XCPC - Templates

north-h

Contents

1	数据	结构	2
	1.1	树状数组(单点修改,区间查询)	2
	1.2	树状数组(区间修改,单点查询)	2
	1.3	树状数组(区间修改,区间查询)	2
	1.4	线段树(朴素)	3
	1.5	线段树 (lazy)	5
	1.6	ST 表	6
	1.7	二维树状数组	7
	1.8	并查集	
2	字符串		
	2.1	单哈希	8
	2.2	双哈希	8
	2.3	自然溢出	9
	2.4	字典树	9
	2.5		10
	2.6		10
3	杂项		11
	3.1		11
	3.2	取模	11

SMU 第2页

1 数据结构

1.1 树状数组(单点修改,区间查询)

```
template <class T>
 2
    struct BIT {
 3
       vector<T> tr;
 4
       int n;
 5
       BIT(int N) {
 6
           n = N;
 7
           tr.resize(n + 1);
 8
 9
       void add(int x, T k) {
           for(int i = x; i < n; i += (i & -i))</pre>
10
11
              tr[i] += k;
12
       }
       T query(int x) {
13
14
           T res = 0;
15
           for(int i = x; i; i -= (i & -i))
16
              res += tr[i];
17
           return res;
18
       }
19
       T range_query(int 1, int r) {
20
           return query(r) - query(l - 1);
21
       }
22
    };
```

1.2 树状数组(区间修改,单点查询)

```
template <class T>
    struct BIT {
 2
       vector<T> tr;
 3
 4
       int n;
 5
       BIT(int N) {
 6
           n = N;
 7
           tr.resize(n + 1);
 8
       }
9
       void add(int x, T k) {
10
           for(int i = x; i < n; i += (i & -i))</pre>
              tr[i] += k;
11
12
       }
13
       void range_add(int 1, int r) {
           add(1, k);
14
15
           add(r + 1, -k);
16
       }
       T query(int x) {
17
18
           T res = 0;
19
           for(int i = x; i; i -= (i & -i))
20
              res += tr[i];
21
           return res;
22
       }
    };
23
```

SMU 第3页

1.3 树状数组(区间修改,区间查询)

```
template <class T>
 2
    struct BIT {
 3
       vector<T> sum1, sum2;
 4
       int n;
 5
       BIT() {}
 6
       void init(int N) {
 7
           n = N;
 8
           sum1.resize(n);
 9
           sum2.resize(n);
10
       }
11
       void add(int x, T k) {
12
           for(int i = x; i < n; i += (i & -i))</pre>
              sum1[i] += k, sum2[i] += x * k;
13
14
       }
15
       void add(int 1, int r, T x) {
16
           add(1, x), add(r + 1, -x);
17
       }
18
       T query(int x) {
19
           T res = 0;
20
           for(int i = x; i > 0; i -= (i \& -i))
21
              res += (x + 1) * sum1[i] - sum2[i];
22
           return res;
23
       }
24
       T query(int 1, int r) {
25
           return query(r) - query(l - 1);
26
27
       }
28
    };
```

1.4 线段树

```
template <class T>
 1
 2
    struct Seg{
 3
       struct Node{ int 1, r, sum, lazy; };
 4
 5
       vector<Node> tr;
 6
       vector<int> a;
 7
       int n;
 8
9
       Seg() {};
10
       void init(int N) {
11
12
          n = N;
13
          tr.resize(n * 4);
14
           a.resize(n);
15
       }
16
17
       void add(int x, int k) {
          a[x] = k;
18
19
       }
```

SMU 第4页

```
20
21
       void pushup(int u) {
22
           tr[u].sum = tr[u << 1].sum + tr[u << 1 | 1].sum;
23
        }
24
25
       void pushdown(int u) {
           if (tr[u].lazy) {
26
27
               tr[u << 1].sum += tr[u].lazy * (tr[u << 1].r - tr[u << 1].l + 1);
               tr[u << 1 \mid 1].sum += tr[u].lazy * (tr[u << 1 \mid 1].r - tr[u << 1 \mid 1].l +
28
                   1);
               tr[u << 1].lazy += tr[u].lazy;</pre>
29
30
               tr[u << 1 | 1].lazy += tr[u].lazy;
31
               tr[u].lazy = 0;
32
           }
33
34
35
       void build(int u, int l,int r) {
36
           tr[u] = {1, r, a[1], 0};
37
           if (1 == r)return;
38
           int mid = l + r \gg 1;
39
           pushdown(u);
40
           build(u << 1, 1, mid);</pre>
41
           build(u << 1 | 1, mid + 1, r);
42
           pushup(u);
43
       }
       //区间修改
44
        void modify(int u,int l,int r,int k) {
45
46
           if (tr[u].l >= 1 && tr[u].r <= r) {</pre>
               tr[u].sum += (tr[u].r - tr[u].l + 1) * k;
47
48
               tr[u].lazy += k;
49
               return;
50
           }
51
           pushdown(u);
52
           int mid = tr[u].l + tr[u].r >> 1;
53
           if (1 <= mid)modify(u << 1, 1, r, k);</pre>
54
           if (r > mid) modify(u << 1 | 1, 1, r, k);</pre>
55
           pushup(u);
56
       }
57
       //单点修改
58
        void modify(int u,int x,int k) {
59
           if (tr[u].1 == tr[u].r) {
60
               tr[u].sum += k;
               return;
61
62
           int mid = tr[u].l + tr[u].r >> 1;
63
64
           if (x <= mid)modify(u << 1, x, k);</pre>
           else modify(u \langle\langle 1 | 1, x, k\rangle\rangle;
65
66
       }
67
       //区间查询
68
        int query(int u,int l,int r) {
           if (tr[u].1 >= 1 && tr[u].r <= r) return tr[u].sum;</pre>
69
70
           pushdown(u);
71
           int sum = 0;
```

