Standard Code Library

111

zu

July 31, 2022

Contents

一切的开始

宏定义

- 需要 C++11

```cpp
#include <bits/stdc++.h>
using namespace std;
using LL = long long;
#define FOR(i, x, y) for (decay<decltype(y)>::type i = (x), _##i = (y); i < _##i; ++i)
#define FORD(i, x, y) for (decay<decltype(x)>::type i = (x), _##i = (y); i > _##i; --i)
#ifdef zerol
#define dbg(x...) do { cout << "\033[32;1m" << #x << " -> "; err(x); } while (0)
void err() { cout << "\033[39;0m" << endl; }
template<template<typename...> class T, typename t, typename... A>
void err(T<t> a, A... x) { for (auto v: a) cout << v << ' '; err(x...); }
template<typename T, typename... A>
void err(T a, A... x) { cout << a << ' '; err(x...); }
#else
#define dbg(...)
#endif
// -----------------------------------------------------------------------
```

数据结构

ST 表

- 二维

```cpp
int f[maxn][maxn][10][10];
inline int highbit(int x) { return 31 - __builtin_clz(x); }
inline int calc(int x, int y, int xx, int yy, int p, int q) {
    return max(
        max(f[x][y][p][q], f[xx - (1 << p) + 1][yy - (1 << q) + 1][p][q]),
        max(f[xx - (1 << p) + 1][y][p][q], f[x][yy - (1 << q) + 1][p][q])
    );
}
void init() {
    FOR (x, 0, highbit(n) + 1)
    FOR (y, 0, highbit(m) + 1)
        FOR (i, 0, n - (1 << x) + 1)
        FOR (j, 0, m - (1 << y) + 1) {
            if (!x && !y) { f[i][j][x][y] = a[i][j]; continue; }
            f[i][j][x][y] = calc(
                i, j,
                i + (1 << x) - 1, j + (1 << y) - 1,
                max(x - 1, 0), max(y - 1, 0)
            );
        }
}
inline int get_max(int x, int y, int xx, int yy) {
    return calc(x, y, xx, yy, highbit(xx - x + 1), highbit(yy - y + 1));
}
```

树状数组

```cpp
int t[N], n;

inline int lowbit(int x) { return x & (-x); }

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

int qry(int x) {
    int ret = 0;
```

```
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';
```

```c++
        }
    }
}
```

数学

类欧几里得

* $m = \lfloor \frac{an+b}{c} \rfloor$.
* $f(a,b,c,n)=\sum_{i=0}^n\lfloor\frac{ai+b}{c}\rfloor$: 当 $a \ge c$ or $b \ge c$ 时, $f(a,b,c,n)=(\frac{a}{c})n(n+1)/2+(\frac{b}{c})(n+1)+f(a \bmod c,b \bmod c,c,n)$; 否则 $f(a,b,c,n)=nm-f(c,c-b-1,a,m-1)$。
* $g(a,b,c,n)=\sum_{i=0}^n i \lfloor\frac{ai+b}{c}\rfloor$: 当 $a \ge c$ or $b \ge c$ 时, $g(a,b,c,n)=(\frac{a}{c})n(n+1)(2n+1)/6+(\frac{b}{c})n(n+1)/2+g(a \bmod c,b \bmod c,c,n)$; 否则 $g(a,b,c,n)=\frac{1}{2} (n(n+1)m-f(c,c-b-1,a,m-1)-h(c,c-b-1,a,m-1))$。
* $h(a,b,c,n)=\sum_{i=0}^n\lfloor \frac{ai+b}{c} \rfloor^2$: 当 $a \ge c$ or $b \ge c$ 时, $h(a,b,c,n)=(\frac{a}{c})^2 n(n+1)(2n+1)/6 +(\frac{b}{c})^2 (n+1)+(\frac{a}{c})(\frac{b}{c})n(n+1)+h(a \bmod c, b \bmod c,c,n)+2(\frac{a}{c})g(a \bmod c,b \bmod c,c,n)+2(\frac{b}{c})f(a \bmod c,b \bmod c,c,n)$; 否则 $h(a,b,c,n)=nm(m+1)-2g(c,c-b-1,a,m-1)-2f(c,c-b-1,a,m-1)-f(a,b,c,n)$。

中国剩余定理

```c++
ll exgcd(ll a, ll b, LL &x, LL &y) {
    if (!b) {
        x = 1;
        y = 0;
        return a;
    }
    ll d = exgcd(b, a % b, x, y);
    ll c = x;
    x = y;
    y = c - (a / b) * y;
    return d;
}
ll a[N],b[N];
ll crt(int n) {
    LL sum = 1;
    LL res = 0;
    for (int i = 1; i <= n; ++i) sum *= b[i];
    for (int i = 1; i <= n; ++i) {
        LL m = sum / b[i];
        LL x,y;
        exgcd(m,b[i],x,y);
        a[i] = (a[i] + sum) % sum;
        res = (res + a[i] * m * x % sum) % sum;
    }
    return (res % sum + sum) % sum;
}
```

