



- [首页](#)
- [留言板](#)
- [2048](#)

「分块」数列分块入门1 – 9 by hzwer

2018年2月1日28,56128

由于 CH 回档导致原题面丢失，感谢诸暨海亮高级中学帮助重写了题面

已上传至 [LOJ](#)

由于每道题题面太长，限于篇幅，只给出大意，具体题目见小组内赛题，代码附在文末

可能涉及的几个词语解释：

区间：数列中连续一段的元素

区间操作：将某个区间[a,b]的所有元素进行某种改动的操作

块：我们将数列划分成若干个不相交的区间，每个区间称为一个块

整块：在一个区间操作时，完整包含于区间的块

不完整的块：在一个区间操作时，只有部分包含于区间的块，即区间左右端点所在的两个块

分块入门 1 by hzwer

给出一个长为n的数列，以及n个操作，操作涉及区间加法，单点查值。

这是一道能用许多数据结构优化的经典题，可以用于不同数据结构训练。

数列分块就是把数列中每m个元素打包起来，达到优化算法的目的。

以此题为例，如果我们把每m个元素分为一块，共有n/m块，每次区间加的操作会涉及O(n/m)个整块，以及区间两侧两个不完整的块中至多2m个元素。

我们给每个块设置一个加法标记（就是记录这个块中元素一起加了多少），每次操作对每个整块直接O(1)标记，而不完整的块由于元素比较少，暴力修改元素的值。

每次询问时返回元素的值加上其所在块的加法标记。

这样每次操作的复杂度是O(n/m)+O(m)，根据均值不等式，当m取 \sqrt{n} 时总复杂度最低，为了方便，我们都默认下文的分块大小为 \sqrt{n} 。

分块入门 2 by hzwer

给出一个长为 n 的数列，以及 n 个操作，操作涉及区间加法，询问区间内小于某个值 x 的元素个数。

有了上一题的经验，我们可以发现，数列简单分块问题实际上有三项东西要我们思考：

对于每次区间操作：

1. **不完整的块** 的 $O(\sqrt{n})$ 个元素怎么处理？
2. $O(\sqrt{n})$ 个 **整块** 怎么处理？
3. 要预处理什么信息（复杂度不能超过后面的操作）？

我们先来思考只有询问操作的情况，不完整的块枚举统计即可；而要在每个整块内寻找小于一个值的元素数，于是我们不得不要块内元素是有序的，这样就能使用二分法对块内查询，需要预处理时每块做一遍排序，复杂度 $O(n\log n)$ ，每次查询在 \sqrt{n} 个块内二分，以及暴力 $2\sqrt{n}$ 个元素，总复杂度 $O(n\log n + n\sqrt{n}\log\sqrt{n})$ 。

可以通过均值不等式计算出更优的分块大小，就不展开讨论了

那么区间加怎么办呢？

套用第一题的方法，维护一个加法标记，略有区别的地方在于，不完整的块修改后可能会使得该块内数字乱序，所以头尾两个不完整块需要重新排序，复杂度分析略。

在加法标记下的询问操作，块外还是暴力，查询小于 $(x - \text{加法标记})$ 的元素个数，块内用 $(x - \text{加法标记})$ 作为二分的值即可。

分块入门 3 by hzwer

给出一个长为 n 的数列，以及 n 个操作，操作涉及区间加法，询问区间内小于某个值 x 的前驱（比其小的最大元素）。

$n \leq 100000$ 其实是为了区分暴力和一些常数较大的写法。

接着第二题的解法，其实只要把块内查询的二分稍作修改即可。

不过这题其实想表达：可以在块内维护其它结构使其更具有拓展性，比如放一个 **set**，这样如果还有插入、删除元素的操作，会更加的方便。

分块的调试检测技巧：

可以生成一些大数据，然后用两份分块大小不同的代码来对拍，还可以根据运行时间尝试调整分块大小，减小常数。

分块入门 4 by hzwer

给出一个长为 n 的数列，以及 n 个操作，操作涉及区间加法，区间求和。

这题的询问变成了区间上的询问，不完整的块还是暴力；而要想快速统计完整块的答案，需要维护每个块的元素和，先要预处理一下。

考虑区间修改操作，不完整的块直接改，顺便更新块的元素和；完整的块类似之前标记的做法，直接根据块的元素和所加的值计算元素和的增量。

分块入门 5 by hzwer

给出一个长为 n 的数列，以及 n 个操作，操作涉及区间开方，区间求和。

稍作思考可以发现，开方操作比较棘手，主要是对于整块开方时，必须要知道每一个元素，才能知道他们开方后的和，也就是说，难以快速对一个块信息进行更新。

看来我们要另辟蹊径。不难发现，这题的修改就只有下取整开方，而一个数经过几次开方之后，它的值就会变成 0 或者 1。

如果每次区间开方只不涉及完整的块，意味着不超过 $2\sqrt{n}$ 个元素，直接暴力即可。

如果涉及了一些完整的块，这些块经过几次操作以后就会都变成 0 / 1，于是我们采取一种分块优化的暴力做法，只要每个整块暴力开方后，记录一下元素是否都变成了 0 / 1，区间修改时跳过那些全为 0 / 1 的块即可。

