

Standard Code Library

111

zu

August 7, 2022

Contents

一切的开始	2
宏定义	2
数据结构	2
ST 表	2
树状数组	2
线段树	3
主席树	4
FFT	5
图论	6
LCA	6
最短路	6
最大流	7
费用流 (dij)	8
费用流 (spfa)	9
差分约束	10
Kruskal 重构树	10
计算几何	11
二维几何: 点与向量	11
点	12
圆	13
字符串	14
AC 自动机	14
KMP	15
SA (dc3)	15
SAM	18
PAM	19
杂项	20
STL	20
int_128	20
背包	21
高精度	21

一切的开始

宏定义

- 需要 C++11

```
1  #include <bits/stdc++.h>
2  using namespace std;
3  using LL = long long;
4  #define FOR(i, x, y) for (decay<decltype(y)>::type i = (x), _##i = (y); i < _##i; ++i)
5  #define FORD(i, x, y) for (decay<decltype(x)>::type i = (x), _##i = (y); i > _##i; --i)
6  #ifdef zero1
7  #define dbg(x...) do { cout << "\033[32;1m" << #x << " -> "; err(x); } while (0)
8  void err() { cout << "\033[39;0m" << endl; }
9  template<template<typename...> class T, typename t, typename... A>
10 void err(T<t> a, A... x) { for (auto v: a) cout << v << ' '; err(x...); }
11 template<typename T, typename... A>
12 void err(T a, A... x) { cout << a << ' '; err(x...); }
13 #else
14 #define dbg(...)
15 #endif
16 // -----
```

数据结构

ST 表

- 二维

```
1  int f[maxn][maxn][10][10];
2  inline int highbit(int x) { return 31 - __builtin_clz(x); }
3  inline int calc(int x, int y, int xx, int yy, int p, int q) {
4      return max(
5          max(f[x][y][p][q], f[xx - (1 << p) + 1][yy - (1 << q) + 1][p][q]),
6          max(f[xx - (1 << p) + 1][y][p][q], f[x][yy - (1 << q) + 1][p][q])
7      );
8  }
9  void init() {
10     FOR (x, 0, highbit(n) + 1)
11     FOR (y, 0, highbit(m) + 1)
12     FOR (i, 0, n - (1 << x) + 1)
13     FOR (j, 0, m - (1 << y) + 1) {
14         if (!x && !y) { f[i][j][x][y] = a[i][j]; continue; }
15         f[i][j][x][y] = calc(
16             i, j,
17             i + (1 << x) - 1, j + (1 << y) - 1,
18             max(x - 1, 0), max(y - 1, 0)
19         );
20     }
21 }
22 inline int get_max(int x, int y, int xx, int yy) {
23     return calc(x, y, xx, yy, highbit(xx - x + 1), highbit(yy - y + 1));
24 }
```

树状数组

```
1  int t[N], n;
2
3  inline int lowbit(int x) { return x & (-x); }
4
5  void add(int x, int k) {
6      while (x <= n) {
7          t[x] += k;
8          x += lowbit(x);
9      }
10 }
11
12 int qry(int x) {
13     int ret = 0;
14 }
```

```

14     while (x) {
15         ret = max(ret,t[x]);
16         x -= lowbit(x);
17     }
18     return ret;
19 }

```

线段树

```

1  struct node {
2      ll sum;
3      ll plz, mlz;
4      int l, r;
5  } tree[N * 4];
6
7  void build(int i, int l, int r) {
8      tree[i].l = l;
9      tree[i].r = r;
10     tree[i].plz = 0;
11     tree[i].mlz = 1;
12     if (l == r) {
13         cin >> tree[i].sum;
14         tree[i].sum = tree[i].sum % p;
15         return;
16     }
17     int mid = (l + r) >> 1;
18     build(i * 2, l, mid);
19     build(i * 2 + 1, mid + 1, r);
20     tree[i].sum = (tree[i * 2].sum + tree[i * 2 + 1].sum) % p;
21 }
22
23 inline void push_down(ll i) {
24     ll k1 = tree[i].mlz, k2 = tree[i].plz;
25     tree[i << 1].sum = (tree[i << 1].sum * k1 + k2 * (tree[i << 1].r - tree[i << 1].l + 1)) % p;
26     tree[i << 1 | 1].sum = (tree[i << 1 | 1].sum * k1 + k2 * (tree[i << 1 | 1].r - tree[i << 1 | 1].l + 1)) % p;
27     tree[i << 1].mlz = (tree[i << 1].mlz * k1) % p;
28     tree[i << 1 | 1].mlz = (tree[i << 1 | 1].mlz * k1) % p;
29     tree[i << 1].plz = (tree[i << 1].plz * k1 + k2) % p;
30     tree[i << 1 | 1].plz = (tree[i << 1 | 1].plz * k1 + k2) % p;
31     tree[i].plz = 0;
32     tree[i].mlz = 1;
33 }
34
35 inline void add(int i, int l, int r, ll k) {
36     if (tree[i].l >= l && tree[i].r <= r) {
37         tree[i].sum = (tree[i].sum + k * (tree[i].r - tree[i].l + 1)) % p;
38         tree[i].plz += k;
39         return;
40     }
41     push_down(i);
42     if (tree[i * 2].r >= l) add(i * 2, l, r, k);
43     if (tree[i * 2 + 1].l <= r) add(i * 2 + 1, l, r, k);
44     tree[i].sum = tree[i * 2].sum + tree[i * 2 + 1].sum;
45 }
46
47 inline void mul(int i, int l, int r, ll k) {
48     if (tree[i].l >= l && tree[i].r <= r) {
49         tree[i].mlz = tree[i].mlz * k % p;
50         tree[i].plz = tree[i].plz * k % p;
51         tree[i].sum = tree[i].sum * k % p;
52         return;
53     }
54     push_down(i);
55     if (tree[i * 2].r >= l) mul(i * 2, l, r, k);
56     if (tree[i * 2 + 1].l <= r) mul(i * 2 + 1, l, r, k);
57     tree[i].sum = tree[i * 2].sum + tree[i * 2 + 1].sum;
58 }
59
60 ll search(int i, int l, int r) {
61     if (tree[i].l >= l && tree[i].r <= r) {
62         return tree[i].sum % p;

