



南京工業大學
NANJING TECH
UNIVERSITY

ICPC Template Manual



作者: 贺梦杰

June 13, 2019

Contents

1	基础	2
2	搜索	3
3	动态规划	4
4	字符串	5
4.1	字符串测试	6
5	数据结构	7
5.1	线段树	8
5.1.1	基础操作	8
5.1.2	单点更新	8
5.1.3	区间更新	9
5.1.4	区间查询	10
5.2	树状数组	11
5.2.1	单点修改, 区间查询	11
5.2.2	区间修改, 单点查询	11
5.2.3	区间修改, 区间查询	12
5.3	二维树状数组	12
5.3.1	单点修改, 区间查询	12
5.3.2	区间修改, 区间查询	13
6	图论	16
7	数学	17
8	计算几何	18
9	其他	19

Chapter 1

基础

Chapter 2

搜索

Chapter 3

动态规划

Chapter 4

字符串

4.1 字符串测试

字符串测试内容

Chapter 5

数据结构

5.1 线段树

5.1.1 基础操作

```
1  const int N = 1e5 + 10;
2  #define ls(a) (a << 1)
3  #define rs(a) (a << 1 | 1)
4  struct node
5  {
6      int val;
7      int lazy;
8  };
9  node tree[N << 2];
10 int a[N];
11 void PushUp(int rt)
12 {
13     tree[rt].val = tree[ls(rt)].val + tree[rs(rt)];
14 }
15 void PushDown(int ls, int rs, int rt)
16 {
17     tree[ls(rt)].val += ls * tree[rt].lazy;
18     tree[rs(rt)].val += rs * tree[rt].lazy;
19     tree[ls(rt)].lazy += tree[rt].lazy;
20     tree[rs(rt)].lazy += tree[rt].lazy;
21     tree[rt].lazy = 0;
22 }
23 void Build(int left, int right, int rt)
24 {
25     if (left == right)
26     {
27         tree[rt].val = a[left];
28         return;
29     }
30     int mid = (left + right) >> 1;
31     Build(left, mid, ls(rt));
32     Build(mid + 1, right, rs(rt));
33     PushUp(rt);
34     //向上更新
35 }
```

5.1.2 单点更新

```
1 void Update(int left, int right, int rt, int pos, int val)
2 {
3     if (left == right && left == pos)
4     {
5         tree[rt].val += val;
6         return;
7     }
8     int mid = (left + right) >> 1;
9     if (tree[rt].lazy)
10    {
11        PushDown(mid - left + 1, right - mid, rt);
12    }
13    if (mid >= pos)
14        Update(left, mid, ls(rt), pos, val);
15    else if (pos > mid)
16        Update(mid + 1, right, rs(rt), pos, val);
17    PushUp(rt);
18 }
```

例题: <https://www.luogu.org/problemnew/show/P3372>

5.1.3 区间更新

```
1 void Update(int left, int right, int rt, int s, int t, int val)
2 {
3     if (left >= s && right <= t)
4     {
5         tree[rt].val += (right - left + 1) * val;
6         tree[rt].lazy += val;
7         return;
8     }
9     int mid = (left + right) >> 1;
10    if (tree[rt].lazy)
11    {
12        PushDown(mid - left + 1, right - mid, rt);
13    }
14    if (mid < s)
15        Update(mid + 1, right, rs(rt), s, t, val);
16    else if (mid >= t)
17        Update(left, mid, ls(rt), s, t, val);
18    else
19    {
20        Update(left, mid, ls(rt), s, t, val);
```

```
21         Update(mid + 1, right, rs(rt), s, t, val);
22     }
23     PushUp(rt);
24 }
```

5.1.4 区间查询

```
1 void Query(int left, int right, int s, int t, int rt)
2 {
3     if (left >= s && right <= t)
4     {
5         return tree[rt].val;
6     }
7     int mid = (left + right) >> 1;
8     if (tree[rt].lazy)
9         PushDown(mid - left + 1, right - mid, rt);
10    long long sum = 0;
11    if (mid < s)
12        sum += Query(mid + 1, right, rs(rt), s, t, val);
13    else if (mid >= t)
14        sum += Query(left, mid, ls(rt), s, t, val);
15    else
16    {
17        sum += Query(left, mid, ls(rt), s, t, val);
18        sum += Query(mid + 1, right, rs(rt), s, t, val);
19    }
20    return sum;
21 }
```