SMU 第5页

```
72
           int mid = tr[u].l + tr[u].r >> 1;
73
           if (1 <= mid)sum += query(u << 1, 1, r);</pre>
           if (r > mid) sum += query(u << 1 | 1, 1, r);
74
75
           return sum;
76
       }
77
       //单点查询
78
       int query(int u,int x) {
79
           if (tr[u].1 == tr[u].r)return tr[u].sum;
80
           pushdown(u);
           int mid = tr[u].l + tr[u].r >> 1;
81
82
           if (x <= mid)return query(u << 1, x);</pre>
83
           else return query(u << 1 | 1, x);</pre>
84
       }
85
   };
```

1.5 ST 表

```
template<class T>
 2
    struct ST {
 3
       Tn;
 4
       vector<vector<T>> f, g;
 5
       vector<int> lg2;
 6
       ST(const vector<T> &a) {
 7
           n = (int)a.size();
 8
           lg2.resize(n + 1);
 9
           lg2[0] = -1;
10
           for(int i = 1; i <= n; i ++) {</pre>
              lg2[i] = lg2[i >> 1] + 1;
11
12
           }
13
           f.resize(n + 1, vector<T>(lg2[n] + 1));
14
           g.resize(n + 1, vector<T>(lg2[n] + 1));
15
           for (int i = 1; i <= n; i++) {</pre>
16
              f[i][0] = a[i];
17
              g[i][0] = a[i];
18
           }
19
           for (int j = 1; (1 << j) <= n; j++) {</pre>
20
              for (int i = 1; i + (1 << j) - 1 <= n; i++) {
21
                  f[i][j] = max(f[i][j - 1], f[i + (1 << (j - 1))][j - 1]);
22
                  g[i][j] = min(g[i][j - 1], g[i + (1 << (j - 1))][j - 1]);
23
              }
24
           }
25
       }
26
27
       T query_max(int 1, int r) {
28
           int k = lg2[r - l + 1];
           return max(f[1][k], f[r - (1 << k) + 1][k]);</pre>
29
30
       }
31
32
       T query_min(int 1, int r) {
33
           int k = \lg 2[r - l + 1];
           return min(g[1][k], g[r - (1 << k) + 1][k]);
34
35
       }
```

SMU 第6页

36 };

1.6 二维树状数组

```
1
    template <class T>
 2
    struct BIT_2D {
 3
       vector<vector<T>> tr;
 4
       int n, m;
 5
       BIT_2D(int N, int M) {
 6
 7
           n = , m = M;
 8
           tr.resize(n + 1, vector<T>(m + 1));
 9
       }
10
11
       int lowbit(int x) { return x & (-x); }
12
13
       void add(int x, int y, T k) {
14
           for (int i = x; i <= n; i += lowbit(i))</pre>
              for (int j = y; j <= m; j += lowbit(j))</pre>
15
16
                 tr[i][j] += k;
17
       }
18
19
       T query(int x, int y) {
20
           T res = 0;
21
           for (int i = x; i; i -= lowbit(i))
22
              for (int j = y; j; j -= lowbit(j))
23
                 res += tr[i][j];
24
           return res;
25
       }
26
       T query(int x1, int y1, int x2, int y2) {
27
28
           return query(x2, y2) - query(x2, y1-1) - query(x1-1, y2) + query(x1-1, y1-1);
29
       }
30
    };
```

1.7 并查集

```
struct DSU {
1
 2
       vector<int> fa;
 3
       DSU(int n) {
 4
 5
           fa.resize(n + 1);
 6
           for (int i = 1; i <= n; i ++) fa[i] = i;</pre>
 7
       }
 8
 9
       int find(int x) {
           if (fa[x] != x) fa[x] = find(fa[x]);
10
           return fa[x];
11
12
       }
13
14
       bool same(int x, int y) {
```

SMU 第7页

```
15
           int px = find(x);
16
           int py = find(y);
17
           return x == y;
18
       }
19
20
       void merge(int x, int y) {
21
           int px = find(x);
22
           int py = find(y);
           if (px != py) {
23
24
              fa[px] = py;
25
           }
26
       }
27
    };
```