```

```

63     }
64     if (tree[i].r < l || tree[i].l > r) return 0;
65     push_down(i);
66     ll s = 0;
67     if (tree[i * 2].r >= l) s = (s + search(i * 2, l, r)) % p;
68     if (tree[i * 2 + 1].l <= r) s = (s + search(i * 2 + 1, l, r)) % p;
69     return s % p;
70 }

```

主席树

```

1  #include <bits/stdc++.h>
2  using namespace std;
3  typedef long long ll;
4  const int N = 2e5 + 7;
5  struct node{
6      int sum,ls,rs;
7  }tree[N << 5];
8  int tot = 0;
9  int a[N];
10 int v[N];
11 int rt[N];
12 int getid(int x,int n) {
13     return lower_bound(v + 1,v + n + 1,x) - v;
14 }
15 int build(int l,int r) {
16     int root = ++tot;
17     tree[root].sum = 0;
18     if (l >= r) return root;
19     int mid = l + r >> 1;
20     tree[root].ls = build(l, mid);
21     tree[root].rs = build(mid + 1, r);
22     return root;
23 }
24 int update(int x,int l, int r,int root) {
25     int d = ++tot;
26     tree[d].ls = tree[root].ls;
27     tree[d].rs = tree[root].rs;
28     tree[d].sum = tree[root].sum + 1;
29     if (l >= r) return d;
30     int mid = l + r >> 1;
31     if (x <= mid) tree[d].ls = update(x, l, mid, tree[d].ls);
32     else tree[d].rs = update(x, mid + 1, r, tree[d].rs);
33     return d;
34 }
35 int query(int u, int v, int l, int r, int k) {
36     if (l >= r) return l;
37     int mid = l + r >> 1;
38     int x = tree[tree[v].ls].sum - tree[tree[u].ls].sum;
39     if (x >= k) return query(tree[u].ls,tree[v].ls,l,mid,k);
40     else query(tree[u].rs, tree[v].rs, mid + 1, r, k - x);
41 }
42
43 void solve() {
44     tot = 0;
45     int n, m;
46     cin >> n >> m;
47     for (int i = 1; i <= n; ++i) {
48         cin >> a[i];
49         v[i] = a[i];
50     }
51     sort(v + 1,v + n + 1);
52     int len = unique(v + 1, v + n + 1) - v - 1;
53     rt[0] = build(1,len);
54     for (int i = 1; i <= n; ++i) {
55         rt[i] = update(getid(a[i],len),1,len,rt[i - 1]);
56     }
57     for (int i = 1; i <= m; ++i) {
58         int l, r, k;
59         cin >> l >> r >> k;
60         cout << v[query(rt[l - 1],rt[r],1,len,k)] << '\n';

```

```

61     }
62 }
63
64 # 数学
65
66 ## 中国剩余定理
67
68 C++
69 ll exgcd(ll a, ll b, LL &x, LL &y) {
70     if (!b) {
71         x = 1;
72         y = 0;
73         return a;
74     }
75     ll d = exgcd(b, a % b, x, y);
76     ll c = x;
77     x = y;
78     y = c - (a / b) * y;
79     return d;
80 }
81 ll a[N], b[N];
82 ll crt(int n) {
83     LL sum = 1;
84     LL res = 0;
85     for (int i = 1; i <= n; ++i) sum *= b[i];
86     for (int i = 1; i <= n; ++i) {
87         LL m = sum / b[i];
88         LL x, y;
89         exgcd(m, b[i], x, y);
90         a[i] = (a[i] + sum) % sum;
91         res = (res + a[i] * m * x % sum) % sum;
92     }
93     return (res % sum + sum) % sum;
94 }

```

FFT

```

1  const int N = 2e6 + 7, Log = 20;
2  const double Pi = acos(-1);
3  int n, m;
4
5  struct CP {
6      CP(double xx = 0, double yy = 0) { x = xx, y = yy; }
7
8      double x, y;
9
10     CP operator+(CP const &B) const { return CP(x + B.x, y + B.y); }
11
12     CP operator-(CP const &B) const { return CP(x - B.x, y - B.y); }
13
14     CP operator*(CP const &B) const { return CP(x * B.x - y * B.y, x * B.y + y * B.x); }
15 } f[N << 1]; //只用了一个复数数组
16 int tr[N << 1];
17
18 void fft(CP *f, bool flag) {
19     for (int i = 0; i < n; i++)
20         if (i < tr[i]) swap(f[i], f[tr[i]]);
21     for (int p = 2; p <= n; p <= 1) {
22         int len = p >> 1;
23         CP tG(cos(2 * Pi / p), sin(2 * Pi / p));
24         if (!flag) tG.y *= -1;
25         for (int k = 0; k < n; k += p) {
26             CP buf(1, 0);
27             for (int l = k; l < k + len; l++) {
28                 CP tt = buf * f[len + l];
29                 f[len + l] = f[l] - tt;
30                 f[l] = f[l] + tt;
31                 buf = buf * tG;
32             }
33         }
34     }

```

```

35 }
36
37 void solve() {
38     string a, b;
39     cin >> a >> b;
40     n = a.size();
41     m = b.size();
42     for (int i = 0; i < n; ++i) f[i].x = a[n - i - 1] - '0';
43     for (int i = 0; i < m; ++i) f[i].y = b[m - i - 1] - '0';
44     for (m += n, n = 1; n <= m; n <= 1);
45     for (int i = 0; i < n; ++i) tr[i] = (tr[i] >> 1) >> 1 | ((i & 1) ? n >> 1 : 0);
46     fft(f, 1);
47     for (int i = 0; i < n; ++i) f[i] = f[i] * f[i];
48     fft(f, 0);
49     for (int i = 0; i <= m; ++i) f[i].y = int(f[i].y / n / 2 + 0.49);
50 }

