 vector<Point> inter(Line a, Line b) { // DEPENDS eq, d*V, V*V, V+V, V^V
156     double c = a.v ^ b.v;
157     if (eq(c, 0)) return {};
158     Vec v = 1 / c * Vec{a.P ^ (a.P + a.v), b.P ^ (b.P + b.v)};
159     return {{v * Vec{-b.v.x, a.v.x}, v * Vec{-b.v.y, a.v.y}}};
160 }
161
162 // 直线与圆交点
163 vector<Point> inter(Line l, Circle C) { // DEPENDS eq, gt, V+V, V-V, V*V, d*V, len, pedal
164     Point P = pedal(C.O, l);
165     double h = len(P - C.O);
166     if (gt(h, C.r)) return {};
167     if (eq(h, C.r)) return {P};
168     double d = sqrt(C.r * C.r - h * h);
169     Vec vec = d / len(l.v) * l.v;
170     return {P + vec, P - vec};
171 }
172
173 // 圆与圆的交点
174 vector<Point> inter(Circle C1, Circle C2) { // DEPENDS eq, gt, V+V, V-V, d*V, len, r90c
175     Vec v1 = C2.O - C1.O, v2 = r90c(v1);
176     double d = len(v1);
177     if (gt(d, C1.r + C2.r) || gt(abs(C1.r - C2.r), d)) return {};
178     if (eq(d, C1.r + C2.r) || eq(d, abs(C1.r - C2.r))) return {C1.O + C1.r / d * v1};
179     double a = ((C1.r * C1.r - C2.r * C2.r) / d + d) / 2;
180     double h = sqrt(C1.r * C1.r - a * a);
181     Vec av = a / len(v1) * v1, hv = h / len(v2) * v2;
182     return {C1.O + av + hv, C1.O + av - hv};
183 }
184
185 // 三角形的重心
186 Point barycenter(Point A, Point B, Point C) { return {(A.x + B.x + C.x) / 3, (A.y + B.y + C.y) / 3}; }
187
188 // 三角形的外心
189 // NOTE 给定圆上三点求圆, 要先判断是否三点共线
190 Point circumcenter(Point A, Point B, Point C) { // DEPENDS r90c, V*V, d*V, V-V, V+V

```

```

191     double a = A * A, b = B * B, c = C * C;
192     double d = 2 * (A.x * (B.y - C.y) + B.x * (C.y - A.y) + C.x * (A.y - B.y));
193     return 1 / d * r90c(a * (B - C) + b * (C - A) + c * (A - B));
194 }
195
196 // 三角形的内心
197 Point incenter(Point A, Point B, Point C) { // DEPENDS len, d*V, V-V, V+V
198     double a = len(B - C), b = len(A - C), c = len(A - B);
199     double d = a + b + c;
200     return 1 / d * (a * A + b * B + c * C);
201 }
202
203 // 三角形的垂心
204 Point orthocenter(Point A, Point B, Point C) { // DEPENDS V*V, d*V, V-V, V^V, r90c
205     double n = B * (A - C), m = A * (B - C);
206     double d = (B - C) ^ (A - C);
207     return 1 / d * r90c(n * (C - B) - m * (C - A));
208 }
209
210 double cross(Point a, Point b) { return a ^ b; }
211 int Quadrant(Point& a) {
212     if(a.x > 0 && a.y >= 0) return 1;
213     if(a.x <= 0 && a.y > 0) return 2;
214     if(a.x < 0 && a.y <= 0) return 3;
215     if(a.x >= 0 && a.y < 0) return 4;
216 }
217 void psort(vector<Point>&ps, Point c = 0) { // 极角排序
218     sort(ps.begin(), ps.end(), [&](auto a, auto b) {
219         if(Quadrant(a) != Quadrant(b)) return Quadrant(a) < Quadrant(b);
220         return cross(a, b) > 0;
221     });
222 }
223
224 int sgn(double x) {
225     if (fabs(x) < EPS) return 0;
226     return (x < 0) ? -1 : 1;
227 }
228 vector<Point>andrew(int top, vector<Point>&ps) { // 凸包
229     sort(ps.begin(), ps.end(), [&](const Point& a, const Point& b) {
230         return a.x < b.x || (sgn(a.x - b.x) == 0 && a.y < b.y);
231     });
232     auto side = [&](Point a, Point b, Point p) {
233         auto ab = b - a, ap = p - a;
234         return cross(ab, ap);
235     };
236     int n = ps.size();
237     if(n < 3) return {};
238     vector<Point>stk;
239     stk.push_back(ps[0]); stk.push_back(ps[1]);
240     top = 1;
241     for(int i = 2; i < n; ++i) {
242         while(top && side(stk[top - 1], stk[top], ps[i]) < 0) --top, stk.pop_back();
243         ++top;
244         stk.push_back(ps[i]);
245     }

```