例题: <https://www.luogu.org/problemnew/show/P3373>

5.2 树状数组

推荐阅读: <https://www.cnblogs.com/RabbitHu/p/BIT.html>

5.2.1 单点修改, 区间查询

```
1 #define N 1000100
2 long long c[N];
3 int n,q;
4 int lowbit(int x)
5 {
6     return x&(-x);
7 }
8 void change(int x,int v)
9 {
10     while(x<=n)
11     {
12         c[x]+=v;
13         x+=lowbit(x);
14     }
15 }
16 long long getsum(int x)
17 {
18     long long ans=0;
19     while(x>=1)
20     {
21         ans+=c[x];
22         x-=lowbit(x);
23     }
24     return ans;
25 }
```

例题: <https://loj.ac/problem/130>

5.2.2 区间修改, 单点查询

引入差分数组来解决树状数组的区间更新

```
1 //初始化
2 change(i,cur-pre);
3 //区间修改
4 change(l,x);
5 change(r+1,-x);
6 //单点查询
```

7 getsum(x)

例题: <https://loj.ac/problem/131>

5.2.3 区间修改, 区间查询

```

1 //初始化
2 change(c1,i,cur-pre);
3 change(c2,i,i*(cur-pre));
4 //为什么这么写? 你需要写一下前缀和的表达式
5 //区间修改
6 change(c1,l,x);
7 change(c2,l,l*x);
8 change(c1,r+1,-x);
9 change(c2,r+1,-(r+1)*x);
10 //区间查询
11 temp1=l*getsum(c1,l-1)-getsum(c2,l-1);
12 temp2=(r+1)*getsum(c1,r)-getsum(c2,r);
13 ans=temp2-temp1

```

例题: <https://loj.ac/problem/132>

5.3 二维树状数组

5.3.1 单点修改, 区间查询

```

1 #define N 5050
2 long long tree[N][N];
3 long long n,m;
4 long long lowbit(long long x)
5 {
6     return x&(-x);
7 }
8 void change(long long x,long long y,long long val)
9 {
10     long long init_y=y;
11     //这里注意n,m的限制
12     while(x<=n)
13     {
14         y=init_y;
15         while(y<=m)
16         {
17             tree[x][y]+=val;

```

```

18         y+=lowbit(y);
19     }
20     x+=lowbit(x);
21 }
22 }
23 long long getsum(long long x,long long y)
24 {
25     long long ans=0;
26     long long init_y=y;
27     while(x>=1)
28     {
29         y=init_y;
30         while(y>=1)
31         {
32             ans+=tree[x][y];
33             y-=lowbit(y);
34         }
35         x-=lowbit(x);
36     }
37     //这里画图理解
38     return ans;
39 }
40 //初始化
41 change(x,y,k);
42 //二维前缀和
43 ans = getsum(c,d)+getsum(a-1,b-1)-getsum(a-1,d)-getsum(c,b-1);

```

例题: <https://loj.ac/problem/133>

5.3.2 区间修改, 区间查询

```

1  #define N 2050
2  long long t1[N][N];
3  long long t2[N][N];
4  long long t3[N][N];
5  long long t4[N][N];
6  long long n,m;
7  long long lowbit(long long x)
8  {
9      return x&(-x);
10 }
11 long long getsum(long long x,long long y)
12 {

```

```

13     long long ans=0;
14     long long init_y=y;
15     long long init_x=x;
16     while(x>=1)
17     {
18         y=init_y;
19         while(y>=1)
20         {
21             ans+=(init_x+1)*(init_y+1)*t1[x][y];
22             ans-=(init_y+1)*t2[x][y];
23             ans-=(init_x+1)*t3[x][y];
24             ans+=t4[x][y];
25             y-=lowbit(y);
26         }
27         x-=lowbit(x);
28     }
29     return ans;
30 }
31 void change(long long x,long long y,long long val)
32 {
33     long long init_x=x;
34     long long init_y=y;
35     while(x<=n)
36     {
37         y=init_y;
38         while(y<=m)
39         {
40             t1[x][y]+=val;
41             t2[x][y]+=init_x*val;
42             t3[x][y]+=init_y*val;
43             t4[x][y]+=init_x*init_y*val;
44             y+=lowbit(y);
45         }
46         x+=lowbit(x);
47     }
48 }
49 //区间修改
50 change(c+1,d+1,x);
51 change(a,b,x);
52 change(a,d+1,-x);
53 change(c+1,b,-x);
54 //区间查询
55 ans=getsum(c,d)+getsum(a-1,b-1)-getsum(c,b-1)-getsum(a-1,d);

```

例题: <https://loj.ac/problem/135>

Chapter 6

图论

Chapter 7

数学

Chapter 8

计算几何

Chapter 9

其他