```

图论

LCA

```

1  vector<int> e[max_n];
2  int fa[max_n][25];
3  int d[max_n];
4  int vis[max_n];
5  int n;
6
7  void dfs(int x) {
8      vis[x]++;
9      for (int i : e[x]) {
10         if (!vis[i]) {
11             d[i] = d[x] + 1;
12             fa[i][0] = x;
13             dfs(i);
14         }
15     }
16 }
17
18 void bz() {
19     for (int j = 1; j <= 20; ++j) {
20         for (int i = 1; i <= n; ++i) {
21             fa[i][j] = fa[fa[i][j - 1]][j - 1];
22         }
23     }
24 }
25
26 int LCA(int u, int v) {
27     if (d[u] < d[v]) swap(u, v);
28     int de = d[u] - d[v];
29     for (int i = 0; i <= 20; ++i) {
30         if ((1 << i) & de) {
31             u = fa[u][i];
32         }
33     }
34     if (u == v) return u;
35     for (int i = 20; i >= 0; --i) {
36         if (fa[u][i] != fa[v][i]) {
37             u = fa[u][i];
38             v = fa[v][i];
39         }
40     }
41     return fa[u][0];
42 }

```

最短路

```

1  #include <bits/stdc++.h>
2
3  using namespace std;

```

```

4  typedef long long ll;
5  typedef pair<int, int> pii;
6  const ll INF = 0x3f3f3f3f3f3f3f3f;
7  const int inf = 0x3f3f3f3f;
8  const int N = 1e5 + 7;
9
10 vector<pii> e[N];
11 ll dis[N];
12 priority_queue<pii, vector<pii>, greater<>> q;
13 vector<int> v1;
14 int vis[N];
15
16 void init() {
17     memset(dis, inf, sizeof dis);
18     memset(vis, 0, sizeof vis);
19 }
20
21 void dij(int s, int n) {
22     v1.push_back(s);
23     int cnt = 0;
24     dis[s] = 0;
25     vis[s]++;
26     while (cnt < n - 1) {
27         for (auto &i : e[v1[cnt]]) {
28             if (vis[i.second]) continue;
29             q.push({i.first + dis[v1[cnt]], i.second});
30         }
31         while (vis[q.top().second]) q.pop();
32         v1.push_back(q.top().second);
33         vis[q.top().second]++;
34         dis[q.top().second] = q.top().first;
35         q.pop();
36         cnt++;
37     }
38 }

```

最大流

```

1  #include <bits/stdc++.h>
2
3  using namespace std;
4  typedef long long ll;
5  typedef pair<int, int> pii;
6  const ll INF = 0x3f3f3f3f3f3f3f3f;
7  const int inf = 0x3f3f3f3f;
8  const int N = 2e2 + 7;
9  typedef __int128 LL;
10 struct node {
11     int to, next;
12     ll val;
13 } e[20011];
14 int cur[2001];
15 int head[2001];
16 int cnt = 1;
17
18 void add(int u, int v, ll w) {
19     e[++cnt].val = w;
20     e[cnt].to = v;
21     e[cnt].next = head[u];
22     head[u] = cnt;
23 }
24
25 int s, t;
26 int d[2001];
27
28 int bfs() {
29     queue<int> q;
30     q.push(s);
31     memset(d, 0, sizeof d);
32     memcpy(cur, head, sizeof head);
33     d[s] = 1;

```



```

34     while (!q.empty()) {
35         int u = q.front();
36         q.pop();
37         for (int i = head[u]; i; i = e[i].next) {
38             int v = e[i].to;
39             if (e[i].val && !d[v]) q.push(v), d[v] = d[u] + 1;
40         }
41     }
42     return d[t];
43 }
44
45 ll dfs(ll u, ll mn) {
46     ll a;
47     if (u == t) return mn;
48     ll tmp = 0;
49     for (int &i = cur[u]; i && mn; i = e[i].next) {
50         int v = e[i].to;
51         if (e[i].val && d[v] == d[u] + 1) {
52             a = dfs(v, min(e[i].val, mn));
53             e[i].val -= a;
54             e[i ^ 1].val += a;
55             mn -= a;
56             tmp += a;
57         }
58     }
59     if (!mn) d[u] = -1;
60     return tmp;
61 }

```

费用流 (dij)

```

1  struct MFMC {
2      struct edge {
3          int v, next;
4          ll f, c;
5      } e[1000005];
6
7      struct node {
8          int v, e;
9      } p[1000005];
10
11     struct mypair {
12         ll dis;
13         int id;
14
15         bool operator<(const mypair &a) const { return dis > a.dis; }
16
17         mypair(ll d, int x) { dis = d, id = x; }
18     };
19
20     int head[500005], vis[500005];
21     ll dis[500005], h[500005];
22     int n, m, s, t, cnt = 1;
23     ll maxf, minc;
24
25     void addedge(int u, int v, ll f, ll c) {
26         e[++cnt].v = v;
27         e[cnt].f = f;
28         e[cnt].c = c;
29         e[cnt].next = head[u];
30         head[u] = cnt;
31     }
32
33     void add(int u, int v, ll f, ll c) {
34         addedge(u, v, f, c);
35         addedge(v, u, 0, -c);
36     }
37
38     bool dijkstra() {
39         priority_queue<mypair> q;
40         for (int i = 1; i <= n; i++) dis[i] = inf;