2 字符串

3 杂项

3.1 快读快写

```
inline int read() {
 2
       int x = 0;
 3
       char ch = getchar();
       while (ch < '0' || ch > '9') ch = getchar();
 5
       while (ch >= '0' && ch <= '9')x = (x << 3) + (x << 1) + ch - '0', ch = getchar()
           ;
 6
       return x;
 7
 8
    inline void write(int x) {
       if (x < 0) putchar('-'), x = -x;
9
10
       if (x > 9) write(x / 10);
11
       putchar(x % 10 + '0');
12
   }
```

3.2 取模

```
1
    template<const int T>
 2
    struct ModInt {
 3
       const static int mod = T;
       int x;
 5
       ModInt(int x = 0) : x(x \% mod) \{\}
       ModInt(long long x) : x(int(x % mod)) {}
 6
 7
       int val() {
 8
           return x;
 9
10
       ModInt operator + (const ModInt &a) const {
           int x0 = x + a.x;
11
12
           return ModInt(x0 < mod ? x0 : x0 - mod);</pre>
13
       }
       ModInt operator - (const ModInt &a) const {
```

SMU 第8页

```
15
           int x0 = x - a.x;
16
           return ModInt(x0 < 0 ? x0 + mod : x0);
17
18
       ModInt operator * (const ModInt &a) const {
19
           return ModInt(1LL * x * a.x % mod);
20
       ModInt operator / (const ModInt &a) const {
21
22
          return *this * a.inv();
23
       }
       bool operator == (const ModInt &a) const {
24
25
          return x == a.x;
26
27
       bool operator != (const ModInt &a) const {
28
          return x != a.x;
29
30
       void operator += (const ModInt &a) {
31
          x += a.x;
32
           if (x >= mod) x -= mod;
33
34
       void operator -= (const ModInt &a) {
35
           x -= a.x;
36
           if (x < 0) x += mod;
37
38
       void operator *= (const ModInt &a) {
           x = 1LL * x * a.x % mod;
39
40
       void operator /= (const ModInt &a) {
41
42
          *this = *this / a;
43
       friend ModInt operator + (int y, const ModInt &a) {
44
45
           int x0 = y + a.x;
46
           return ModInt(x0 < mod ? x0 : x0 - mod);</pre>
47
48
       friend ModInt operator - (int y, const ModInt &a) {
49
           int x0 = y - a.x;
50
          return ModInt(x0 < 0 ? x0 + mod : x0);
51
52
       friend ModInt operator * (int y, const ModInt &a) {
53
           return ModInt(1LL * y * a.x % mod);
54
55
       friend ModInt operator / (int y, const ModInt &a) {
56
           return ModInt(y) / a;
57
       }
       friend ostream &operator<<(ostream &os, const ModInt &a) {</pre>
58
59
          return os << a.x;</pre>
60
       friend istream &operator>>(istream &is, ModInt &t) {
61
          return is >> t.x;
62
63
64
65
       ModInt pow(int64_t n) const {
66
          ModInt res(1), mul(x);
67
          while(n) {
```

SMU 第9页

```
if (n & 1) res *= mul;
68
69
              mul *= mul;
70
              n >>= 1;
71
           }
72
           return res;
       }
73
74
75
       ModInt inv() const {
           int a = x, b = mod, u = 1, v = 0;
76
77
           while (b) {
78
              int t = a / b;
              a -= t * b;
79
80
              swap(a, b);
81
              u -= t * v;
              swap(u, v);
82
83
           }
84
           if (u < 0) u += mod;
85
           return u;
86
       }
87
88
    };
89
    using mint = ModInt<998244353>;
```

4 数学

4.1 组合数 (递推)

```
1
    int C[N][N];
 2
 3
    void comb(){
 4
        for(int i = 0; i < N; i++) {</pre>
 5
           for(int j = 0; j <= i; j++) {</pre>
 6
               if(!j) {
 7
                   C[i][j] = 1;
 8
                   continue;
 9
               }
10
               C[i][j] = C[i - 1][j] + C[i - 1][j - 1];
               C[i][j] %= mod;
11
12
           }
13
        }
14
    }
```

4.2 组合数(逆元)

```
1 struct comb2{
2  vector<int> fac,inf;
3
4  comb2(int n) {
5   fac.resize(n + 1);
6   inf.resize(n + 1);
7  fac[0] = inf[0] = 1;
```

第 10 页

```
8
          for(int i = 1; i <= n; i++){</pre>
9
              fac[i] = fac[i - 1] * i % mod;
              inf[i] = inf[i - 1] * ksm(i, mod - 2) % mod;
10
11
          }
12
13
14
       int ksm(int a, int b){ // 快速幂
15
          int res = 1;
16
          while(b){
17
              if(b & 1) res = res * a % mod;
18
              a = a * a % mod;
19
              b >>= 1;
20
          }
21
          return res;
22
23
24
       int query(int n, int m) {
25
          int ans;
26
          ans = fac[n] * inf[n - m] % mod * inf[m] % mod;
27
          return ans;
28
       }
29
    };
```