这样每个元素至多被开方不超过4次，显然复杂度没有问题。

分块入门 6 by hzwer

给出一个长为 n 的数列，以及 n 个操作，操作涉及单点插入，单点询问，数据随机生成。

先说随机数据的情况

之前提到过，如果我们块内用数组以外的数据结构，能够支持其它不一样的操作，比如此题每块内可以放一个动态的数组，每次插入时先找到位置所在的块，再暴力插入，把块内的其它元素直接向后移动一位，当然用链表也是可以的。

查询的时候类似，复杂度分析略。

但是这样做有个问题，如果数据不随机怎么办？

如果先在一个块有大量单点插入，这个块的大小会大大超过 \sqrt{n} ，那块内的暴力就没有复杂度保证了。

还需要引入一个操作：重新分块（重构）

每根号 n 次插入后，重新把数列平均分一下块，重构需要的复杂度为 $O(n)$ ，重构的次数为 \sqrt{n} ，所以重构的复杂度没有问题，而且保证了每个块的大小相对均衡。

当然，也可以当某个块过大时重构，或者只把这个块分成两半。

分块入门 7 by hzwer

给出一个长为 n 的数列，以及 n 个操作，操作涉及区间乘法，区间加法，单点询问。

很显然，如果只有区间乘法，和分块入门 1 的做法没有本质区别，但要思考如何同时维护两种标记。

我们让乘法标记的优先级高于加法（如果反过来的话，新的加法标记无法处理）

若当前的一个块乘以 m_1 后加上 a_1 ，这时进行一个乘 m_2 的操作，则原来的标记变成 m_1*m_2 ， a_1*m_2

若当前的一个块乘以 m_1 后加上 a_1 ，这时进行一个加 a_2 的操作，则原来的标记变成 m_1 ， a_1+a_2

分块入门 8 by hzwer

给出一个长为 n 的数列，以及 n 个操作，操作涉及区间询问等于一个数 c 的元素，并将这个区间的所有元素改为 c 。

区间修改没有什么难度，这题难在区间查询比较奇怪，因为权值种类比较多，似乎没有什么好的维护方法。

模拟一些数据可以发现，询问后一整段都会被修改，几次询问后数列可能只剩下几段不同的区间了。

我们思考这样一个暴力，还是分块，维护每个分块是否只有一种权值，区间操作的时候，对于同权值的一个块就 $O(1)$ 统计答案，否则暴力统计答案，并修改标记，不完整的块也暴力。

这样看似最差情况每次都会耗费 $O(n)$ 的时间，但其实可以这样分析：

假设初始序列都是同一个值，那么查询是 $O(\sqrt{n})$ ，如果这时进行一个区间操作，它最多破坏首尾2个块的标记，所以只能使后面的询问至多多2个块的暴力时间，所以均摊每次操作复杂度还是 $O(\sqrt{n})$ 。

换句话说，要想让一个操作耗费 $O(n)$ 的时间，要先花费 \sqrt{n} 个操作对数列进行修改。

初始序列不同值，经过类似分析后，就可以放心的暴力啦。

分块入门 9 by hzwer

给出一个长为 n 的数列，以及 n 个操作，操作涉及询问区间的最小众数。

这是一道经典难题，其实可以支持修改操作，具体见陈立杰大神的区间众数解题报告。

而且不强制在线的话有很多做法，可以看我blog一道类似题目 [czy的后宫3](#)

bzoj2724 是道强制在线区间众数，而且题目背景写的不错，这道题的题解就贴传送门咯

[\[bzoj2724\]](#) [\[Violet 6\]](#)蒲公英

程序

分块入门1:

```
1  #include<map>
2  #include<set>
3  #include<cmath>
4  #include<stack>
5  #include<queue>
6  #include<cstdio>
7  #include<vector>
8  #include<cstring>
9  #include<cstdlib>
10 #include<iostream>
11 #include<algorithm>
12 #define mod 998244353
13 #define pi acos(-1)
14 #define inf 0x7fffffff
15 #define ll long long
16 using namespace std;
17 ll read()
18 {
19     ll x=0,f=1;char ch=getchar();
20     while(ch<'0' || ch>'9'){if(ch=='-')f=-1;ch=getchar();}
21     while(ch>='0' && ch<='9'){x=x*10+ch-'0';ch=getchar();}
22     return x*f;
23 }
```

```

24 int n,blo;
25 int v[50005],bl[50005],atag[50005];
26 void add(int a,int b,int c)
27 {
28     for(int i=a;i<=min(bl[a]*blo,b);i++)
29         v[i]+=c;
30     if(bl[a]!=bl[b])
31         for(int i=(bl[b]-1)*blo+1;i<=b;i++)
32             v[i]+=c;
33     for(int i=bl[a]+1;i<=bl[b]-1;i++)
34         atag[i]+=c;
35 }
36 int main()
37 {
38     n=read();blo=sqrt(n);
39     for(int i=1;i<=n;i++)v[i]=read();
40     for(int i=1;i<=n;i++)bl[i]=(i-1)/blo+1;
41     for(int i=1;i<=n;i++)
42     {
43         int f=read(),a=read(),b=read(),c=read();
44         if(f==0)add(a,b,c);
45         if(f==1)printf("%d\n",v[b]+atag[bl[b]]);
46     }
47     return 0;
48 }