```

```

41     memset(vis, 0, sizeof(vis));
42     dis[s] = 0;
43     q.push(mypair(0, s));
44     while (!q.empty()) {
45         int u = q.top().id;
46         q.pop();
47         if (vis[u]) continue;
48         vis[u] = 1;
49         for (int i = head[u]; i; i = e[i].next) {
50             int v = e[i].v;
51             ll nc = e[i].c + h[u] - h[v];
52             if (e[i].f && dis[v] > dis[u] + nc) {
53                 dis[v] = dis[u] + nc;
54                 p[v].v = u;
55                 p[v].e = i;
56                 if (!vis[v]) q.push(mypair(dis[v], v));
57             }
58         }
59     }
60     return dis[t] != inf;
61 }
62
63 void spfa() {
64     queue<int> q;
65     memset(h, 63, sizeof(h));
66     h[s] = 0, vis[s] = 1;
67     q.push(s);
68     while (!q.empty()) {
69         int u = q.front();
70         q.pop();
71         vis[u] = 0;
72         for (int i = head[u]; i; i = e[i].next) {
73             int v = e[i].v;
74             if (e[i].f && h[v] > h[u] + e[i].c) {
75                 h[v] = h[u] + e[i].c;
76                 if (!vis[v]) {
77                     vis[v] = 1;
78                     q.push(v);
79                 }
80             }
81         }
82     }
83 }
84
85 void mfmc() {
86     maxf = 0;
87     minc = 0;
88     spfa();
89     while (dijkstra()) {
90         ll minf = inf;
91         for (int i = 1; i <= n; i++) h[i] += dis[i];
92         for (int i = t; i != s; i = p[i].v) minf = min(minf, e[p[i].e].f);
93         for (int i = t; i != s; i = p[i].v) {
94             e[p[i].e].f -= minf;
95             e[p[i].e ^ 1].f += minf;
96         }
97         maxf += minf;
98         minc += minf * h[t];
99     }
100 }
101
102 void init() {
103     cnt = 1;
104     memset(head, 0, sizeof head);
105 }
106 } mfmc;

```

费用流 (spfa)

```

1  const int N = 5e3 + 5, M = 1e5 + 5;
2  const int INF = 0x3f3f3f3f;

```

```

3  int n, m, tot = 1, lnk[N], cur[N], ter[M], nxt[M], cap[M], cost[M], dis[N], ret;
4  bool vis[N];
5
6  void add(int u, int v, int w, int c) {
7      ter[++tot] = v, nxt[tot] = lnk[u], lnk[u] = tot, cap[tot] = w, cost[tot] = c;
8  }
9
10 void addedge(int u, int v, int w, int c) { add(u, v, w, c), add(v, u, 0, -c); }
11
12 bool spfa(int s, int t) {
13     memset(dis, 0x3f, sizeof(dis));
14     memcpy(cur, lnk, sizeof(lnk));
15     std::queue<int> q;
16     q.push(s), dis[s] = 0, vis[s] = 1;
17     while (!q.empty()) {
18         int u = q.front();
19         q.pop(), vis[u] = 0;
20         for (int i = lnk[u]; i; i = nxt[i]) {
21             int v = ter[i];
22             if (cap[i] && dis[v] > dis[u] + cost[i]) {
23                 dis[v] = dis[u] + cost[i];
24                 if (!vis[v]) q.push(v), vis[v] = 1;
25             }
26         }
27     }
28     return dis[t] != INF;
29 }
30
31 int dfs(int u, int t, int flow) {
32     if (u == t) return flow;
33     vis[u] = 1;
34     int ans = 0;
35     for (int &i = cur[u]; i && ans < flow; i = nxt[i]) {
36         int v = ter[i];
37         if (!vis[v] && cap[i] && dis[v] == dis[u] + cost[i]) {
38             int x = dfs(v, t, std::min(cap[i], flow - ans));
39             if (x) ret += x * cost[i], cap[i] -= x, cap[i ^ 1] += x, ans += x;
40         }
41     }
42     vis[u] = 0;
43     return ans;
44 }
45
46 int mcmf(int s, int t) {
47     int ans = 0;
48     while (spfa(s, t)) {
49         int x;
50         while ((x = dfs(s, t, INF))) ans += x;
51     }
52     return ans;
53 }

```

差分约束

```

1  xa - xb >= c --> add(a, b, -c);
2  xa - xb <= c --> add(b, a, c);
3  xa == xb --> add(b, a, 0), add(a, b, 0);

```

Kruskal 重构树

```

1  struct kruskal {
2      struct edge {
3          int u, v, w;
4      } ed[N];
5
6      vector<int> e[N];
7      int w[N];
8      int n, m;
9      int fa[N];
10     int root;

```

```

11     int find(int x) {
12         return fa[x] == x ? x : fa[x] = find(fa[x]);
13     }
14
15     void init() {
16         for (int i = 1; i <= 2 * n; ++i) {
17             fa[i] = i;
18             e[i].clear();
19             w[i] = 0;
20         }
21         root = build();
22     }
23
24     int build() {
25         int cnt = n;
26         for (int i = 1; i <= m; ++i) {
27             int u = find(ed[i].u);
28             int v = find(ed[i].v);
29             if (u != v) {
30                 w[++cnt] = ed[i].w;
31                 fa[u] = fa[v] = cnt;
32                 e[cnt].push_back(u);
33                 e[cnt].push_back(v);
34             }
35         }
36         return cnt;
37     }
38
39 } k;

```

计算几何

二维几何：点与向量

```

1  #define y1 yy1
2  #define nxt(i) ((i + 1) % s.size())
3  typedef double LD;
4  const LD PI = 3.14159265358979323846;
5  const LD eps = 1E-10;
6  int sgn(LD x) { return fabs(x) < eps ? 0 : (x > 0 ? 1 : -1); }
7  struct L;
8  struct P;
9  typedef P V;
10 struct P {
11     LD x, y;
12     explicit P(LD x = 0, LD y = 0): x(x), y(y) {}
13     explicit P(const L& l);
14 };
15 struct L {
16     P s, t;
17     L() {}
18     L(P s, P t): s(s), t(t) {}
19 };
20
21 P operator + (const P& a, const P& b) { return P(a.x + b.x, a.y + b.y); }
22 P operator - (const P& a, const P& b) { return P(a.x - b.x, a.y - b.y); }
23 P operator * (const P& a, LD k) { return P(a.x * k, a.y * k); }
24 P operator / (const P& a, LD k) { return P(a.x / k, a.y / k); }
25 inline bool operator < (const P& a, const P& b) {
26     return sgn(a.x - b.x) < 0 || (sgn(a.x - b.x) == 0 && sgn(a.y - b.y) < 0);
27 }
28 bool operator == (const P& a, const P& b) { return !sgn(a.x - b.x) && !sgn(a.y - b.y); }
29 P::P(const L& l) { *this = l.t - l.s; }
30 ostream &operator << (ostream &os, const P &p) {
31     return (os << "(" << p.x << ", " << p.y << ")");
32 }
33 istream &operator >> (istream &is, P &p) {
34     return (is >> p.x >> p.y);
35 }
36

