

退役的Shawn Zhou的博客

Away From OI.

博客园

管理

随笔 - 114 文章 - 0 评论 - 2

公告

本博客所有文章允许您进行规范的转载和引用，您的转载和引用是支持我写下去的最大动力。但博主码字不易，还请您注明出处及作者。

昵称：ShawnZhou_Aether

园龄：3年3个月

粉丝：53

关注：12

+加关注

<	2020年11月						>
日	一	二	三	四	五	六	
1	2	3	4	5	6	7	
8	9	10	11	12	13	14	
15	16	17	18	19	20	21	
22	23	24	25	26	27	28	
29	30	1	2	3	4	5	
6	7	8	9	10	11	12	

我的标签

题解(54)

杂记(21)

[转载]完全版线段树 by notonlysuccess大牛

原文出处：<http://www.notonlysuccess.com/>

(好像现在这个博客已经挂掉了，在网上找到的全部都是转载)

今天在清北学堂听课，听到了一些很令人吃惊的消息。至于这消息具体是啥，等我晚上再说
23333

我大概的看了一下所讲内容及实例，果不愧是大牛级别的人，讲的很详细，所以决定转载过来。

对原文作者致以崇高的敬意。

以下是**转载**内容。

【完全版】线段树

很早前写的那篇线段树专辑至今一直是本博客阅读量最高的文章，当年觉得挺自豪的，还去pku看了看那篇文章，风格实在是太丑

夏令营(17)
模拟(15)
图(11)
字符串(10)
动态规划(10)
清北学堂(8)
数学相关(8)
贪心(8)
更多

随笔档案

2018年6月(1)
2017年11月(13)
2017年10月(46)
2017年9月(32)
2017年8月(22)

友情链接

Virtualman的博客
kench的博客
lewis | 青岛一中蒟蒻
BK-Edwina的博客
Okami 's Blog
rqy的博客
syp的博客
Kamigen
杰克部落

最新评论

- 1. Re:新博客即将启用
"一个人一辈子能把一件事情做好，就堪称完美。一事精致，便已动人。从一而终，便是深邃"【暖心良言】"少而好学，如日出之阳；壮而好学，如日中之光；志而好学，如炳烛之光。"【暖心良言】 ...
--前方一片光明
- 2. Re:夏令营讲课内容整理 Day 7.
加油奥力给
--辛绍航
- 3. Re:树状数组 学习笔记

了,很多线段树的初学者可能就是看着这篇文章来练习的,如果不小心被我培养出了这么糟糕的风格,实在是过意不去,正好过几天又要给集训队讲解线段树,所以决定把这些题目重新写一遍,顺便把近年我接触到的一些新题更新上去~;并且学习了splay等更高级的数据结构后对线段树的体会有更深了一层,线段树的写法也就比以前飘逸,简洁且方便多了.

在代码前先介绍一些我的线段树风格:

- o maxn是题目给的最大区间,而节点数要开4倍,确切的来说节点数要开大于maxn的最小2x的两倍
- o lson和rson分辨表示结点的左儿子和右儿子,由于每次传参数的时候都固定是这几个变量,所以可以用预定于比较方便的表示
- o 以前的写法是另外开两个数组记录每个结点所表示的区间,其实这个区间不必保存,一边算一边传下去就行,只需要写函数的时候多两个参数,结合lson和rson的预定义可以很方便
- o PushUP(int rt)是把当前结点的信息更新到父结点
- o PushDown(int rt)是把当前结点的信息更新给儿子结点
- o rt表示当前子树的根(root),也就是当前所在的结点

整理这些题目后我觉得线段树的题目整体上可以分成以下四个部分:

单点更新:最最基础的线段树,只更新叶子节点,然后把信息用PushUP(int r)这个函数更新上来

hdu1166 敌兵布阵

题意:O(-1)



膜

--萌面人

4. Re:结束吧，为这不圆满的故事划
上一个残缺的句号
加油，您会激励我们的

--cyter

5. Re:[转载]完全版线段树 by
notonlysuccess大牛
这个太全了，今晚的线段树经典题
自闭，没看过，抽时间好好学习
一下。

--一届书生

阅读排行榜

1. 前缀和与差分(7636)
2. 结束吧，为这不圆满的故事划上
一个残缺的句号(2415)
3. NOIP2016提高组初赛（C++语
言）试题 个人的胡乱分析 Part
2.(1258)
4. NOIP 2017 day 1 游记(968)
5. 济南清北学堂游记 Day 1.(807)
6. 平衡树 学习笔记(750)
7. OI黑科技：读入优化(708)
8. 新博客即将启用(529)
9. NOIP 2017 Day 0. 游记(511)
10. 自用线段树模板(498)

评论排行榜

1. 结束吧，为这不圆满的故事划上
一个残缺的句号(5)
2. 关于博主 | 联系博主(4)
3. 很想说点什么(3)
4. 做图与树做到吐的一天(2)
5. 蚯蚓(2)