```

分块入门2:

```

1  #include<map>
2  #include<set>
3  #include<cmath>
4  #include<stack>
5  #include<queue>
6  #include<cstdio>
7  #include<vector>
8  #include<cstring>
9  #include<cstdlib>
10 #include<iostream>
11 #include<algorithm>
12 #define mod 998244353
13 #define pi acos(-1)
14 #define inf 0x7fffffff
15 #define ll long long
16 using namespace std;
17 ll read()
18 {
19     ll x=0,f=1;char ch=getchar();
20     while(ch<'0' || ch>'9'){if(ch=='-')f=-1;ch=getchar();}
21     while(ch>='0' && ch<='9'){x=x*10+ch-'0';ch=getchar();}
22     return x*f;
23 }
24 int n,blo;
25 int v[50005],bl[50005],atag[50005];
26 vector<int>ve[505];
27 void reset(int x)
28 {
29     ve[x].clear();
30     for(int i=(x-1)*blo+1;i<=min(x*blo,n);i++)
31         ve[x].push_back(v[i]);
32     sort(ve[x].begin(),ve[x].end());
33 }
34 void add(int a,int b,int c)
35 {
36     for(int i=a;i<=min(bl[a]*blo,b);i++)
37         v[i]+=c;
38     reset(bl[a]);
39     if(bl[a]!=bl[b])
40     {

```

```

41     for(int i=(bl[b]-1)*blo+1;i<=b;i++)
42         v[i]+=c;
43     reset(bl[b]);
44 }
45 for(int i=bl[a]+1;i<=bl[b]-1;i++)
46     atag[i]+=c;
47 }
48 int query(int a,int b,int c)
49 {
50     int ans=0;
51     for(int i=a;i<=min(bl[a]*blo,b);i++)
52         if(v[i]+atag[bl[a]]<c)ans++;
53     if(bl[a]!=bl[b])
54         for(int i=(bl[b]-1)*blo+1;i<=b;i++)
55             if(v[i]+atag[bl[b]]<c)ans++;
56     for(int i=bl[a]+1;i<=bl[b]-1;i++)
57     {
58         int x=c-atag[i];
59         ans+=lower_bound(ve[i].begin(),ve[i].end(),x)-ve[i].begin();
60     }
61     return ans;
62 }
63 int main()
64 {
65     n=read();blo=sqrt(n);
66     for(int i=1;i<=n;i++)v[i]=read();
67     for(int i=1;i<=n;i++)
68     {
69         bl[i]=(i-1)/blo+1;
70         ve[bl[i]].push_back(v[i]);
71     }
72     for(int i=1;i<=bl[n];i++)
73         sort(ve[i].begin(),ve[i].end());
74     for(int i=1;i<=n;i++)
75     {
76         int f=read(),a=read(),b=read(),c=read();
77         if(f==0)add(a,b,c);
78         if(f==1)printf("%d\n",query(a,b,c*c));
79     }
80     return 0;
81 }

```

分块入门3:

```

1  #include<map>
2  #include<set>
3  #include<cmath>
4  #include<stack>
5  #include<queue>
6  #include<cstdio>
7  #include<vector>
8  #include<cstring>
9  #include<cstdlib>
10 #include<iostream>
11 #include<algorithm>
12 #define mod 998244353
13 #define pi acos(-1)
14 #define inf 0x7fffffff
15 #define ll long long
16 using namespace std;
17 ll read()
18 {
19     ll x=0,f=1;char ch=getchar();
20     while(ch<'0' || ch>'9'){if(ch=='-')f=-1;ch=getchar();}
21     while(ch>='0'&&ch<='9'){x=x*10+ch-'0';ch=getchar();}
22     return x*f;
23 }
24 int n,blo;