```

246     ++top;
247     stk.push_back(ps[n-2]);
248     for(int i=n-3; i>=0; --i) {
249         while(top&&side(stk[top-1], stk[top], ps[i])<0) --top, stk.pop_back();
250         ++top;
251         stk.push_back(ps[i]);
252     }
253     return stk;
254 }
255 void solve() {
256
257 }
258 signed main() {
259     std::ios::sync_with_stdio(false); cin.tie(0);
260     solve();
261 }

```

## 7. gsa2

```

1  #include<bits/stdc++.h>
2  using namespace std;
3  #define t node
4  const int N = 4000005;
5  struct Node {
6      int ch[26]; //现在不要管
7      int len; //最大长度
8      int fa; //父节点
9  } node [ N + 5 ];
10 int last=1, tot=1;
11 void extend(int c) {
12     if (node[last].ch[c]) {
13         int p = last, q = node[p].ch[c];
14         if (node[q].len == node[p].len + 1) last = q;
15         else {
16             int nq = last = ++tot;
17             node[nq] = node[q], node[nq].len = node[p].len + 1;
18             node[q].fa = nq;
19             for (; p && node[p].ch[c] == q; p = node[p].fa) node[p].ch[c] = nq;
20         }
21     }
22     else {
23         int p = last, np = last = ++tot;
24         node[np].len = node[p].len + 1;
25         for (; p && !node[p].ch[c]; p = node[p].fa) node[p].ch[c] = np;
26         if (!p) node[np].fa = 1;
27         else {
28             int q = node[p].ch[c];
29             if (node[q].len == node[p].len + 1) node[np].fa = q;
30             else {
31                 int nq = ++tot;
32                 node[nq] = node[q], node[nq].len = node[p].len + 1;
33                 node[np].fa = node[q].fa = nq;
34                 for (; p && node[p].ch[c] == q; p = node[p].fa) node[p].ch[c] = nq;
35             }

```

```

36     }
37 }
38 }
39 //=====板子=====
40 vector<int>g[N];
41 string s[N];
42 int a[N];
43 int fa[N];
44 int find(int f) {
45     if(f==fa[f]) return f;
46     return fa[f]=find(fa[f]);
47 }
48 void slove() {
49     int n;
50     cin>>n;
51     for(int i=1; i<=n; ++i) {
52         fa[i]=i;
53         cin>>s[i];
54         last=1;
55         for(int j=0; j<s[i].length(); ++j) extend(s[i][j]-'a');
56     }
57     for(int i=1; i<=n; ++i) {
58         for(int j=0, p=1; j<s[i].size(); ++j) {
59             p=t[p].ch[s[i][j]-'a'];
60             g[p].push_back(i);
61         }
62     }
63     for(int i=2; i<=tot; ++i) a[i]=i;
64
65     sort(a+2, a+tot+1, [&](int x, int y) {
66         return t[x].len>t[y].len;
67     });
68     long long ans=0;
69     for(int i=2; i<=tot; ++i) {
70         for(int j=0; j<(int)g[a[i]].size()-1; ++j) {
71             int x=g[a[i]][j], y=g[a[i]][j+1];
72             int px=find(x), py=find(y);
73             if(px!=py) {
74                 fa[px]=py;
75                 ans+=t[a[i]].len;
76             }
77         }
78         if(t[a[i]].fa) g[t[a[i]].fa].push_back(g[a[i]][0]);
79     }
80     cout<<ans<<"\n";
81 }
82 signed main() {
83     ios::sync_with_stdio(false); cin.tie(nullptr);
84     slove();
85 }

```

## 8. int\_fenkua i

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



