

# Standard Code Library

111

zu

July 28, 2022

# Contents

- 一切的开始 . . . . . 2
  - 宏定义 . . . . . 2
- 数据结构 . . . . . 2
  - ST 表 . . . . . 2
- 数学 . . . . . 2
  - 类欧几里得 . . . . . 2
- 图论 . . . . . 3
  - LCA . . . . . 3
- 计算几何 . . . . . 3
  - 二维几何：点与向量 . . . . . 3
- 字符串 . . . . . 4
  - AC 自动机 . . . . . 4
  - KMP . . . . . 5
  - SA (dc3) . . . . . 5
  - SAM . . . . . 7
- 杂项 . . . . . 9
  - STL . . . . . 9

## 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 }
```

## 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  $a \geq c$  或  $b \geq c$  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  $a \geq c$  或  $b \geq c$  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  $a \geq c$  或  $b \geq c$  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

$$h(a, b, c, n) = nm(m+1) - 2g(c, c-b-1, a, m-1) - 2f(c, c-b-1, a, m-1) - f(a, b, c, n)。$$

## 图论

### LCA

- 倍增

```

1 void dfs(int u, int fa) {
2     pa[u][0] = fa; dep[u] = dep[fa] + 1;
3     FOR (i, 1, SP) pa[u][i] = pa[pa[u][i-1]][i-1];
4     for (int& v: G[u]) {
5         if (v == fa) continue;
6         dfs(v, u);
7     }
8 }
9
10 int lca(int u, int v) {
11     if (dep[u] < dep[v]) swap(u, v);
12     int t = dep[u] - dep[v];
13     FOR (i, 0, SP) if (t & (1 << i)) u = pa[u][i];
14     FOR (i, SP-1, -1) {
15         int uu = pa[u][i], vv = pa[v][i];
16         if (uu != vv) { u = uu; v = vv; }
17     }
18     return u == v ? u : pa[u][0];
19 }

```

## 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;
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; 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 }

```

```

61     }
62     bfs();
63     string s;
64     cin >> s;
65     quart(s);
66     int tmp = 0;
67     int ans;
68     for (int i = 1; i <= n; ++i) {
69         if (res[mp[i]] > tmp) {
70             tmp = res[mp[i]];
71             ans = i;
72         }
73     }
74     cout << res[mp[ans]] << '\n';
75     for (int i = 1; i <= n; ++i) {
76         if (res[mp[i]] == tmp) {
77             cout << ss[i] << '\n';
78         }
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]) = 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);
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

```

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

```