推荐排行榜

1. 前缀和与差分(25)
2. 结束吧，为这不圆满的故事划上
一个残缺的句号(21)
3. 关于博主 | 联系博主(5)
4. 新博客即将启用(3)



```
1 #include <cstdio>
2
3
4
5 #define lson l , m , rt << 1
6
7 #define rson m + 1 , r , rt << 1 | 1
8
9 const int maxn = 55555;
10
11 int sum[maxn<<2];
12
13 void PushUP(int rt) {
14
15     sum[rt] = sum[rt<<1] + sum[rt<<1|1];
16
17 }
18
19 void build(int l,int r,int rt) {
20
21     if (l == r) {
22
23         scanf("%d",&sum[rt]);
24
25         return ;
26
27     }
28
29     int m = (l + r) >> 1;
30
31     build(lson);
32
33     build(rson);
34
35     PushUP(rt);
36
37 }
38
39 void update(int v,int l,int r,int rt) {
40
41     if (l == r) {
42         sum[rt] = v;
43     }
44     else {
45         int m = (l + r) >> 1;
46         if (l <= m) update(v,l,m,rt<<1);
47         else update(v,m+1,r,rt<<1|1);
48         PushUP(rt);
49     }
50 }
```

5. 济南清北学堂游记 Day 0.(2)

```
43         sum[rt] += add;
44
45         return ;
46
47     }
48
49     int m = (l + r) >> 1;
50
51     if (p <= m) update(p , add , lson);
52
53     else update(p , add , rson);
54
55     PushUP(rt);
56
57 }
58
59 int query(int L,int R,int l,int r,int rt)
60
61     if (L <= l && r <= R) {
62
63         return sum[rt];
64
65     }
66
67     int m = (l + r) >> 1;
68
69     int ret = 0;
70
71     if (L <= m) ret += query(L , R , l , m , sum[lson]);
72
73     if (R > m) ret += query(L , R , m + 1 , r , sum[rson]);
74
75     return ret;
76
77 }
78
79 int main() {
80
81     int T , n;
82
83
84     0      0
85
86     cas <= T ; cas
```

```
86
87         printf("Case %d:\n", cas);
88
89         scanf("%d", &n);
90
91         build(1 , n , 1);
92
93         char op[10];
94
95         while (scanf("%s", op)) {
96
97             if (op[0] == 'E') k
98
99             int a , b;
100
101             scanf("%d%d", &a, &b)
102
103             if (op[0] == 'Q') g
104
105             else if (op[0] == '
106
107             else update(a , b ,
108
109         }
110
111     }
112
113     return 0;
114
115 }
```



hdu1754 I Hate It

题意:O(-1)

思路:O(-1)

线段树功能:

0

0

ry:区间最值



```
1 #include <cstdio>
2
3 #include <algorithm>
4
5 using namespace std;
6
7
8
9 #define lson l , m , rt << 1
10
11 #define rson m + 1 , r , rt << 1 | 1
12
13 const int maxn = 222222;
14
15 int MAX[maxn<<2];
16
17 void PushUP(int rt) {
18
19     MAX[rt] = max(MAX[rt<<1] , MAX[rt<<1|1]);
20
21 }
22
23 void build(int l,int r,int rt) {
24
25     if (l == r) {
26
27         scanf("%d",&MAX[rt]);
28
29         return ;
30
31     }
32
33     int m = (l + r) >> 1;
34
35     build(lson);
36
37     build(rson);
38
39
40     0      0
41 }
42
```

```
43 void update(int p,int sc,int l,int r,int
44
45     if (l == r) {
46
47         MAX[rt] = sc;
48
49         return ;
50
51     }
52
53     int m = (l + r) >> 1;
54
55     if (p <= m) update(p , sc , lson)
56
57     else update(p , sc , rson);
58
59     PushUP(rt);
60
61 }
62
63 int query(int L,int R,int l,int r,int rt
64
65     if (L <= l && r <= R) {
66
67         return MAX[rt];
68
69     }
70
71     int m = (l + r) >> 1;
72
73     int ret = 0;
74
75     if (L <= m) ret = max(ret , query
76
77     if (R > m) ret = max(ret , query
78
79     return ret;
80
81 }
82
83 int ma
84     0      0
85
```

```

86
87     while (~scanf("%d%d",&n,&m)) {
88
89         build(1 , n , 1);
90
91         while (m --) {
92
93             char op[2];
94
95             int a , b;
96
97             scanf("%s%d%d",op,&a,&b);
98
99             if (op[0] == 'Q') {
100
101                 else update(a , b , 1);
102
103             }
104
105         }
106
107     return 0;
108
109 }