```

```

37 LD dist(const P& p) { return sqrt(p.x * p.x + p.y * p.y); }
38 LD dot(const V& a, const V& b) { return a.x * b.x + a.y * b.y; }
39 LD det(const V& a, const V& b) { return a.x * b.y - a.y * b.x; }
40 LD cross(const P& s, const P& t, const P& o = P()) { return det(s - o, t - o); }
41 // -----

```

点

```

1  const double eps = 1e-8;
2  const double pi = acos(-1.0);
3
4  int sgn(double x) {
5      if (fabs(x) < eps) return 0;
6      if (x < 0) return -1;
7      return 1;
8  }
9
10 struct Point {
11     double x, y;
12
13     Point() {}
14
15     Point(double _x, double _y) {
16         x = _x;
17         y = _y;
18     }
19
20     bool operator==(const Point &b) const {
21         return sgn(x - b.x) == 0 && sgn(y - b.y) == 0;
22     }
23
24     bool operator<(const Point &b) const {
25         return sgn(x - b.x) == 0 ? sgn(y - b.y) < 0 : x < b.x;
26     }
27
28     Point operator-(const Point &b) const {
29         return Point(x - b.x, y - b.y);
30     }
31
32     Point operator+(const Point &b) const {
33         return Point(x + b.x, y + b.y);
34     }
35
36     Point operator*(const double &k) const {
37         return Point(x * k, y * k);
38     }
39
40     Point operator/(const double &k) const {
41         return Point(x / k, y / k);
42     }
43
44     double operator^(const Point &b) const {
45         return x * b.y - y * b.x;
46     }
47
48     double operator*(const Point &b) const {
49         return x * b.x + y * b.y;
50     }
51
52     double len() {
53         return hypot(x, y);
54     }
55
56     double len2() {
57         return x * x + y * y;
58     }
59
60     double distance(Point p) {
61         return hypot(x - p.x, y - p.y);
62     }
63 }

```

```

64
65 double rad(Point a, Point b) {
66     Point p = *this;
67     return fabs(atan2(fabs((a - p) ^ (b - p)), (a - p) * (b - p)));
68 }
69
70 Point trunc(double r) {
71     double l = len();
72     if (!sgn(l)) return *this;
73     r /= l;
74     return Point(x * r, y * r);
75 }
76
77 Point rotright() {
78     return Point(-y, x);
79 }
80
81 Point rotright() {
82     return Point(y, -x);
83 }
84
85 Point rotate(Point p, double angle) {
86     Point v = (*this) - p;
87     double c = cos(angle), s = sin(angle);
88     return Point(p.x + v.x * c - v.y * s, p.y + v.x * s + v.y * c);
89 }
90 };

```

圆

```

1 struct circle {
2     Point p;
3     double r;
4
5     circle() {}
6
7     circle(Point _p, double _r) {
8         p = _p;
9         r = _r;
10    }
11
12    circle(double _x, double _y, double _r) {
13        p = Point(_x, _y);
14        r = _r;
15    }
16
17    bool operator==(circle v) const {
18        return (p == v.p) && sgn(r - v.r) == 0;
19    }
20
21    bool operator<(circle v) const {
22        return ((p < v.p) || ((p == v.p) && sgn(r - v.r) < 0));
23    }
24
25    double area() {
26        return pi * r * r;
27    }
28
29    double circumference() {
30        return 2 * pi * r;
31    }
32
33    int relation(Point b) {
34        double dis = b.distance(p);
35        if (sgn(dis - r) < 0) return 2; // 圆内
36        if (sgn(dis - r) == 0) return 1; // 圆上
37        return 0; // 圆外
38    }
39
40    int relation(circle v) {
41        double d = p.distance(v.p);

```

```

42     if (sgn(d - r - v.r) > 0) return 5;
43     if (sgn(d - r - v.r) == 0) return 4;
44     double l = fabs(r - v.r);
45     if (sgn(d - l) > 0) return 3;
46     if (sgn(d - l) == 0) return 2;
47     return 1;
48 }
49
50 int pointcrosscircle(circle v, Point &p1, Point &p2) {
51     int rel = relation(v);
52     if (rel == 5) return 0;
53     if (rel == 1) {
54         if (sgn(r - v.r) < 0) return 0;
55         return 3;
56     }
57     double d = p.distance(v.p);
58     double l = (d * d + r * r - v.r * v.r) / (2 * d);
59     double h = sqrt(r * r - l * l);
60     Point tmp = p + (v.p - p).trunc(l);
61     p1 = tmp + ((v.p - p).rotleft().trunc(h));
62     p2 = tmp + ((v.p - p).rotright().trunc(h));
63     if (rel == 2 || rel == 4) return 1;
64     return 2;
65 }
66 }

```

字符串

AC 自动机

```

1  #include <bits/stdc++.h>
2  using namespace std;
3  const int N = 1e6 + 7;
4  int z[N][26];
5  int fail[N];
6  int res[N];
7  int cnt = 0;
8  int re = 0;
9  int mp[N];
10 void insert(string s) {
11     int r = 0;
12     re++;
13     for (int i = 0; i < s.size(); ++i) {
14         if (!z[r][s[i] - 'a']) {
15             z[r][s[i] - 'a'] = ++cnt;
16         }
17         r = z[r][s[i] - 'a'];
18     }
19     mp[re] = r;
20 }
21 void bfs() {
22     queue<int> q;
23     for (int i = 0; i < 26; ++i) {
24         if (z[0][i]) {
25             fail[z[0][i]] = 0;
26             q.push(z[0][i]);
27         }
28     }
29     while (!q.empty()) {
30         int now = q.front();
31         q.pop();
32         for (int i = 0; i < 26; ++i) {
33             if (z[now][i]) {
34                 fail[z[now][i]] = z[fail[now]][i];
35                 q.push(z[now][i]);
36             }
37             else z[now][i] = z[fail[now]][i];
38         }
39     }
40 }