```

```

25 int v[100005],bl[100005],atag[100005];
26 set<int>st[105];
27 void add(int a,int b,int c)
28 {
29     for(int i=a;i<=min(bl[a]*blo,b);i++)
30     {
31         st[bl[a]].erase(v[i]);
32         v[i]+=c;
33         st[bl[a]].insert(v[i]);
34     }
35     if(bl[a]!=bl[b])
36     {
37         for(int i=(bl[b]-1)*blo+1;i<=b;i++)
38         {
39             st[bl[b]].erase(v[i]);
40             v[i]+=c;
41             st[bl[b]].insert(v[i]);
42         }
43     }
44     for(int i=bl[a]+1;i<=bl[b]-1;i++)
45         atag[i]+=c;
46 }
47 int query(int a,int b,int c)
48 {
49     int ans=-1;
50     for(int i=a;i<=min(bl[a]*blo,b);i++)
51     {
52         int val=v[i]+atag[bl[a]];
53         if(val<c)ans=max(val,ans);
54     }
55     if(bl[a]!=bl[b])
56     for(int i=(bl[b]-1)*blo+1;i<=b;i++)
57     {
58         int val=v[i]+atag[bl[b]];
59         if(val<c)ans=max(val,ans);
60     }
61     for(int i=bl[a]+1;i<=bl[b]-1;i++)
62     {
63         int x=c-atag[i];
64         set<int>::iterator it=st[i].lower_bound(x);
65         if(it==st[i].begin())continue;
66         --it;
67         ans=max(ans,*it+atag[i]);
68     }
69     return ans;
70 }
71 int main()
72 {
73     n=read();blo=1000;
74     for(int i=1;i<=n;i++)v[i]=read();
75     for(int i=1;i<=n;i++)
76     {
77         bl[i]=(i-1)/blo+1;
78         st[bl[i]].insert(v[i]);
79     }
80     for(int i=1;i<=n;i++)
81     {
82         int f=read(),a=read(),b=read(),c=read();
83         if(f==0)add(a,b,c);
84         if(f==1)printf("%d\n",query(a,b,c));
85     }
86     return 0;
87 }

```

分块入门4:

```

1 #include<map>
2 #include<set>

```

```

3  #include<cmath>
4  #include<stack>
5  #include<queue>
6  #include<cstdio>
7  #include<vector>
8  #include<cstring>
9  #include<cstdlib>
10 #include<iostream>
11 #include<algorithm>
12 #define mod 998244353
13 #define pi acos(-1)
14 #define inf 0x7fffffff
15 #define ll long long
16 using namespace std;
17 ll read()
18 {
19     ll x=0,f=1;char ch=getchar();
20     while(ch<'0' || ch>'9'){if(ch=='-')f=-1;ch=getchar();}
21     while(ch>='0'&&ch<='9'){x=x*10+ch-'0';ch=getchar();}
22     return x*f;
23 }
24 int n,blo;
25 int bl[50005];
26 ll v[50005],atag[50005],sum[50005];
27 void add(int a,int b,int c)
28 {
29     for(int i=a;i<=min(bl[a]*blo,b);i++)
30         v[i]+=c,sum[bl[a]]+=c;;
31     if(bl[a]!=bl[b])
32         for(int i=(bl[b]-1)*blo+1;i<=b;i++)
33             v[i]+=c,sum[bl[b]]+=c;
34     for(int i=bl[a]+1;i<=bl[b]-1;i++)
35         atag[i]+=c;
36 }
37 ll query(int a,int b)
38 {
39     ll ans=0;
40     for(int i=a;i<=min(bl[a]*blo,b);i++)
41         ans+=v[i]+atag[bl[a]];
42     if(bl[a]!=bl[b])
43         for(int i=(bl[b]-1)*blo+1;i<=b;i++)
44             ans+=v[i]+atag[bl[b]];
45     for(int i=bl[a]+1;i<=bl[b]-1;i++)
46         ans+=sum[i]+blo*atag[i];
47     return ans;
48 }
49 int main()
50 {
51     n=read();blo=sqrt(n);
52     for(int i=1;i<=n;i++)v[i]=read();
53     for(int i=1;i<=n;i++)
54     {
55         bl[i]=(i-1)/blo+1;
56         sum[bl[i]]+=v[i];
57     }
58     for(int i=1;i<=n;i++)
59     {
60         int f=read(),a=read(),b=read(),c=read();
61         if(f==0)add(a,b,c);
62         if(f==1)
63             printf("%d\n",query(a,b)%(c+1));
64     }
65     return 0;
66 }