```

hdu1394 Minimum Inversion Number

题意:求Inversion后的最小逆序数

思路:用 $O(n\log n)$ 复杂度求出最初逆序数后,就可以用 $O(1)$ 的复杂度分别递推出其他解

线段树功能:update:单点增减 query:区间求和

```

1 #include <cstdio>
2
3 #include
4
5 using

```



```
6
7
8
9 #define lson l , m , rt << 1
10
11 #define rson m + 1 , r , rt << 1 | 1
12
13 const int maxn = 5555;
14
15 int sum[maxn<<2];
16
17 void PushUP(int rt) {
18
19     sum[rt] = sum[rt<<1] + sum[rt<<1|1];
20
21 }
22
23 void build(int l,int r,int rt) {
24
25     sum[rt] = 0;
26
27     if (l == r) return ;
28
29     int m = (l + r) >> 1;
30
31     build(lson);
32
33     build(rson);
34
35 }
36
37 void update(int p,int l,int r,int rt) {
38
39     if (l == r) {
40
41         sum[rt] ++;
42
43         return ;
44
45     }
46
47     0      0      1;
48
```

```
49         if (p <= m) update(p , lson);
50
51         else update(p , rson);
52
53         PushUP(rt);
54
55     }
56
57     int query(int L,int R,int l,int r,int rt)
58
59         if (L <= l && r <= R) {
60
61             return sum[rt];
62
63         }
64
65         int m = (l + r) >> 1;
66
67         int ret = 0;
68
69         if (L <= m) ret += query(L , R ,
70
71         if (R > m) ret += query(L , R , 1
72
73         return ret;
74
75     }
76
77     int x[maxn];
78
79     int main() {
80
81         int n;
82
83         while (~scanf("%d",&n)) {
84
85             build(0 , n - 1 , 1);
86
87             int sum = 0;
88
89             for (int i = 0 ; i < n ; i++)
90                 scanf("%d",&x[i]);
91
```

```

92
93         sum += query(x[i] ,
94
95         update(x[i] , 0 , r
96
97     }
98
99     int ret = sum;
100
101     for (int i = 0 ; i < n ; i
102
103         sum += n - x[i] - x
104
105         ret = min(ret , sum
106
107     }
108
109     printf("%d\n",ret);
110
111 }
112
113     return 0;
114
115 }

```



hdu2795 Billboard

题意: $h*w$ 的木板,放进一些 $1*L$ 的物品,求每次放空间能容纳且最上边的位子

思路:每次找到最大值的位子,然后减去 L

线段树功能:query:区间求最大值的位子(直接把update的操作在query里做了)



```

1 #include
2

```

0

0

```
3 #include <algorithm>
4
5 using namespace std;
6
7
8
9 #define lson l , m , rt << 1
10
11 #define rson m + 1 , r , rt << 1 | 1
12
13 const int maxn = 222222;
14
15 int h , w , n;
16
17 int MAX[maxn<<2];
18
19 void PushUP(int rt) {
20
21     MAX[rt] = max(MAX[rt<<1] , MAX[rt<
22
23 }
24
25 void build(int l,int r,int rt) {
26
27     MAX[rt] = w;
28
29     if (l == r) return ;
30
31     int m = (l + r) >> 1;
32
33     build(lson);
34
35     build(rson);
36
37 }
38
39 int query(int x,int l,int r,int rt) {
40
41     if (l == r) {
42
43         0 0;
44
45
```

```
46
47     }
48
49     int m = (l + r) >> 1;
50
51     int ret = (MAX[rt<<1] >= x) ? query(l, r, m, x) : query(l, r, m+1, x);
52
53     PushUP(rt);
54
55     return ret;
56
57 }
58
59 int main() {
60
61     while (~scanf("%d%d%d", &h, &w, &n)) {
62
63         if (h > n) h = n;
64
65         build(1, h, 1);
66
67         while (n --) {
68
69             int x;
70
71             scanf("%d", &x);
72
73             if (MAX[1] < x) puts("Yes");
74
75             else printf("%d\n", MAX[1]);
76
77         }
78
79     }
80
81     return 0;
82
83 }
84
```



0

0

练习:

poj2828 Buy Tickets

poj2886 Who Gets the Most Candies?

成段更新(通常这对初学者来说是一道坎),需要用到延迟标记(或者说懒惰标记),简单来说就是每次更新的时候不要更新到底,用延迟标记使得更新延迟到下次需要更新或询问到的时候

hdu1698 Just a Hook

题意: $O(-1)$

思路: $O(-1)$

线段树功能:update:成段替换 (由于只query一次总区间,所以可以直接输出1结点的信息)



```
1 #include <cstdio>
2
3 #include <algorithm>
4
5 using namespace std;
6
7
8
9 #define lson l , m , rt << 1
10
11 #define rson m + 1 , r , rt << 1 | 1
12
13 const int maxn = 111111;
14
15 int h , w , n;
16
17 int col[maxn<<2];
18
19 int su
20
21 void E
22
```

0

0

```
23         sum[rt] = sum[rt<<1] + sum[rt<<1|1]
24
25     }
26
27     void PushDown(int rt,int m) {
28
29         if (col[rt]) {
30
31             col[rt<<1] = col[rt<<1|1]
32
33             sum[rt<<1] = (m - (m >> 1)
34
35             sum[rt<<1|1] = (m >> 1) *
36
37             col[rt] = 0;
38
39         }
40
41     }
42
43     void build(int l,int r,int rt) {
44
45         col[rt] = 0;
46
47         sum[rt] = 1;
48
49         if (l == r) return ;
50
51         int m = (l + r) >> 1;
52
53         build(lson);
54
55         build(rson);
56
57         PushUp(rt);
58
59     }
60
61     void update(int L,int R,int c,int l,int
62
63         0      0      R) {
64
65     ;
```

```
66
67         sum[rt] = c * (r - l + 1);
68
69         return ;
70
71     }
72
73     PushDown(rt , r - l + 1);
74
75     int m = (l + r) >> 1;
76
77     if (L <= m) update(L , R , c , ls);
78
79     if (R > m) update(L , R , c , rs);
80
81     PushUp(rt);
82
83 }
84
85 int main() {
86
87     int T , n , m;
88
89     scanf("%d",&T);
90
91     for (int cas = 1 ; cas <= T ; cas++)
92     {
93         scanf("%d%d",&n,&m);
94
95         build(1 , n , 1);
96
97         while (m --) {
98
99             int a , b , c;
100
101             scanf("%d%d%d",&a,&b,&c);
102
103             update(a , b , c , 1);
104
105         }
106
107         printf("Case %d: The total sum is %d\n",cas,sum[1]);
108     }
```



```

109     }
110
111     return 0;
112
113 }