```

2 using namespace std;
3 #define int long long
4 int block(int st, int en, int num) { //return for (int i=st; i<=en; ++i) ans+=num/i;
5     int L=0;
6     int _res=0;
7     en=min(en, num);
8     for (int i=st; i<=en; i=L+1) {
9         L=min(en, num/(num/i));
10        _res+=(L-i+1)*(num/i);
11    }
12    return _res;
13 }
14 int up(int l, int r) {
15     return (l-1)/r+1;
16 }
17 int work(int st, int en) {
18     int ans=max(0*1LL, en/2-st+1);
19     for (int start=1, right; start<st; start=right+1) {
20         int l=start, r=st;
21         int f=up(st, start);
22         for (int i=1; i<=30; ++i) {
23             int mid=(l+r)>>1;
24             if (up(st, mid)==f) l=mid+1;
25             else r=mid;
26         }
27         right=l-1;
28         ans+=max(0LL, min(right, en/(f+1))-start+1);
29     }
30     return ans;
31 }
32 void solve() {
33     int l, r;
34     cin>>l>>r;
35     cout<<work(l, r)<<"\n";
36 }
37 signed main() {
38     std::ios::sync_with_stdio(false);
39     int t;
40     cin>>t;
41     while (t--)
42         solve();
43 }

```

## 9. lct3

```

1 #include<bits/stdc++.h>
2 using namespace std;
3 const int N = 400050;
4 struct LCT {
5     int ch[N][2], fa[N], tag[N], val[N], siz[N];
6     void clear(int x) { ch[x][0] = ch[x][1] = fa[x] = tag[x] = val[x] = siz[x] = 0; }
7     int getch(int x) { return ch[fa[x]][1] == x; }
8     int isroot(int x) { return ch[fa[x]][0] != x && ch[fa[x]][1] != x; }
9     void pushup(int x) { siz[x] = val[x] ^ siz[ch[x][0]] ^ siz[ch[x][1]]; }

```

```

10 void pushdown(int x) {
11     if (tag[x]) {
12         if (ch[x][0]) swap(ch[ch[x][0]][0], ch[ch[x][0]][1]), tag[ch[x][0]] ^= 1;
13         if (ch[x][1]) swap(ch[ch[x][1]][0], ch[ch[x][1]][1]), tag[ch[x][1]] ^= 1;
14         tag[x] = 0;
15     }
16 }
17 void update(int x) {
18     if (!isroot(x)) update(fa[x]);
19     pushdown(x);
20 }
21 void rotate(int x) {
22     int y = fa[x], z = fa[y], chx = getch(x), chy = getch(y);
23     fa[x] = z;
24     if (!isroot(y)) ch[z][chy] = x;
25     ch[y][chx] = ch[x][chx ^ 1];
26     fa[ch[x][chx ^ 1]] = y;
27     ch[x][chx ^ 1] = y;
28     fa[y] = x;
29     pushup(y), pushup(x);
30 }
31 void splay(int x) {
32     update(x);
33     for (int f = fa[x]; f = fa[x], !isroot(x); rotate(x))
34         if (!isroot(f)) rotate(getch(x) == getch(f) ? f : x);
35 }
36 void access(int x) {
37     int y = 0;
38     while (x) {
39         splay(x);
40         ch[x][1] = y;
41         pushup(x);
42         y = x;
43         x = fa[x];
44     }
45 }
46 void makeroot(int x) {
47     access(x);
48     splay(x);
49     swap(ch[x][0], ch[x][1]);
50     tag[x] ^= 1;
51 }
52 int find(int x) {
53     access(x);
54     splay(x);
55     while (ch[x][0]) x = ch[x][0];
56     splay(x);
57     return x;
58 }
59 int findfa(int x) {
60     access(x), splay(x);
61     pushdown(x), x = ch[x][0];
62     while (pushdown(x), ch[x][1]) x = ch[x][1];
63     return x;
64 }

```