```

分块入门5:

```
1 #include<map>
```



```

2  #include<set>
3  #include<cmath>
4  #include<stack>
5  #include<queue>
6  #include<cstdio>
7  #include<vector>
8  #include<cstring>
9  #include<cstdlib>
10 #include<iostream>
11 #include<algorithm>
12 #define mod 998244353
13 #define pi acos(-1)
14 #define inf 0x7fffffff
15 #define ll long long
16 using namespace std;
17 ll read()
18 {
19     ll x=0,f=1;char ch=getchar();
20     while(ch<'0' || ch>'9'){if(ch=='-')f=-1;ch=getchar();}
21     while(ch>='0' && ch<='9'){x=x*10+ch-'0';ch=getchar();}
22     return x*f;
23 }
24 int n,blo;
25 int bl[50005];
26 int v[50005],sum[50005],flag[50005];
27 void solve_sqrt(int x)
28 {
29     if(flag[x])return;
30     flag[x]=1;
31     sum[x]=0;
32     for(int i=(x-1)*blo+1;i<=x*blo;i++)
33     {
34         v[i]=sqrt(v[i]),sum[x]+=v[i];
35         if(v[i]>1)flag[x]=0;
36     }
37 }
38 void add(int a,int b,int c)
39 {
40     for(int i=a;i<=min(bl[a]*blo,b);i++)
41     {
42         sum[bl[a]]-=v[i];
43         v[i]=sqrt(v[i]);
44         sum[bl[a]]+=v[i];
45     }
46     if(bl[a]!=bl[b])
47         for(int i=(bl[b]-1)*blo+1;i<=b;i++)
48         {
49             sum[bl[b]]-=v[i];
50             v[i]=sqrt(v[i]);
51             sum[bl[b]]+=v[i];
52         }
53     for(int i=bl[a]+1;i<=bl[b]-1;i++)
54         solve_sqrt(i);
55 }
56 int query(int a,int b)
57 {
58     int ans=0;
59     for(int i=a;i<=min(bl[a]*blo,b);i++)
60         ans+=v[i];
61     if(bl[a]!=bl[b])
62         for(int i=(bl[b]-1)*blo+1;i<=b;i++)
63             ans+=v[i];
64     for(int i=bl[a]+1;i<=bl[b]-1;i++)
65         ans+=sum[i];
66     return ans;
67 }
68 int main()
69 {
70     n=read();blo=sqrt(n);

```

```

71     for(int i=1;i<=n;i++)v[i]=read();
72     for(int i=1;i<=n;i++)
73     {
74         bl[i]=(i-1)/blo+1;
75         sum[bl[i]]+=v[i];
76     }
77     for(int i=1;i<=n;i++)
78     {
79         int f=read(),a=read(),b=read(),c=read();
80         if(f==0)add(a,b,c);
81         if(f==1)
82             printf("%d\n",query(a,b));
83     }
84     return 0;
85 }

```

分块入门6:

```

1  #include<map>
2  #include<set>
3  #include<cmath>
4  #include<stack>
5  #include<queue>
6  #include<cstdio>
7  #include<vector>
8  #include<cstring>
9  #include<cstdlib>
10 #include<iostream>
11 #include<algorithm>
12 #define mod 998244353
13 #define pi acos(-1)
14 #define inf 0x7fffffff
15 #define ll long long
16 using namespace std;
17 ll read()
18 {
19     ll x=0,f=1;char ch=getchar();
20     while(ch<'0' || ch>'9'){if(ch=='-')f=-1;ch=getchar();}
21     while(ch>='0'&&ch<='9'){x=x*10+ch-'0';ch=getchar();}
22     return x*f;
23 }
24 int n,blo,m;
25 int v[100005];
26 vector<int>ve[1005];
27 int st[200005],top;
28 pair<int,int> query(int b)
29 {
30     int x=1;
31     while(b>ve[x].size())
32         b-=ve[x].size(),x++;
33     return make_pair(x,b-1);
34 }
35 void rebuild()
36 {
37     top=0;
38     for(int i=1;i<=m;i++)
39     {
40         for(vector<int>::iterator j=ve[i].begin();j!=ve[i].end();j++)
41             st[++top]=*j;
42         ve[i].clear();
43     }
44     int blo2=sqrt(top);
45     for(int i=1;i<=top;i++)
46         ve[(i-1)/blo2+1].push_back(st[i]);
47     m=(top-1)/blo2+1;
48 }
49 void insert(int a,int b)
50 {

```

```

51     pair<int,int> t=query(a);
52     ve[t.first].insert(ve[t.first].begin()+t.second,b);
53     if(ve[t.first].size()>20*blo)
54         rebuild();
55 }
56 int main()
57 {
58     n=read();blo=sqrt(n);
59     for(int i=1;i<=n;i++)v[i]=read();
60     for(int i=1;i<=n;i++)
61         ve[(i-1)/blo+1].push_back(v[i]);
62     m=(n-1)/blo+1;
63     for(int i=1;i<=n;i++)
64     {
65         int f=read(),a=read(),b=read(),c=read();
66         if(f==0)insert(a,b);
67         if(f==1)
68         {
69             pair<int,int> t=query(b);
70             printf("%d\n",ve[t.first][t.second]);
71         }
72     }
73     return 0;
74 }