```



poj3468 A Simple Problem with Integers

题意: $O(-1)$

思路: $O(-1)$

线段树功能:update:成段增减 query:区间求和



```

1  #include <cstdio>
2
3  #include <algorithm>
4
5  using namespace std;
6
7
8
9  #define lson l , m , rt << 1
10
11 #define rson m + 1 , r , rt << 1 | 1
12
13 #define LL long long
14
15 const int maxn = 111111;
16
17 LL add[maxn<<2];
18
19 LL sum[maxn<<2];
20
21 void PushUp(int rt) {
22
23     sum[rt] = sum[rt<<1] + sum[rt<<1
24
25 }
26
27 void Build(int l, int r, int rt) {

```

```
28
29     if (add[rt]) {
30
31         add[rt<<1] += add[rt];
32
33         add[rt<<1|1] += add[rt];
34
35         sum[rt<<1] += add[rt] * (r
36
37         sum[rt<<1|1] += add[rt] *
38
39         add[rt] = 0;
40
41     }
42
43 }
44
45 void build(int l,int r,int rt) {
46
47     add[rt] = 0;
48
49     if (l == r) {
50
51         scanf("%lld",&sum[rt]);
52
53         return ;
54
55     }
56
57     int m = (l + r) >> 1;
58
59     build(lson);
60
61     build(rson);
62
63     PushUp(rt);
64
65 }
66
67 void update(int L,int R,int c,int l,int
68
69     0      0      R) {
70
```

```
71         add[rt] += c;
72
73         sum[rt] += (LL)c * (r - l + 1);
74
75         return ;
76
77     }
78
79     PushDown(rt , r - l + 1);
80
81     int m = (l + r) >> 1;
82
83     if (L <= m) update(L , R , c , l , m , rt * 2);
84
85     if (m < R) update(L , R , c , m + 1 , r , rt * 2 + 1);
86
87     PushUp(rt);
88
89 }
90
91 LL query(int L,int R,int l,int r,int rt)
92 {
93     if (L <= l && r <= R) {
94
95         return sum[rt];
96
97     }
98
99     PushDown(rt , r - l + 1);
100
101     int m = (l + r) >> 1;
102
103     LL ret = 0;
104
105     if (L <= m) ret += query(L , R , l , m , rt * 2);
106
107     if (m < R) ret += query(L , R , m + 1 , r , rt * 2 + 1);
108
109     return ret;
110
111 }
112
113 int main() {
```

0	0
---	---

```
114
115     int N , Q;
116
117     scanf("%d%d",&N,&Q);
118
119     build(1 , N , 1);
120
121     while (Q --) {
122
123         char op[2];
124
125         int a , b , c;
126
127         scanf("%s",op);
128
129         if (op[0] == 'Q') {
130
131             scanf("%d%d",&a,&b);
132
133             printf("%lld\n",que
134
135         } else {
136
137             scanf("%d%d%d",&a,&
138
139             update(a , b , c ,
140
141         }
142
143     }
144
145     return 0;
146
147 }
```



poj2528 Mayor' s posters

题意:在墙上贴海报 海报可以互相覆盖,问最后可以看

0 0

思路:这题数据范

超内存,需要离散

化:

离散化简单的来说就是只取我们需要的值来用,比如说区间 $[1000,2000]$, $[1990,2012]$ 我们用不到 $[-\infty,999]$ $[1001,1989]$ $[1991,1999]$ $[2001,2011]$ $[2013,+\infty]$ 这些值,所以我只需要1000,1990,2000,2012就够了,将其分别映射到0,1,2,3,在于复杂度就大大的降下来了

所以离散化要保存所有需要用到的值,排序后,分别映射到 $1\sim n$,这样复杂度就会小很多很多

而这题的难点在于每个数字其实表示的是一个单位长度(并且一个点),这样普通的离散化会造成许多错误(包括我以前的代码,poj这题数据奇弱)

给出下面两个简单的例子应该能体现普通离散化的缺陷:

1-10 1-4 5-10

1-10 1-4 6-10

为了解决这种缺陷,我们可以在排序后的数组上加些处理,比如说 $[1,2,6,10]$

如果相邻数字间距大于1的话,在其中加上任意一个数字,比如加成 $[1,2,3,6,7,10]$,然后再做线段树就好了.