图论

LCA

```
vector<int> e[max_n];
int fa[max_n][25];
int d[max_n];
int vis[max_n];
int n;

void dfs(int x) {
    vis[x]++;
    for (int i : e[x]) {
        if (!vis[i]) {
            d[i] = d[x] + 1;
            fa[i][0] = x;
            dfs(i);
        }
    }
}

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
```

```cpp
using namespace std;
typedef long long ll;
typedef pair<int, int> pii;
const ll INF = 0x3f3f3f3f3f3f3f3f;
const int inf = 0x3f3f3f3f;
const int N = 2e2+ 7;
typedef __int128 LL;
struct node {
    int to, next;
    ll val;
} e[20011];
int cur[2001];
int head[2001];
int cnt = 1;

void add(int u, int v, ll w) {
    e[++cnt].val = w;
    e[cnt].to = v;
    e[cnt].next = head[u];
    head[u] = cnt;
}

int s, t;
int d[2001];

int bfs() {
    queue<int> q;
    q.push(s);
    memset(d, 0, sizeof d);
    memcpy(cur, head, sizeof head);
    d[s] = 1;
    while (!q.empty()) {
        int u = q.front();
        q.pop();
        for (int i = head[u]; i; i = e[i].next) {
            int v = e[i].to;
            if (e[i].val && !d[v]) q.push(v), d[v] = d[u] + 1;
        }
    }
    return d[t];
}

ll dfs(ll u, ll mn) {
    ll a;
    if (u == t) return mn;
    ll tmp = 0;
    for (int &i = cur[u]; i && mn; i = e[i].next) {
        int v = e[i].to;
        if (e[i].val && d[v] == d[u] + 1) {
            a = dfs(v, min(e[i].val, mn));
            e[i].val -= a;
            e[i ^ 1].val += a;
            mn -= a;
            tmp += a;
        }
    }
    if (!mn) d[u] = -1;
    return tmp;
}
```

费用流 (dij)

```cpp
struct MFMC {
    struct edge {
        int v, next;
        ll f, c;
    } e[1000005];

    struct node {
        int v, e;
    } p[100005];
```

```
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[50005], vis[50005];
21    ll dis[50005], h[50005];
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  // ----------------------------------------
```

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;i
9   nt 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      }
```

```cpp
        while (!q.empty()) {
            int now = q.front();
            q.pop();
            for (int i = 0; i < 26; ++i) {
                if (z[now][i]) {
                    fail[z[now][i]] = z[fail[now]][i];
                    q.push(z[now][i]);
                }
                else z[now][i] = z[fail[now]][i];
            }
        }
}
void quert(string s) {
    int now = 0;
    int ans = 0;
    for (int i = 0; i < s.size(); ++i) {
        now = z[now][s[i]-'a'];
        for (int j = now;j ; j = fail[j]) {
            res[j]++;;
        }
    }
}int n;
string ss[155];
void solve() {
    memset(z,0,sizeof z);
    memset(res,0,sizeof res);
    memset(fail,0,sizeof fail);
    cnt = 0;
    re = 0;
    for (int i = 1; i <= n; ++i) {
        cin >> ss[i];
        insert(ss[i]);
    }
    bfs();
    string s;
    cin >> s;
    quert(s);
    int tmp = 0;
    int ans;
    for (int i = 1; i <= n; ++i) {
        if (res[mp[i]] > tmp) {
            tmp = res[mp[i]];
            ans = i;
        }
    }
    cout << res[mp[ans]] << '\n';
    for (int i = 1; i <= n; ++i) {
        if (res[mp[i]] == tmp) {
            cout << ss[i] << '\n';
        }
    }

}
int main() {
    ios::sync_with_stdio(0);
    while (cin >> n && n) {
        solve();
    }
}
```

KMP

```cpp
void get(string s) {
    int j = 0, k = -1;
    next[0] = -1;
    while (j < s.length()) {
        if (k == -1 || s[j] == s[k]) {
            j++, k++;
            if (s[j] != s[k]) {
                next[j] = k;
            }
```

```
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])] = 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]--] = 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);
```

```cpp
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]--] = 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 }
```

杂项

STL

- copy

16

```cpp
template <class InputIterator, class OutputIterator>
  OutputIterator copy (InputIterator first, InputIterator last, OutputIterator result);
```