```

分块入门7:

```

1  #include<map>
2  #include<set>
3  #include<cmath>
4  #include<stack>
5  #include<queue>
6  #include<cstdio>
7  #include<vector>
8  #include<cstring>
9  #include<cstdlib>
10 #include<iostream>
11 #include<algorithm>
12 #define mod 10007
13 #define pi acos(-1)
14 #define inf 0x7fffffff
15 #define ll long long
16 using namespace std;
17 ll read()
18 {
19     ll x=0,f=1;char ch=getchar();
20     while(ch<'0' || ch>'9'){if(ch=='-')f=-1;ch=getchar();}
21     while(ch>='0'&&ch<='9'){x=x*10+ch-'0';ch=getchar();}
22     return x*f;
23 }
24 int n,blo;
25 int v[100005],bl[100005],atag[1005],mtag[1005];
26 void reset(int x)
27 {
28     for(int i=(x-1)*blo+1;i<=min(n,x*blo);i++)
29         v[i]=(v[i]*mtag[x]+atag[x])%mod;
30     atag[x]=0;mtag[x]=1;
31 }
32 void solve(int f,int a,int b,int c)
33 {
34     reset(bl[a]);
35     for(int i=a;i<=min(bl[a]*blo,b);i++)
36     {
37         if(f==0)v[i]+=c;
38         else v[i]*=c;
39         v[i]%=mod;
40     }
41     if(bl[a]!=bl[b])

```

```

42 {
43     reset(bl[b]);
44     for(int i=(bl[b]-1)*blo+1;i<=b;i++)
45     {
46         if(f==0)v[i]+=c;
47         else v[i]*=c;
48         v[i]%=mod;
49     }
50 }
51 for(int i=bl[a]+1;i<=bl[b]-1;i++)
52 {
53     if(f==0)atag[i]=(atag[i]+c)%mod;
54     else
55     {
56         atag[i]=(atag[i]*c)%mod;
57         mtag[i]=(mtag[i]*c)%mod;
58     }
59 }
60 }
61 int main()
62 {
63     n=read();blo=sqrt(n);
64     for(int i=1;i<=n;i++)v[i]=read();
65     for(int i=1;i<=n;i++)bl[i]=(i-1)/blo+1;
66     for(int i=1;i<=bl[n];i++)mtag[i]=1;
67     for(int i=1;i<=n;i++)
68     {
69         int f=read(),a=read(),b=read(),c=read();
70         if(f==2)printf("%d\n",(v[b]*mtag[bl[b]]+atag[bl[b]])%mod);
71         else solve(f,a,b,c);
72     }
73     return 0;
74 }

```

分块入门8:

```

1  #include<map>
2  #include<set>
3  #include<cmath>
4  #include<stack>
5  #include<queue>
6  #include<cstdio>
7  #include<vector>
8  #include<cstring>
9  #include<cstdlib>
10 #include<iostream>
11 #include<algorithm>
12 #define mod 998244353
13 #define pi acos(-1)
14 #define inf 0x7fffffff
15 #define ll long long
16 using namespace std;
17 ll read()
18 {
19     ll x=0,f=1;char ch=getchar();
20     while(ch<'0' || ch>'9'){if(ch=='-')f=-1;ch=getchar();}
21     while(ch>='0'&&ch<='9'){x=x*10+ch-'0';ch=getchar();}
22     return x*f;
23 }
24 int n,blo;
25 int v[100005],bl[100005],tag[100005];
26 void reset(int x)
27 {
28     if(tag[x]==-1)return;
29     for(int i=(x-1)*blo+1;i<=blo*x;i++)
30         v[i]=tag[x];
31     tag[x]=-1;
32 }

```

```

33 int solve(int a,int b,int c)
34 {
35     int ans=0;
36     reset(bl[a]);
37     for(int i=a;i<=min(bl[a]*blo,b);i++)
38         if(v[i]!=c)v[i]=c;
39         else ans++;
40     if(bl[a]!=bl[b])
41     {
42         reset(bl[b]);
43         for(int i=(bl[b]-1)*blo+1;i<=b;i++)
44             if(v[i]!=c)v[i]=c;
45             else ans++;
46     }
47     for(int i=bl[a]+1;i<=bl[b]-1;i++)
48         if(tag[i]!=-1)
49         {
50             if(tag[i]!=c)tag[i]=c;
51             else ans+=blo;
52         }
53     else
54     {
55         for(int j=(i-1)*blo+1;j<=i*blo;j++)
56             if(v[j]!=c)v[j]=c;
57             else ans++;
58         tag[i]=c;
59     }
60     return ans;
61 }
62 int main()
63 {
64     memset(tag,-1,sizeof(tag));
65     n=read();blo=sqrt(n);
66     for(int i=1;i<=n;i++)v[i]=read();
67     for(int i=1;i<=n;i++)bl[i]=(i-1)/blo+1;
68     for(int i=1;i<=n;i++)
69     {
70         int a=read(),b=read(),c=read();
71         printf("%d\n",solve(a,b,c));
72     }
73     return 0;
74 }