线段树功能:update:成段替换 query:简单hash



```
1 #include <stdio>
2
3 #include <cstring>
4
5 #include <algorithm>
6
7 using namespace std;
8
9 #define lson l , m , rt << 1
10
11 #define rson m + 1 , rt << 1 | 1
12
13
```

```
14
15 const int maxn = 11111;
16
17 bool hash[maxn];
18
19 int li[maxn] , ri[maxn];
20
21 int X[maxn*3];
22
23 int col[maxn<<4];
24
25 int cnt;
26
27
28
29 void PushDown(int rt) {
30
31     if (col[rt] != -1) {
32
33         col[rt<<1] = col[rt<<1|1]
34
35         col[rt] = -1;
36
37     }
38
39 }
40
41 void update(int L,int R,int c,int l,int
42
43     if (L <= l && r <= R) {
44
45         col[rt] = c;
46
47         return ;
48
49     }
50
51     PushDown(rt);
52
53     int m = (l + r) >> 1;
54
55     0      0      (L , R , c , ls
56
```

```
57         if (m < R) update(L , R , c , rsc
58
59     }
60
61 void query(int l,int r,int rt) {
62
63     if (col[rt] != -1) {
64
65         if (!hash[col[rt]]) cnt ++
66
67         hash[ col[rt] ] = true;
68
69         return ;
70
71     }
72
73     if (l == r) return ;
74
75     int m = (l + r) >> 1;
76
77     query(lson);
78
79     query(rson);
80
81 }
82
83 int Bin(int key,int n,int X[]) {
84
85     int l = 0 , r = n - 1;
86
87     while (l <= r) {
88
89         int m = (l + r) >> 1;
90
91         if (X[m] == key) return m;
92
93         if (X[m] < key) l = m + 1;
94
95         else r = m - 1;
96
97
98
99
```

0

0

```
100
101 }
102
103 int main() {
104
105     int T , n;
106
107     scanf("%d",&T);
108
109     while (T --) {
110
111         scanf("%d",&n);
112
113         int nn = 0;
114
115         for (int i = 0 ; i < n ; i++) {
116
117             scanf("%d%d",&li[i]&ri[i]);
118
119             X[nn++] = li[i];
120
121             X[nn++] = ri[i];
122
123         }
124
125         sort(X , X + nn);
126
127         int m = 1;
128
129         for (int i = 1 ; i < nn; i++) {
130
131             if (X[i] != X[i-1])
132                 m++;
133
134         }
135
136         for (int i = m - 1 ; i > 0 ; i--) {
137
138             if (X[i] != X[i-1])
139                 m++;
140
141             0 0 + m);
142
143         }
```



```

143         memset(col , -1 , sizeof(c
144
145         for (int i = 0 ; i < n ; i
146
147             int l = Bin(li[i] ,
148
149             int r = Bin(ri[i] ,
150
151             update(l , r , i ,
152
153         }
154
155         cnt = 0;
156
157         memset(hash , false , size
158
159         query(0 , m , 1);
160
161         printf("%d\n",cnt);
162
163     }
164
165     return 0;
166
167 }

```



poj3225 Help with Intervals

题意:区间操作,交,并,补等

思路:

我们一个一个操作来分析:(用0和1表示是否包含区间,-1表示该区间内既有包含又有不包含)

U:把区间[l,r]覆盖成1

I:把 $[-\infty, l)(r, \infty]$ 覆盖成0

D:把区间[l,r]覆盖成0

C:把 $[-\infty, l)(r, \infty]$ 覆盖成0 且[l,r]区间0/1互换

0 0

成段覆盖的操作很简单,比较特殊的就是区间0/1互换
这个操作,我们可以称之为异或操作

很明显我们可以知道这个性质:当一个区间被覆盖后,不管之前有没有异或标记都没有意义了

所以当有一个节点得到覆盖标记时把异或标记清空

而当一个节点得到异或标记的时候,先判断覆盖标记,如果是0或1,直接改变一下覆盖标记,不然的话改变异或标记

开区间闭区间只要数字乘以2就可以处理(偶数表示端点,奇数表示两端点间的区间)