```

```

41 void quert(string s) {
42     int now = 0;
43     int ans = 0;
44     for (int i = 0; i < s.size(); ++i) {
45         now = z[now][s[i]-'a'];
46         for (int j = now; j < fail[j]) {
47             res[j]++;
48         }
49     }
50 }
51 int n;
52 string ss[155];
53 void solve() {
54     memset(z,0,sizeof z);
55     memset(res,0,sizeof res);
56     memset(fail,0,sizeof fail);
57     cnt = 0;
58     re = 0;
59     for (int i = 1; i <= n; ++i) {
60         cin >> ss[i];
61         insert(ss[i]);
62     }
63     bfs();
64     string s;
65     cin >> s;
66     quert(s);
67     int tmp = 0;
68     int ans;
69     for (int i = 1; i <= n; ++i) {
70         if (res[mp[i]] > tmp) {
71             tmp = res[mp[i]];
72             ans = i;
73         }
74     }
75     cout << res[mp[ans]] << '\n';
76     for (int i = 1; i <= n; ++i) {
77         if (res[mp[i]] == tmp) {
78             cout << ss[i] << '\n';
79         }
80     }
81 }
82 int main() {
83     ios::sync_with_stdio(0);
84     while (cin >> n && n) {
85         solve();
86     }
87 }

```

KMP

```

1 void get(string s) {
2     int j = 0, k = -1;
3     next[0] = -1;
4     while (j < s.length()) {
5         if (k == -1 || s[j] == s[k]) {
6             j++, k++;
7             if (s[j] != s[k]) {
8                 next[j] = k;
9             }
10            else next[j] = next[k];
11        }
12        else k = next[k];
13    }
14 }

```

SA (dc3)

```

1 //大小开 3 倍
2 /*
3     suffix[i]: 以 i 为起始位置的后缀

```



```

4     sa[i]: 排名第 i 的后缀的起始位置
5     rk[i]: 表示 suffix[i] 的排名
6     height[i]: suffix(sa[i-1]) 和 suffix(sa[i]) 的最长公共前缀
7         · h[i] = height[rak[i]], h[i] >= h[i-1]-1
8         · suffix[i] 和 suffix[j] 之间的最长公共前缀 = min(height[rak[i]+1]...height[rak[j]])
9  */
10
11 #define F(x) ((x) / 3 + ((x) % 3 == 1 ? 0 : tb))
12 #define G(x) ((x) < tb ? (x) * 3 + 1 : ((x) - tb) * 3 + 2)
13
14 int wa[N], wb[N], wss[N], wv[N], sa[N * 3];
15 int rk[N], height[N], r[N], lcp[N][30];
16
17
18 int c0(int *r, int a, int b) {
19     return r[a] == r[b] && r[a + 1] == r[b + 1] && r[a + 2] == r[b + 2];
20 }
21
22 int c12(int k, int *r, int a, int b) {
23     if (k == 2)
24         return r[a] < r[b] || r[a] == r[b] && c12(1, r, a + 1, b + 1);
25     return r[a] < r[b] || r[a] == r[b] && wv[a + 1] < wv[b + 1];
26 }
27
28 void Rsort(int *r, int *a, int *b, int n, int m) {
29     for (int i = 0; i < n; i++) wv[i] = r[a[i]];
30     for (int i = 0; i < m; i++) wss[i] = 0;
31     for (int i = 0; i < n; i++) wss[wv[i]]++;
32     for (int i = 1; i < m; i++) wss[i] += wss[i - 1];
33     for (int i = n - 1; i >= 0; i--) b[--wss[wv[i]]] = a[i];
34 }
35
36 void dc3(int *r, int *sa, int n, int m) {
37     int i, j, *rn = r + n, *san = sa + n, ta = 0, tb = (n + 1) / 3, tbc = 0, p;
38     r[n] = r[n + 1] = 0;
39     for (i = 0; i < n; i++) if (i % 3 != 0) wa[tbc++] = i;
40     Rsort(r + 2, wa, wb, tbc, m);
41     Rsort(r + 1, wb, wa, tbc, m);
42     Rsort(r, wa, wb, tbc, m);
43     for (p = 1, rn[F(wb[0])] = 0, i = 1; i < tbc; i++)
44         rn[F(wb[i])] = c0(r, wb[i - 1], wb[i]) ? p - 1 : p++;
45     if (p < tbc) dc3(rn, san, tbc, p);
46     else for (i = 0; i < tbc; i++) san[rn[i]] = i;
47     for (i = 0; i < tbc; i++) if (san[i] < tb) wb[ta++] = san[i] * 3;
48     if (n % 3 == 1) wb[ta++] = n - 1;
49     Rsort(r, wb, wa, ta, m);
50     for (i = 0; i < tbc; i++) wv[wb[i]] = G(san[i]);
51     for (i = 0, j = 0, p = 0; i < ta && j < tbc; p++)
52         sa[p] = c12(wb[j] % 3, r, wa[i], wb[j]) ? wa[i++] : wb[j++];
53     for (; i < ta; p++) sa[p] = wa[i++];
54     for (; j < tbc; p++) sa[p] = wb[j++];
55 }
56
57 void calHeight(int *r, int *sa, int n) {
58     int i, j, k = 0;
59     for (i = 1; i <= n; i++) rk[sa[i]] = i;
60     for (i = 0; i < n; height[rk[i++]] = k)
61         for (k ? k-- : 0, j = sa[rk[i] - 1]; r[i + k] == r[j + k]; k++);
62
63     // for (int i = 1; i <= n; ++i) {
64     //     dbg(i, height[i]);
65     // }
66     // 用 getLcp 要去掉下面的注释
67     // for (int i = 1; i <= n; ++i) lcp[i][0] = height[i];
68     // for (int l = 1; (1 << l) <= n; l++) {
69     //     for (int i = 1; i + (1 << l) - 1 <= n; ++i) {
70     //         lcp[i][l] = min(lcp[i][l - 1], lcp[i + (1 << (l - 1))][l - 1]);
71     //     }
72     // }
73 }
74