```

分块入门9:

```

1  #include<map>
2  #include<set>
3  #include<cmath>
4  #include<stack>
5  #include<queue>
6  #include<cstdio>
7  #include<vector>
8  #include<cstring>
9  #include<cstdlib>
10 #include<iostream>
11 #include<algorithm>
12 #define mod 10007
13 #define pi acos(-1)
14 #define inf 0x7fffffff
15 #define ll long long
16 using namespace std;
17 ll read()
18 {
19     ll x=0,f=1;char ch=getchar();
20     while(ch<'0' || ch>'9'){if(ch=='-')f=-1;ch=getchar();}
21     while(ch>='0' && ch<='9'){x=x*10+ch-'0';ch=getchar();}
22     return x*f;
23 }

```

```

24 int n,blo,id;
25 int v[50005],bl[50005];
26 int f[505][505];
27 map<int,int>mp;
28 int val[50005],cnt[50005];
29 vector<int>ve[50005];
30 void pre(int x)
31 {
32     memset(cnt,0,sizeof(cnt));
33     int mx=0,ans=0;
34     for(int i=(x-1)*blo+1;i<=n;i++)
35     {
36         cnt[v[i]]++;
37         int t=bl[i];
38         if(cnt[v[i]]>mx||(cnt[v[i]]==mx&&val[v[i]]<val[ans]))
39             ans=v[i],mx=cnt[v[i]];
40         f[x][t]=ans;
41     }
42 }
43 int query(int l,int r,int x)
44 {
45     int t=upper_bound(ve[x].begin(),ve[x].end(),r)-lower_bound(ve[x].begin(),ve[x].end(),l);
46     return t;
47 }
48 int query(int a,int b)
49 {
50     int ans,mx;
51     ans=f[bl[a]+1][bl[b]-1];
52     mx=query(a,b,ans);
53     for(int i=a;i<=min(bl[a]*blo,b);i++)
54     {
55         int t=query(a,b,v[i]);
56         if(t>mx||(t==mx&&val[v[i]]<val[ans]))ans=v[i],mx=t;
57     }
58     if(bl[a]!=bl[b])
59         for(int i=(bl[b]-1)*blo+1;i<=b;i++)
60         {
61             int t=query(a,b,v[i]);
62             if(t>mx||(t==mx&&val[v[i]]<val[ans]))ans=v[i],mx=t;
63         }
64     return ans;
65 }
66 int main()
67 {
68     n=read();
69     blo=200;
70     for(int i=1;i<=n;i++)
71     {
72         v[i]=read();
73         if(!mp[v[i]])
74         {
75             mp[v[i]]=++id;
76             val[id]=v[i];
77         }
78         v[i]=mp[v[i]];
79         ve[v[i]].push_back(i);
80     }
81     for(int i=1;i<=n;i++)bl[i]=(i-1)/blo+1;
82     for(int i=1;i<=bl[n];i++)pre(i);
83     for(int i=1;i<=n;i++)
84     {
85         int a=read(),b=read();
86         if(a>b)swap(a,b);
87         printf("%d\n",val[query(a,b)]);
88     }
89     return 0;
90 }

```

说点什么

提醒 新跟进评论

邮箱



b

i

link

b-quote

u

ul

ol

li

code

spoiler

说点什么



Lob

分块9

萌新的疑惑，为什么这里右边是l,不是l-1,前缀和思想查询l-r区间不应该是 [r] - [l-1] 么
int t=upper_bound(ve[x].begin(),ve[x].end(),r)-lower_bound(ve[x].begin(),ve[x].end(),l);

↩ 回复

🕒 17 天 24 分钟 之前



Bocity

分块9这边标算好像挂了

↩ 回复

🕒 22 天 9 小时 之前



Steauk

分块入门9一堆人离散化直接基于O(数值的种类的平方)暴力离散化过了，然后求答案时，直接基于O(数值的种类乘n)的暴力，也过了，啥时候卡卡？

↩ 回复

🕒 3 月 24 天 之前



kcfzyhq

<https://loj.ac/article/485> 学长看看这个帖子

↩ 回复

🕒 6 月 1 小时 之前



WHZ0325

LibreOJ上的题目链接: <https://loj.ac/problems/tag/207>

↩ 回复

🕒 6 月 26 天 之前



DaCong

问一个问题，为什么在第二个程序和第九个程序中，分块大小是指定的数字，而其他的则是 \sqrt{n} ，有什么讲究吗？为什么要在那两个程序中用具体的数字呢？

↩ 回复

🕒 7 月 7 天 之前 ^



hzwer

有时分块大小理论上是带根号内 \log 的

把它设置成具体的数字也可能更方便调试

↩ 回复

🕒 7 月 5 天 之前



enfris

hzwer大神! 我觉得在分块入门2 中, 查询的时候好像应该是
query (a,b,c) 啊, 这里好像把c平方了, 跑了一个小数据, 好像有点问题:
7
1 2 3 4 5 6 7
1 4 7 3
答案应该是0, 但这里成了4.....