```

65     void link(int x, int y) {
66         //if (find(x) != find(y)) makeroot(x), fa[x] = y;
67         access(x), splay(x), fa[x]=y;
68     }
69     void cut(int x, int y) {
70         access(x), splay(x);
71         ch[x][0]=fa[ch[x][0]]=0;
72         //if (ch[y][0] == x && !ch[x][1])ch[y][0] = fa[x] = 0;
73         if (ch[y][0] == x && fa[x]==y)ch[y][0] = fa[x] = 0, pushup(y);
74     }
75 } tree;
76 set<pair<int, int>>ps[400000+50];
77 int idx=0, bl[400000+50];
78 int get(int id, int x) {
79     auto& s=ps[id];
80     auto it =s. lower_bound({x, 0});
81     if(it->first==x)return it->second;
82     int old=it->second, oldv=it->first;
83     s.erase(it);
84     bl[++idx]=id;
85     int fa=tree.findfa(old);
86     if(fa)tree.cut(old, fa);
87     tree.link(old, idx);
88     if(fa)tree.link(idx, fa);
89     s.insert({x, old});
90     s.insert({oldv, idx});
91     return old;
92 }
93 void solve() {
94     int n, m, q;
95     cin>>n>>m>>q;
96     for(int i=1; i<=n; ++i)ps[i].insert({m+1, ++idx}), bl[idx]=i;
97     while(q--) {
98         int op;
99         cin>>op;
100         if(op==1) {
101             int a, b;
102             cin>>a>>b;
103             int v1=get(a, b), v2=get(a+1, b);
104             int f1=tree.findfa(v1), f2=tree.findfa(v2);
105             if(f1)tree.cut(v1, f1);
106             if(f2)tree.cut(v2, f2);
107             if(f1)tree.link(v2, f1);
108             if(f2)tree.link(v1, f2);
109         }
110         else{
111             int a;
112             cin>>a;
113             cout<<bl[tree.find(ps[a].begin()->second)]<<"\n";
114         }
115     }
116 }
117 signed main() {
118     ios::sync_with_stdio(0);cin.tie(0);
119     solve();

```

## 10. lct4

```

1  #include<bits/stdc++.h>
2  using namespace std;
3  #define int long long
4  using namespace std;
5  int ch[500050][2], fa[500050], tag[500050], siz[500050], val[500050], siz2[500000+50];
6  void clear(int x){ch[x][0]=ch[x][1]=fa[x]=tag[x]=siz2[x]=siz[x]=0;}
7  int getch(int x){return x==ch[fa[x]][1];}
8  int isroot(int x){return ch[fa[x]][0]!=x&&ch[fa[x]][1]!=x;}
9  void pushup(int x){clear(0);if(x)siz[x]=siz[ch[x][0]]+siz[ch[x][1]]+1+siz2[x];}
10 void pushdown(int x){
11     if(tag[x]){
12         if(ch[x][0])swap(ch[ch[x][0]][0],ch[ch[x][0]][1]),tag[ch[x][0]]^=1;
13         if(ch[x][1])swap(ch[ch[x][1]][0],ch[ch[x][1]][1]),tag[ch[x][1]]^=1;
14         tag[x]=0;
15     }
16 }
17
18 //黑盒 旋转到当前块的根
19 //也就是说，我们对于同一块内的splay进行选择，他之后还是中序遍历的结果还是对应着原树中的一条实链
20 void updata(int x){
21     if(!isroot(x))updata(fa[x]);
22     pushdown(x);
23 }
24 void rorate(int x){
25     int y=fa[x],z=fa[y],chx=getch(x),chy=getch(y);
26     fa[x]=z;
27     if(!isroot(y))ch[z][chy]=x;
28     ch[y][chx]=ch[x][chx^1];
29     fa[ch[x][chx^1]]=y;
30     ch[x][chx^1]=y;
31     fa[y]=x;
32     pushup(y),pushup(x);
33 }
34 void splay(int x){
35     updata(x);
36     for(int f=fa[x];f=fa[x],!isroot(x);rorate(x)){
37         if(!isroot(f))rorate(getch(x)==getch(f)?f:x);
38     }
39 }
40
41 //access 在原树中把从根到x的所有点放在一条实链里，使根到x成为一条实路径，并且在同一棵 Splay 里。并且
42 //下面没有实边。
43 void access(int x){
44     int y=0;
45     while(x){
46         splay(x);
47         siz2[x]+=siz[ch[x][1]]-siz[y];
48         ch[x][1]=y;
49         pushup(x);
50         y=x;
51     }
52 }

```