线段树功能:update:成段替换,区间异或 query:简单hash



```
1 #include <cstdio>
2
3 #include <cstring>
4
5 #include <cctype>
6
7 #include <algorithm>
8
9 using namespace std;
10
11 #define lson l , m , rt << 1
12
13 #define rson m + 1 , r , rt << 1 | 1
14
15
16
17 const int maxn = 131072;
18
19 bool hash[maxn];
20
21 int co 0 0
22
```

```
23 int XOR[maxn<<2];
24
25 void FXOR(int rt) {
26
27     if (cover[rt] != -1) cover[rt] ^=
28
29     else XOR[rt] ^= 1;
30
31 }
32
33 void PushDown(int rt) {
34
35     if (cover[rt] != -1) {
36
37         cover[rt<<1] = cover[rt<<1]
38
39         XOR[rt<<1] = XOR[rt<<1]^1]
40
41         cover[rt] = -1;
42
43     }
44
45     if (XOR[rt]) {
46
47         FXOR(rt<<1);
48
49         FXOR(rt<<1^1);
50
51         XOR[rt] = 0;
52
53     }
54
55 }
56
57 void update(char op,int L,int R,int l,int r)
58
59     if (L <= l && r <= R) {
60
61         if (op == 'U') {
62
63             cover[rt] = 1;
64
65             XOR[rt] = 0;
```

```
66
67         } else if (op == 'D') {
68
69             cover[rt] = 0;
70
71             XOR[rt] = 0;
72
73         } else if (op == 'C' || op == 'I') {
74
75             FXOR(rt);
76
77         }
78
79         return ;
80
81     }
82
83     PushDown(rt);
84
85     int m = (l + r) >> 1;
86
87     if (L <= m) update(op , L , R , rt, l, r, m);
88
89     else if (op == 'I' || op == 'C')
90
91         XOR[rt<<1] = cover[rt<<1];
92
93     }
94
95     if (m < R) update(op , L , R , rt, m+1, r, m+1);
96
97     else if (op == 'I' || op == 'C')
98
99         XOR[rt<<1|1] = cover[rt<<1|1];
100
101     }
102
103 }
104
105 void query(int l,int r,int rt) {
106     int ans = 0;
107     if (l <= r) {
108         if (l == r) {
109             if (op == 'I')
110                 ans = XOR[rt];
111             else if (op == 'C')
112                 ans = cover[rt];
113             else if (op == 'D')
114                 ans = 0;
115         }
116         else {
117             query(l, r, rt<<1);
118             query(l, r, rt<<1|1);
119             if (op == 'I')
120                 ans = XOR[rt];
121             else if (op == 'C')
122                 ans = cover[rt];
123             else if (op == 'D')
124                 ans = 0;
125         }
126     }
127 }
```

```
109         for (int it = 1 ; it <= r
110
111             hash[it] = true;
112
113         }
114
115         return ;
116
117     } else if (cover[rt] == 0) return
118
119     if (l == r) return ;
120
121     PushDown(rt);
122
123     int m = (l + r) >> 1;
124
125     query(lson);
126
127     query(rson);
128
129 }
130
131 int main() {
132
133     cover[1] = XOR[1] = 0;
134
135     char op , l , r;
136
137     int a , b;
138
139     while ( ~scanf("%c %c%d,%d%c\n", &
140
141         a <= 1 , b <= 1;
142
143         if (l == '(') a ++;
144
145         if (r == ')') b --;
146
147         if (a > b) {
148
149             op == 'C' || op
150             0      0
151             cover[1] = 2
```

```
152
153         }
154
155         } else update(op , a , b ,
156
157     }
158
159     query(0 , maxn , 1);
160
161     bool flag = false;
162
163     int s = -1 , e;
164
165     for (int i = 0 ; i <= maxn ; i ++
166
167         if (hash[i]) {
168
169             if (s == -1) s = i;
170
171             e = i;
172
173         } else {
174
175             if (s != -1) {
176
177                 if (flag) pr
178
179                 flag = true;
180
181                 printf("%c%c
182
183                 s = -1;
184
185             }
186
187         }
188
189     }
190
191     if (!flag) printf("empty set");
192
193     0      0
194
```

```
195         return 0;
196
197     }
198
```



练习:

poj1436 Horizontally Visible Segments

poj2991 Crane

Another LCIS

Bracket Sequence

区间合并

这类题目会询问区间中满足条件的连续最长区间,
所以PushUp的时候需要对左右儿子的区间进行
合并

poj3667 Hotel

题意:1 a:询问是不是有连续长度为a的空房间,有的话
住进最左边

2 a b:将[a,a+b-1]的房间清空

思路:记录区间中最长的空房间

线段树操作:update:区间替换 query:询问满足条件的
最左断点



```
1 #include <stdio>
2
3 #inclu
4
```

0

0

```

5 #include <cctype>
6
7 #include <algorithm>
8
9 using namespace std;
10
11 #define lson l , m , rt << 1
12
13 #define rson m + 1 , r , rt << 1 | 1
14
15
16
17 const int maxn = 55555;
18
19 int lsum[maxn<<2] , rsum[maxn<<2] , msum
20
21 int cover[maxn<<2];
22
23
24
25 void PushDown(int rt,int m) {
26
27     if (cover[rt] != -1) {
28
29         cover[rt<<1] = cover[rt<<1];
30
31         msum[rt<<1] = lsum[rt<<1];
32
33         msum[rt<<1|1] = lsum[rt<<1];
34
35         cover[rt] = -1;
36
37     }
38
39 }
40
41 void PushUp(int rt,int m) {
42
43     lsum[rt] = lsum[rt<<1];
44
45     msum[rt] = lsum[rt<<1|1];
46
47     rsum[rt] = (m >> 1) lsum[rt<>1];