↩ 回复

🕒 7月14天之前 ^



purple_bro

分块2的代码是对应分块2的题写的, LOJ上面分块二要求是查询 c^2 的

↩ 回复

🕒 5月13天之前



23forever

分块入门8中, 在统计整块信息时, 如果 $tag[i] = c$ 的话答案直接加了一个整块的大小。但如果当前处理的是最后一个块 (有可能不完整) 的话, 答案不就不对了吗?

↩ 回复

🕒 8月25天之前 ^



bestFy

不完整的块是暴力算的, 完整的块才是整块地统计

↩ 回复

🕒 8月23天之前 ^



vicotirque

不是这个意思吧, 是分完块之后最后的一块可能不够那么多啊, 就会少。

↩ 回复

🕒 7月22天之前 ^



xMinh

对啊, 最后一块是会少。但是我们实现程序的时候是不会把这个最后一块直接加上block的, 而是会暴力来算.....把程序读懂就明白了。

↩ 回复

🕒 7月22天之前



Ays

分块算法真的挺厉害的啊。。

↩ 回复

🕒 8月25天之前



数列分块入门九题 - PerfectPan's Blog

[...] 黄学长博客传送 数列分块九题 [...]

↩ 回复

🕒 9月1天之前



bestFy

大爱hzwer出教程!

↩ 回复

🕒 9月21天之前



thmyl

好喜欢黄学长的代码风格

↩ 回复

🕒 9月27天之前



lijiannan

学长 比赛挂了QAQ

回复

1 年 7 月 之前



byene

<http://www.contesthunter.org>。这个网址是内网吗？怎么无法注册？

回复

2 年 4 月 之前



hzwer

注册确认的时候把网址改成IP

回复

2 年 4 月 之前



Cha

这个网站证书过期了注册不了。。

回复

2 年 3 月 之前



基础算法分类刷题表 by hzwer - 算法 - hzwer.com

[...] 【分块】数列分块入门1-9 by hzwer [...]

回复

2 年 4 月 之前



LYZer

%%%

回复

2 年 4 月 之前



CZHer

%%%

回复

2 年 4 月 之前



LXHer

%%%

回复

2 年 4 月 之前



CJHer

%%%

回复

2 年 4 月 之前



黄学长好棒！

%%%黄学长

回复

2 年 4 月 之前



Flandre-Scarlet

大爱hzwer出教程！

回复

2 年 4 月 之前

分类目录

分类目录

计数器 5471947

「优于别人，并不高贵，真正的高贵应该是优于过去的自己。——真实的高贵」

点击开始 Adobe Flash
Player.

热门文章

- 28561 [「分块」数列分块入门1 - 9 by hzwer](#)
- 9835 [「BZOJ2002」「HNOI2010」Bounce 弹飞绵羊](#)
- 7948 [「BZOJ2724」「Violet 6」蒲公英](#)
- 7361 [「BZOJ3809」Gty的二逼妹子序列](#)
- 5901 [「BZOJ3343」教主的魔法](#)
- 5547 [「BZOJ2821」作诗\(Poetize\)](#)
- 5506 [「BZOJ2453」维护队列](#)
- 5453 [「BZOJ2120」数颜色](#)
- 4256 [「BZOJ2141」排队](#)
- 3560 [「BZOJ2388」旅行规划](#)
- 2835 [「小奇模拟赛2」小奇的危机](#)
- 2541 [「CF551X」Codeforces Round #307 \(Div. 2\)](#)
- 2518 [「JoyOI1463」智商问题](#)
- 2468 [「codechefFNCS」Chef and Churu](#)

Myfriends

- [The one](#)
- [floz](#)
- [dx](#)
- [kzoacn](#)
- [wmdcstdio](#)
- [ExfJoe](#)
- [miskcoo](#)
- [jmas2711](#)
- [n+e](#)
- [Mektpoy](#)
- [skydec](#)
- [kuribohG](#)
- [Memphis](#)
- [touko](#)
- [zls](#)

近期评论

- [Flandre-Scarlet](#)发表在《[北京大学计算概论A 2018年期中考试](#)》
- [Flandre-Scarlet](#)发表在《[北京大学计算概论A 2018年期中考试](#)》

- [Flandre·Scarlet](#)发表在《[OI课件题目分享 by hzwer](#)》
- [Flandre·Scarlet](#)发表在《[「czy系列赛」czy的后宫3](#)》
- [Flandre·Scarlet](#)发表在《[「NOIP模拟赛」序列问题](#)》

Copyright © 2009-2012 [hzwer.com](#) All Rights Reserved! Powered by WordPress And Theme By [lianyue](#)

[登录](#) | [联系站长](#)