```

50     x=fa[x];
51 }
52 }
53 // makeroot(x) 在原树中, 把x当成根
54 void makeroot(int x) {
55     access(x); // 此操作后 x的splay块的 根 变成了 原树的根
56     splay(x);
57     swap(ch[x][0], ch[x][1]);
58     tag[x]^=1;
59 }
60 int find(int x) {
61     access(x);
62     splay(x);
63     while (ch[x][0]) x=ch[x][0];
64     splay(x);
65     return x;
66 }
67 void link(int x, int y) {
68     if (find(x) != find(y)) makeroot(x), fa[x]=y;
69 }
70 void split(int x, int y) {
71     makeroot(x);
72     access(y);
73     splay(y);
74     // 在辅助树上, y的子树就包含了 路径上的信息
75 }
76 void cut(int x, int y) {
77     split(x, y);
78     if (ch[y][0] == x && !ch[x][1]) ch[y][0] = fa[x] = 0;
79 }
80 int que(int x, int y) {
81     split(x, y);
82     return siz[y];
83 }
84 void sets(int x, int y) {
85     splay(x);
86     val[x]=y;
87     pushup(x);
88 }
89 void solve() {
90     map<int, vector<pair<int, int>>>>mp;
91     int n;
92     cin>>n;
93     for (int i=1; i<=n; ++i) siz[i]=1;
94     for (int i=1; i<=n-1; ++i) {
95         int u, v, c;
96         cin>>u>>v>>c;
97         mp[c].push_back({u, v});
98         makeroot(u);
99         makeroot(v);
100         fa[u]=v;
101         siz2[v]+=siz[u];
102     }
103     int ans=0;
104     for (auto j:mp) {

```

```

105     for(auto pii:j.second) {
106         int x=pii.first,y=pii.second;
107         split(x,y);
108         ch[y][0]=fa[x]=0;
109         pushup(x);
110         makeroot(x);
111         makeroot(y);
112     }
113     for(auto pii:j.second) {
114         int x=pii.first,y=pii.second;
115         makeroot(x),makeroot(y);
116         ans+=(siz[x])*(siz[y]);
117     }
118     for(auto pii:j.second) {
119         int x=pii.first,y=pii.second;
120         makeroot(x);
121         makeroot(y);
122         fa[x]=y;
123         siz2[y]+=siz[x];
124     }
125 }
126 cout<<ans<<"\n";
127 }
128 signed main() {
129     std::ios::sync_with_stdio(false);cin.tie(0);
130     solve();
131 }

```

## 11. modu i