```



```
48
49     if (rsum[rt] == (m >> 1)) rsum[rt] = msum[rt];
50
51     msum[rt] = max(lsum[rt<<1|1] + rsum[rt], rsum[rt]);
52
53 }
54
55 void build(int l,int r,int rt) {
56
57     msum[rt] = lsum[rt] = rsum[rt] = 0;
58
59     cover[rt] = -1;
60
61     if (l == r) return ;
62
63     int m = (l + r) >> 1;
64
65     build(lson);
66
67     build(rson);
68
69 }
70
71 void update(int L,int R,int c,int l,int r,int rt) {
72
73     if (L <= l && r <= R) {
74
75         msum[rt] = lsum[rt] = rsum[rt] = c;
76
77         cover[rt] = c;
78
79         return ;
80
81     }
82
83     PushDown(rt , r - l + 1);
84
85     int m = (l + r) >> 1;
86
87     if (L <= m) update(L , R , c , l , m , rt<<1);
88
89     if (R > m) update(L , R , c , m+1 , r , rt<<1|1);
90 }
```

```
91     PushUp(rt , r - 1 + 1);
92
93 }
94
95 int query(int w,int l,int r,int rt) {
96
97     if (l == r) return l;
98
99     PushDown(rt , r - 1 + 1);
100
101     int m = (l + r) >> 1;
102
103     if (msum[rt<<1] >= w) return query(w,l,m,rt<<1);
104
105     else if (rsum[rt<<1] + lsum[rt<<1] < w) return query(w,m,r,rt<<1);
106
107     return query(w , rson);
108
109 }
110
111 int main() {
112
113     int n , m;
114
115     scanf("%d%d",&n,&m);
116
117     build(1 , n , 1);
118
119     while (m --) {
120
121         int op , a , b;
122
123         scanf("%d",&op);
124
125         if (op == 1) {
126
127             scanf("%d",&a);
128
129             if (msum[1] < a) pu
130
131             {
132                 0      0
133
134             int p = que
```

```
134
135                                     printf("%d\n", r);
136
137                                     update(p, l, r, v);
138
139                                     }
140
141                                     } else {
142
143                                     scanf("%d%d", &a, &b);
144
145                                     update(a, a + b - 1, v);
146
147                                     }
148
149                                     }
150
151                                     return 0;
152
153 }
```



练习:

hdu3308 LCIS

hdu3397 Sequence operation

hdu2871 Memory Control

hdu1540 Tunnel Warfare

CF46-D Parking Lot

扫描线

这类题目需要将一些操作排序,然后从左到右用一根扫描线(当

0

0

过去

最典型的就是一道等题

hdu1542 Atlantis

题意:矩形面积并

思路:浮点数先要离散化;然后把矩形分成两条边,上边和下边,对横轴建树,然后从下到上扫描上去,用cnt表示该区间下边比上边多几个

线段树操作:update:区间增减 query:直接取根节点的值



```
1 #include <cstdio>
2
3 #include <cstring>
4
5 #include <cctype>
6
7 #include <algorithm>
8
9 using namespace std;
10
11 #define lson l , m , rt << 1
12
13 #define rson m + 1 , r , rt << 1 | 1
14
15
16
17 const int maxn = 2222;
18
19 int cnt[maxn << 2];
20
21 double sum[maxn << 2];
22
23 double X[maxn];
24
25 struct Seg {
26
27     0      0
28
29 }
```

```
30
31     Seg() {}
32
33     Seg(double a, double b, double c, int l, int r) {
34
35         bool operator < (const Seg &cmp) {
36
37             return h < cmp.h;
38
39         }
40
41     } ss[maxn];
42
43 void PushUp(int rt, int l, int r) {
44
45     if (cnt[rt]) sum[rt] = X[r+1] - X[l];
46
47     else if (l == r) sum[rt] = 0;
48
49     else sum[rt] = sum[rt<<1] + sum[rt>>1];
50
51 }
52
53 void update(int L, int R, int c, int l, int r) {
54
55     if (L <= l && r <= R) {
56
57         cnt[rt] += c;
58
59         PushUp(rt, l, r);
60
61         return ;
62
63     }
64
65     int m = (l + r) >> 1;
66
67     if (L <= m) update(L, R, c, l, m);
68
69     if (m < R) update(L, R, c, m+1, r);
70
71     PushUp(rt, l, r);
72
```

```
73 }
74
75 int Bin(double key,int n,double X[]) {
76
77     int l = 0 , r = n - 1;
78
79     while (l <= r) {
80
81         int m = (l + r) >> 1;
82
83         if (X[m] == key) return m;
84
85         if (X[m] < key) l = m + 1;
86
87         else r = m - 1;
88
89     }
90
91     return -1;
92
93 }
94
95 int main() {
96
97     int n , cas = 1;
98
99     while (~scanf("%d",&n) && n) {
100
101         int m = 0;
102
103         while (n --) {
104
105             double a , b , c ,
106
107             scanf("%lf%lf%lf%lf",
108
109             X[m] = a;
110
111             ss[m++] = Seg(a , c
112
113             = c;
114
115             ++] = Seg(a , c
```

```
116
117     }
118
119     sort(X , X + m);
120
121     sort(ss , ss + m);
122
123     int k = 1;
124
125     for (int i = 1 ; i < m ; i++)
126
127         if (X[i] != X[i-1])
128
129     }
130
131     memset(cnt , 0 , sizeof(cnt));
132
133     memset(sum , 0 , sizeof(sum));
134
135     double ret = 0;
136
137     for (int i = 0 ; i < m - 1 ; i++)
138
139         int l = Bin(ss[i].l);
140
141         int r = Bin(ss[i].r);
142
143         if (l <= r) update
144
145         ret += sum[l] * (ss
146
147     }
148
149     printf("Test case #%d\nTot
150
151 }
152
153 return 0;
154
155 }
```



0

0

题意:矩形周长并

思路:与面积不同的地方是还要记录竖的边有几个
(numseg记录),并且当边界重合的时候需要合并(用
lbd和rbd表示边界来辅助)