```

```

75 int getLcp(int i, int j, int n) {
76     if (i == j) return n - i;
77     int l = rk[i], r = rk[j];
78     if (l > r) swap(l, r);
79     l++;
80     int k = __lg(r - l + 1);
81     return min(lcp[l][k], lcp[r - (1 << k) + 1][k]);
82 }
83
84 char s[N];
85
86 void solve() {
87     int n = 0;
88     cin >> s;
89     for (int i = 0; s[i]; ++i) {
90         r[n++] = s[i];
91     }
92     r[n] = 0;
93     dc3(r, sa, n + 1, 256);
94     calHeight(r, sa, n);
95     for (int i = 1; i <= n; ++i) {
96         cout << sa[i] + 1 << " \n" [i == n];
97     }
98     for (int i = 1; i <= n; ++i) {
99         cout << height[i] << " \n" [i == n];
100     }
101     // 不同子串个数
102     /*
103     ll ans = 1ll * n * (n + 1) / 2;
104     for (int i = 1; i <= n; ++i) {
105         ans -= height[i];
106     }
107     // 两个串的最长公共子串
108     int n = 0;
109     scanf("%s", s);
110     scanf("%s", t);
111     int l = strlen(s);
112     s[l] = '!';
113     int tag = l;
114     for (int i = 0; t[i]; ++i) {
115         s[++l] += t[i];
116     }
117     for (int i = 0; s[i]; ++i) {
118         r[n++] = s[i];
119     }
120     r[n] = 0;
121     dc3(r, sa, n + 1, 256);
122     calHeight(r, sa, n);
123     int ans = 0;
124     for (int i = 1; i <= n; ++i) {
125         int x1 = sa[i - 1], x2 = sa[i];
126         if ((x1 < tag && x2 > tag) || (x1 > tag && x2 < tag)) {
127             ans = max(ans, height[i]);
128         }
129     }
130     printf("%lld", ans);
131     //不同公共子串的个数
132     ll ans = 0;
133     int tmp = 0;
134     for (int i = 1; i <= n; ++i) {
135         int x1 = sa[i - 1], x2 = sa[i];
136         if ((x1 < tag && x2 > tag) || (x1 > tag && x2 < tag)) {
137             ans += height[i];
138             if (tmp > 0) ans -= min(getLcp(sa[i], sa[tmp], n), getLcp(sa[i - 1], sa[tmp - 1], n));
139             tmp = i;
140         }
141     }
142     printf("%lld", ans);
143     */
144 }

```

SAM

```
1 struct SAM {
2     struct state {
3         int len, link;
4         map<char, int> next;
5     };
6
7     state st[N * 2];
8     int sz, last;
9     int cnt[N * 2];
10    int siz[N * 2];
11    int a[N * 2];
12
13    void init() {
14        st[0].len = 0;
15        st[0].link = -1;
16        sz = 1;
17        last = 0;
18    }
19
20    void extend(char c) {
21        int cur = sz++;
22        st[cur].len = st[last].len + 1;
23        int p = last;
24        while (p != -1 && !st[p].next.count(c)) {
25            st[p].next[c] = cur;
26            p = st[p].link;
27        }
28        if (p == -1) {
29            st[cur].link = 0;
30        } else {
31            int q = st[p].next[c];
32            if (st[p].len + 1 == st[q].len) {
33                st[cur].link = q;
34            } else {
35                int clone = sz++;
36                st[clone].len = st[p].len + 1;
37                st[clone].next = st[q].next;
38                st[clone].link = st[q].link;
39                while (p != -1 && st[p].next[c] == q) {
40                    st[p].next[c] = clone;
41                    p = st[p].link;
42                }
43                st[q].link = st[cur].link = clone;
44            }
45        }
46        last = cur;
47        siz[cur]++;
48    }
49
50    void run() { // 求子串出现次数
51        ll ans = 0;
52        for (int i = 1; i <= sz; ++i) cnt[st[i].len]++;
53        for (int i = 1; i <= sz; ++i) cnt[i] += cnt[i - 1];
54        for (int i = 1; i <= sz; ++i) a[cnt[st[i].len] - 1] = i;
55        for (int i = sz; i; --i) {
56            int p = a[i];
57            siz[st[p].link] += siz[p];
58            if (siz[p] > 1) ans = max(ans, 1LL * siz[p] * st[p].len);
59        }
60        cout << ans;
61    }
62    /* 不同子串个数
63    void run1() {
64        for (int i = 1; i <= sz; ++i) cnt[st[i].len]++;
65        for (int i = 1; i <= sz; ++i) cnt[i] += cnt[i - 1];
66        for (int i = 1; i <= sz; ++i) a[cnt[st[i].len] - 1] = i;
67        for (int i = sz; i >= 0; --i) {
68            int p = a[i];
69            siz[p] = 1;
```

```

70         for (auto j : st[p].next) {
71             siz[p] += siz[j.second];
72         }
73     }
74     cout << siz[0] - 1;
75 }
76 void run2(int x) {
77     dp[x] = 1;
78     for (auto i : st[x].next) {
79         if (!dp[i.second]) run2(i.second);
80         dp[x] += dp[i.second];
81     }
82 }
83 dp[0] = 1;
84 void run3() {
85     ll ans = 0;
86     for (int i = 1; i <= sz; ++i) {
87         ans += st[i].len - st[st[i].link].len;
88     }
89     cout << ans;
90 }
91 */
92
93 string lcs(string t) { //最长公共子串
94     int v = 0, l = 0, mx = 0, mx_end = 0;
95     for (int i = 1; i <= t.size(); ++i) {
96         while (v && !st[v].next.count(t[i - 1])) {
97             v = st[v].link;
98             l = st[v].len;
99         }
100         if (st[v].next.count(t[i - 1])) {
101             v = st[v].next[t[i - 1]];
102             l++;
103         }
104         if (l > mx) {
105             mx = l;
106             mx_end = i;
107         }
108     }
109     return t.substr(mx_end - mx + 1, mx);
110 }
111 } sam;
112
113
114 void solve() {
115     string s;
116     cin >> s;
117     sam.init();
118     for (char i: s) {
119         sam.extend(i);
120     }
121 }