```

1  #include<bits/stdc++.h>
2  using namespace std;
3  #define int long long
4  int sq,n;
5  struct node{
6      int l,r,id;
7      bool operator<(const node &o) const
8      {
9          if(l/sq!=o.l/sq) return l<o.l;
10         if(l/sq&1) return r<o.r;
11         return r>o.r;
12     }
13 }qt[205000];
14 int a[205000];
15 //int mu[200000+50];
16 map<int,int>mu;
17 int res=0;
18 int c3(int x) {
19     return (x*(x-1)*(x-2))/6;
20 }
21 void add(int p) {
22     res-=c3(mu[p]);
23     mu[p]++;
24     res+=c3(mu[p]);

```

```

25 }
26 void del(int p) {
27     res-=c3(mu[p]);
28     mu[p]--;
29     res+=c3(mu[p]);
30
31 }
32 int q;
33 int ans[205000];
34 void solve() {
35     int l=1,r=0;
36     for(int i=1;i<=q;++i) {
37         while (l > qt[i].l) add(a[--l]);
38         while (l < qt[i].l) del(a[l++]);
39         while (r < qt[i].r) add(a[++r]);
40         while (r > qt[i].r) del(a[r--]);
41         ans[qt[i].id]=res;
42     }
43 }
44 signed main() {
45     std::ios::sync_with_stdio(false);
46     cin.tie(0);cout.tie(0);
47     cin>>n;cin>>q;
48     sq=sqrt(n);
49     for(int i=1;i<=n;++i)cin>>a[i];
50     for(int i=1;i<=q;++i) {
51         cin>>qt[i].l>>qt[i].r;
52         qt[i].id=i;
53     }
54     sort(qt+1,qt+q+1);
55     solve();
56     for(int i=1;i<=q;++i) {
57         cout<<ans[i]<<"\n";
58     }
59 }

```

## 12. SA2

```

1 string s, t;
2 int n, k;
3 ll ans = 0;
4 ll pres[N], pret[N];
5 int ls[N], rs[N], stk[N];
6 int lens;
7 int que(ll* pre, int L, int R) {
8     if (L > R) return 0;
9     if (L == 0) return pre[R];
10    return pre[R] - pre[L - 1];
11 }
12 void dfs(int u, int L, int R) {
13     ll Lnum = que(pres, L - 1, u - 1);
14     ll Rnum = que(pret, u, R);
15     ans += Lnum * Rnum * max(0, height[u] - k);
16     Lnum = que(pret, L - 1, u - 1);

```

```

17     Rnum = que(pres, u, R);
18     ans += Lnum * Rnum * max(0, height[u] - k);
19     if (ls[u]) dfs(ls[u], L, u - 1);
20     if (rs[u]) dfs(rs[u], u + 1, R);
21 }
22 void slove() {
23     while (cin >> k) {
24         if (k == 0) break;
25         cin >> s >> t;
26         k--;
27         SA(s + "$" + t);
28         n = s.length() + t.length() + 1;
29         lens = s.length();
30         for (int i = 1; i <= n; i++) {
31             if (sa[i] <= lens)
32                 pres[i] = pres[i - 1] + 1, pret[i] = pret[i - 1];
33             else
34                 pres[i] = pres[i - 1], pret[i] = pret[i - 1] + 1;
35         }
36         int top = 0;
37         for (int i = 1; i <= n; i++) stk[i] = 0, ls[i] = 0, rs[i] = 0;
38         for (int i = 1; i <= n; i++) {
39             int k = top;
40             while (k && height[i] <= height[stk[k]]) k--;
41             if (k) rs[stk[k]] = i;
42             if (k < top) ls[i] = stk[k + 1];
43             stk[++k] = i; top = k;
44         }
45         ans = 0;
46         dfs(stk[1], 1, n);
47         cout << ans << endl;
48     }
49 }
50

```

## 13. scan