线段树操作:update:区间增减 query:直接取根节点的
值



```
1 #include <cstdio>
2
3 #include <cstring>
4
5 #include <cctype>
6
7 #include <algorithm>
8
9 using namespace std;
10
11 #define lson l , m , rt << 1
12
13 #define rson m + 1 , r , rt << 1 | 1
14
15
16
17 const int maxn = 22222;
18
19 struct Seg{
20
21     int l , r , h , s;
22
23     Seg() {}
24
25     Seg(int a,int b,int c,int d):l(a)
26
27     bool operator < (const Seg &cmp)
28
29         return h < cmp.h;
30
31     0      0
32
```



```
33 }ss[maxn];
34
35 bool lbd[maxn<<2] , rbd[maxn<<2];
36
37 int numseg[maxn<<2];
38
39 int cnt[maxn<<2];
40
41 int len[maxn<<2];
42
43 void PushUP(int rt,int l,int r) {
44
45     if (cnt[rt]) {
46
47         lbd[rt] = rbd[rt] = 1;
48
49         len[rt] = r - l + 1;
50
51         numseg[rt] = 2;
52
53     } else if (l == r) {
54
55         len[rt] = numseg[rt] = lbd[rt] = rbd[rt] = 1;
56
57     } else {
58
59         lbd[rt] = lbd[rt<<1];
60
61         rbd[rt] = rbd[rt<<1|1];
62
63         len[rt] = len[rt<<1] + len[rt<<1|1];
64
65         numseg[rt] = numseg[rt<<1] + numseg[rt<<1|1];
66
67         if (lbd[rt<<1|1] && rbd[rt<<1])
68             cnt[rt]++;
69     }
70
71 }
72
73 void build(int c,int l,int r) {
74     0      0
75     R) {
```

```
76
77         cnt[rt] += c;
78
79         PushUP(rt , l , r);
80
81         return ;
82
83     }
84
85     int m = (l + r) >> 1;
86
87     if (L <= m) update(L , R , c , ls);
88
89     if (m < R) update(L , R , c , rs);
90
91     PushUP(rt , l , r);
92
93 }
94
95 int main() {
96
97     int n;
98
99     while (~scanf("%d",&n)) {
100
101         int m = 0;
102
103         int lbd = 10000, rbd = -10000;
104
105         for (int i = 0 ; i < n ; i++) {
106
107             int a , b , c , d;
108
109             scanf("%d%d%d%d",&a,&b,&c,&d);
110
111             lbd = min(lbd , a);
112
113             rbd = max(rbd , c);
114
115             ss[m++] = Seg(a , c , d);
116
117             ss[m++] = Seg(a , b , -d);
118
119             ss[m++] = Seg(b , c , d);
```

```
119         }
120
121         sort(ss , ss + m);
122
123         int ret = 0 , last = 0;
124
125         for (int i = 0 ; i < m ; i++)
126
127             if (ss[i].l < ss[i].r)
128
129                 ret += numseg[1] *
130
131                 ret += abs(len[1] -
132
133                 last = len[1];
134
135         }
136
137         printf("%d\n",ret);
138
139     }
140
141     return 0;
142
143 }
```



练习

hdu3265 Posters

hdu3642 Get The Treasury

poj2482 Stars in Your Window

poj2464 Brownie Points II

hdu3255 Farming

ural1707 Hypnotoad's Secret

uva11983 V

0

0

线段树与其他结合练习(欢迎大家补充):

hdu3333 Turing Tree

hdu3874 Necklace

hdu3016 Man Down

hdu3340 Rain in ACStar

zju3511 Cake Robbery

UESTC1558 Charitable Exchange

CF85-D Sum of Medians

spojGSS2 Can you answer these queries II

一切无法杀死我的，都将使我变得更加强大。

标签: [线段树](#)

好文要顶

关注我

收藏该文



ShawnZhou_Aether

关注 - 12

粉丝 - 53

[+加关注](#)

« [上一篇: Kitty猫基因编码](#)

» [下一篇: 济南清北学堂游记 Day 2.](#)

posted @ 2017-10-29 12:48 ShawnZhou_Aether

阅读(385) 评论(1) 编辑 收藏

评论列表

#1楼 2019-03-0

这个太全了，

0

0

没看过，抽时间

好好学习一下。

[支持\(0\)](#) [反对\(0\)](#)

[刷新评论](#) [刷新页面](#) [返回顶部](#)

**登录后才能发表评论，立即 [登录](#) 或 [注册](#)，[访问](#)
[网站首页](#)**

博客园派送云上免费午餐，AWS注册立享12个月
免费套餐

【推荐】News: 大型组态、工控、仿真、CADGIS 50
万行VC++源码免费下载

【推荐】博客园 & 陌上花开HIMMR 给单身的程序员
小哥哥助力脱单啦~

【推荐】网络安全攻防第1课：攻防靶场、Web渗
透、漏洞利用

【推荐】了不起的开发者，挡不住的华为，园子里的
品牌专区

【推荐】未知数的距离，毫秒间的传递，声网与你实
时互动

【福利】AWS携手博客园为开发者送免费套餐与抵扣
券

Copyright © 2020 ShawnZhou_Aether

Powered by .NET 5.0.0 on Kubernetes

0

0