```

PAM

```

1  struct PAM {
2      struct PAM_Trie {
3          int ch[26];
4          int fail, len, num, pre; //pre 回文串个数, num 以 i 结尾回文串个数
5      } b[N];
6      int last, cnt, s[N];
7
8      void init() {
9          s[0] = 26;
10         b[0].len = 0;
11         b[1].len = -1;
12         b[0].fail = 1;
13         b[1].fail = 0;
14         last = 0;
15         cnt = 1;
16     }

```

```

17
18 int get_fail(int x, int n) {
19     while (s[n - b[x].len - 1] != s[n]) x = b[x].fail;
20     return x;
21 }
22
23 void insert(int n) {
24     int p = get_fail(last, n);
25     if (!b[p].ch[s[n]]) {
26         b[++cnt].len = b[p].len + 2;
27         int tmp = get_fail(b[p].fail, n);
28         b[cnt].fail = b[tmp].ch[s[n]];
29         b[cnt].num = b[b[cnt].fail].num + 1;
30         b[p].ch[s[n]] = cnt;
31     }
32     last = b[p].ch[s[n]];
33     b[last].pre++;
34 }
35
36 ll run() {
37     ll ans = 0;
38     for (int i = cnt; i ; --i) {
39         b[b[i].fail].pre += b[i].pre;
40         ans = max(ans, 1ll * b[i].pre * b[i].len);
41     }
42     return ans;
43 }
44
45 } pam;
46
47 void solve() {
48     string s;
49     cin >> s;
50     s = ' ' + s;
51     int ans = 0;
52     pam.init();
53     for (int i = 1; i < s.size(); ++i) {
54         s[i] = (s[i] - 97 + ans) % 26 + 97;
55         pam.s[i] = s[i] - 'a';
56         pam.insert(i);
57         ans = pam.b[pam.last].num;
58         cout << ans << ' ';
59     }
60 }

```

杂项

STL

- copy

```

1 template <class InputIterator, class OutputIterator>
2 OutputIterator copy (InputIterator first, InputIterator last, OutputIterator result);

```

int_128

```

1 typedef __int128 ll;
2 inline __int128 read() {
3     __int128 x = 0, f = 1;
4     char ch = getchar();
5     while (ch < '0' || ch > '9') {
6         if (ch == '-') f = -1;
7         ch = getchar();
8     }
9     while (ch >= '0' && ch <= '9') {
10         x = x * 10 + ch - '0';
11         ch = getchar();
12     }
13     return x * f;

```

```

14 }
15
16 inline void print(__int128 x) {
17     if (x < 0) {
18         putchar(-1);
19         x = -x;
20     }
21     if (x > 9) print(x / 10);
22     putchar(x % 10 + '0');
23 }

```

背包

```

1  int val[105];
2  int num[105];
3  int nva;
4  int dp[100010];
5
6  void ZeroOnePack(int cost, int weight) {
7      for (int i = nva; i >= cost; i--)
8          dp[i] = max(dp[i], dp[i - cost] + weight);
9  }
10
11 void CompletePack(int cost, int weight) {
12     for (int i = cost; i <= nva; ++i)
13         dp[i] = max(dp[i], dp[i - cost] + weight);
14 }
15
16 void MultiplePack(int cost, int weight, int amount) {
17     if (cost * amount >= nva) {
18         CompletePack(cost, weight);
19         return;
20     }
21     for (int i = 1; i < amount; i <= 1) {
22         ZeroOnePack(cost * i, weight * i);
23         amount -= i;
24     }
25     if (amount > 0) ZeroOnePack(cost * amount, weight * amount);
26 }
27
28 int main() {
29     int n, m;
30     while (cin >> n >> m && n && m) {
31         for (int i = 0; i < n; ++i) {
32             cin >> val[i];
33         }
34         for (int i = 0; i < n; ++i) {
35             cin >> num[i];
36         }
37         nva = m;
38         memset(dp, 0, sizeof dp);
39         for (int i = 0; i < n; ++i) {
40             MultiplePack(val[i], val[i], num[i]);
41         }
42         int ans = 0;
43         for (int i = 1; i <= m; ++i) {
44             if (dp[i] == i) ans++;
45         }
46         cout << ans << endl;
47     }
48 }

```

高精度

```

1  string a[100][100];
2  int t[100];
3  int w[100];
4  string ans;
5  string pl(string x, string y) {
6      reverse(x.begin(), x.end());

```

```

7     reverse(y.begin(),y.end());
8     int z[1000];
9     memset(z,0,sizeof z);
10    for (int i = 0; i < min(x.size(),y.size()); ++i) {
11        z[i] = x[i] - '0' + y[i] - '0';
12    }
13    for (int i = min(x.size(),y.size()); i < max(x.size(),y.size()); ++i) {
14        if (x.size() < y.size()) z[i] = y[i] - '0';
15        else z[i] = x[i] - '0';
16    }
17    int cnt = max(x.size(),y.size());
18    for (int i = 0; i < cnt; ++i) {
19        If (z[i] >= 10) {
20            z[i + 1] += (z[i]) / 10;
21            z[i] = (z[i]) % 10;
22            if (i == cnt - 1) cnt++;
23        }
24    }
25    string p;
26    for (int i = 0; i < cnt; ++i) {
27        p += z[i] + '0';
28    }
29    reverse(p.begin(),p.end());
30    return p;
31 }
32 string me(string x, string y) {
33     reverse(x.begin(),x.end());
34     reverse(y.begin(),y.end());
35     int z[1000];
36     memset(z,0,sizeof z);
37     for (int i = 0; i < y.size(); ++i) {
38         for (int j = 0; j < x.size(); ++j) {
39             z[i + j] += (x[j] - '0') * (y[i] - '0');
40         }
41     }
42     int cnt = max(x.size(),y.size());
43     for (int i = 0; i < cnt; ++i) {
44         if (z[i] >= 10) {
45             z[i + 1] += (z[i]) / 10;
46             z[i] = (z[i]) % 10;
47             if (i == cnt - 1) cnt++;
48         }
49     }
50     string p;
51     for (int i = 0; i < cnt; ++i) {
52         p += z[i] + '0';
53     }
54     reverse(p.begin(),p.end());
55     return p;
56 }

```