```

1  #include<bits/stdc++.h>
2  #define int long long
3  using namespace std;
4  #define ls(p) p<<1
5  #define rs(p) p<<1|1
6  int cover[200000*4+50];
7  double length[200000+50], yy[200000*4+50];
8  struct scan{
9      double x, upy, downy;
10     int in_out;
11     scan() {} ;
12     scan(double x, double y1, double y2, int in):x(x), upy(y1), downy(y2), in_out(in) {} ;
13     const bool operator<(const scan& o) const {return x<o.x;}
14 } line[200000+50];
15 void pushup(int p, int l, int r) {
16     if(cover[p]) length[p]=yy[r]-yy[l];
17     else if(l+1==r) length[p]=0;

```



```

18     else length[p]=length[ls(p)]+length[rs(p)];
19 }
20 void updata(int p,int l,int r,int yl,int yr,int io){
21     if(yl>r||yr<l)return;
22     if(yl<=l&&r<=yr){
23         cover[p]+=io;
24         pushup(p,l,r);
25         return;
26     }
27     if(l+1==r)return;
28     int mid=(l+r)>>1;
29     updata(ls(p),l,mid,yl,yr,io),updata(rs(p),mid,r,yl,yr,io);
30     pushup(p,l,r);
31 }
32 void solve() {
33     int n;
34     cin>>n;
35     int cnt=0;
36     double x1,x2,y1,y2;
37     int yl,yr,io;
38     for(int i=1;i<=n;++i){
39         cin>>x1>>y1>>x2>>y2;
40         line[++cnt]=scan(x1,y2,y1,1);
41         yy[cnt]=y1;
42         line[++cnt]=scan(x2,y2,y1,-1);
43         yy[cnt]=y2;
44     }
45     sort(yy+1,yy+1+cnt);
46     sort(line+1,line+1+cnt);
47     int len=unique(yy+1,yy+1+cnt)-yy-1;
48     memset(cover,0,sizeof cover);memset(length,0,sizeof length);
49     double ans=0;
50     for(int i=1;i<=cnt;++i){
51         ans+=length[1]*(line[i].x-line[i-1].x);
52         yl=lower_bound(yy+1,yy+1+len,line[i].downy)-yy;
53         yr=lower_bound(yy+1,yy+1+len,line[i].upy)-yy;
54         io=line[i].in_out;
55         updata(1,1,len,yl,yr,io);
56     }
57     cout<<ans<<"\n";
58 }
59 signed main() {
60     ios::sync_with_stdio(false);cin.tie(nullptr);
61     solve();
62 }

```

## 14. 组合

```

1 #include<iostream>
2
3 using namespace std;
4
5 const int N = 2020;
6 int mod=998244353;

```

```

7
8 int c[N][N]; //相当于数学中的组合公式Cab (数学公式不知如何插入le...)
9
10 void init() {
11     for(int i = 0; i < N; i++)
12         for(int j = 0; j <= i; j++)
13             if(!j) c[i][j] = 1;
14             else c[i][j] = (c[i - 1][j] + c[i - 1][j - 1]) % mod; //递推公式
15
16 }
17 /*
18 int main() {
19     init();
20     int n; cin >> n;
21     while(n --) {
22         int a, b;
23         cin >> a >> b;
24         cout << c[a][b] << endl;
25     }
26     return 0;
27 }*/
28 #include<iostream>
29 using namespace std;
30 #define int long long
31 const int N = 1e6 + 7, mod = 1e9 + 7;
32 int n, m, k, t;
33 int inv[N];
34 int fact[N], infact[N];
35
36 void init (int n)
37 {
38     fact[0] = infact[0] = inv[0] = inv[1] = 1;
39     for (int i = 2; i <= n; ++ i)
40         inv[i] = 1ll * (mod - mod / i) * inv[mod % i] % mod;
41     for (int i = 1; i <= n; ++ i) {
42         fact[i] = 1ll * fact[i - 1] * i % mod;
43         infact[i] = 1ll * infact[i - 1] * inv[i] % mod;
44     }
45 }
46
47 int c(int n, int m)
48 {
49     if(n < m) return 0;
50     if(m == 0 || n == m) return 1;
51     return 1ll * fact[n] * infact[m] % mod * infact[n - m] % mod;
52 }

```