

并查集

[P3367 【模板】并查集](#)

```
1  #include<bits/stdc++.h>
2  #define FOR(i,a,b) for(int i=(a);i<=(b);++i)
3  using namespace std;
4
5  int f[10007];
6
7  int find(int x){
8      if(f[x]==x) return x;
9      return f[x]=find(f[x]);
10 }
11
12 int main(){
13     ios_base::sync_with_stdio(0);
14     cin.tie(0);
15     int n,m;
16     cin>>n>>m;
17     FOR(i,1,n)
18         f[i]=i;
19
20     int z,x,y;
21     FOR(i,1,m){
22         cin>>z>>x>>y;
23         if(z==1) f[find(x)]=find(y); //合并
24         if(z==2){ //查找
25             if(find(x)==find(y)) cout<<"Y\n";
26             else cout<<"N\n";
27         }
28     }
29     return 0;
30 }
```

ST表

[P3865 【模板】ST 表](#)

[参考1](#)

[参考2](#)

```
1  #include<bits/stdc++.h>
2  #define FOR(i,a,b) for(int i=(a);i<=(b);++i)
3  using namespace std;
4
```

```

5  const int MAXJ=log2((int)(1e5))+1;//17
6  const int MAXN=1e6+10;
7
8  inline int IntRead(){//快速读入比关闭同步的cin快得多
9      char ch=getchar();
10     int s=0,w=1;
11     while(ch<'0' || ch>'9'){
12         if(ch=='-') w=-1;
13         ch=getchar();
14     }
15     while(ch>='0' && ch<='9'){
16         s=s*10+ch-'0',
17         ch=getchar();
18     }
19     return s*w;
20 }
21
22 int Max[MAXN][17];
23
24 int Query(int l,int r){
25     int k=log2(r-l+1);
26     return max(Max[l][k],Max[r-(1<<k)+1][k]);//把拆出来的区间分别取最值
27 }
28
29 int main(){
30     int n=IntRead(),m=IntRead();
31     FOR(i,1,n)
32         Max[i][0]=IntRead();
33     FOR(j,1,MAXJ)
34         for(int i=1;i+(1<<j)-1<=n;i++)//注意这里要控制边界
35             Max[i][j]=max(Max[i][j-1],Max[i+(1<<(j-1))][j-1]);
36     FOR(i,1,m){
37         int l=IntRead(),r=IntRead();
38         printf("%d\n",Query(l,r));
39     }
40     return 0;
41 }

```

树状数组

单点修改、区间查询

[P3374 【模板】树状数组 1](#)

```

1  #include<bits/stdc++.h>
2  #define FOR(i,a,b) for(int i=(a);i<=(b);++i)
3  #define elif else if
4  using namespace std;

```

```

5
6  int a[500100],c[500100],n,m;
7
8  int lowbit(int x){
9      return x&-x;
10 }
11
12 void add(int x,int v){
13     while(x<=n)
14         c[x]+=v,x+=lowbit(x);
15 }
16
17 int sum(int x){
18     int ans=0;
19     while(x>=1)
20         ans+=c[x],x-=lowbit(x);
21     return ans;
22 }
23
24 int SUM(int l,int r){
25     return sum(r)-sum(l-1);
26 }
27
28 int main(){
29     ios_base::sync_with_stdio(0);
30     cin.tie(0);
31     int i,cmd,x,y;
32     cin>>n>>m;
33     FOR(i,1,n)
34         cin>>a[i],add(i,a[i]);
35     while(m--){
36         cin>>cmd>>x>>y;
37         if(cmd==1) add(x,y);
38         elif(cmd==2) cout<<SUM(x,y)<<'\\n';
39     }
40     return 0;
41 }

```

区间修改、单点查询

[P3368 【模板】树状数组 2](#)

```

1  #include<bits/stdc++.h>
2  #define FOR(i,a,b) for(int i=(a);i<=(b);++i)
3  #define elif else if
4  using namespace std;
5
6  int a[500100],c[500100],n,m;
7

```

```

8  int lowbit(int x){
9      return x&-x;
10 }
11
12 void add(int x,int v){
13     while(x<=n)
14         c[x]+=v,x+=lowbit(x);
15 }
16
17 int query(int x){
18     int ans=0;
19     while(x>=1)
20         ans+=c[x],x-=lowbit(x);
21     return ans;
22 }
23
24 int main(){
25     ios_base::sync_with_stdio(0);
26     cin.tie(0);
27     int cmd,x,y,k;
28     cin>>n>>m;
29     int past=0,now;
30     FOR(i,1,n){
31         cin>>now;
32         add(i,now-past);
33         past=now;
34     }
35     while(m--){
36         cin>>cmd;
37         if(cmd==1){
38             cin>>x>>y>>k;
39             add(x,k);
40             add(y+1,-k);
41         }
42         elif(cmd==2){
43             cin>>x;
44             cout<<query(x)<<'\\n';
45         }
46     }
47     return 0;
48 }

```

区间最大值

```

1  #include<bits/stdc++.h>
2  #define FOR(i,a,b) for(int i=(a);i<=(b);++i)
3  using namespace std;
4

```

```

5  const int MAXN=5e7+7;
6  int a[MAXN],h[MAXN];
7  int n,m;
8
9  int lowbit(int x){
10     return x&(-x);
11 }
12
13 void updata(int x){
14     int lx,i;
15     while(x<=n){
16         h[x]=a[x];
17         lx=lowbit(x);
18         for(i=1;i<lx;i<=1)
19             h[x]=max(h[x],h[x-i]);
20         x+=lowbit(x);
21     }
22 }
23
24 int query(int x,int y){
25     int ans=0;
26     while(y>=x){
27         ans=max(a[y],ans);
28         --y;
29         for(;y-lowbit(y)>=x;y-=lowbit(y))
30             ans=max(h[y],ans);
31     }
32     return ans;
33 }
34
35 int main(){
36     cin.tie(0)->sync_with_stdio(0);
37     int x,y,ans;
38     cin>>n>>m;
39     FOR(i,1,n){
40         cin>>a[i]
41         updata(i);
42     }
43     FOR(i,1,m){
44         cin>>x>>y;
45         ans=query(x,y);
46         cout<<ans<<'\\n';
47     }
48     return 0;
49 }

```

线段树

P3372 【模板】线段树 1

```
1  #include<bits/stdc++.h>
2  #define FOR(i,a,b) for(int i=(a);i<=(b);++i)
3  using namespace std;
4
5  #define maxn 100017//元素总个数
6  #define elif else if
7  #define LL long long
8
9  struct SegmentTree{
10     #define ls (rt<<1)//lson,左子树
11     #define rs (rt<<1|1)
12     LL Sum[maxn<<2],Add[maxn<<2];//Sum求和,Add为懒惰标记
13     LL A[maxn],n;//存原数组数据下标[1,n]
14
15     void PushUp(int rt){
16         //更新节点信息 , 这里是求和
17         Sum[rt]=Sum[ls]+Sum[rs];
18     }
19
20     void Build(int l,int r,int rt){
21         //l,r:当前节点区间, rt:当前节点编号
22         if(l==r) { //若到达叶节点
23             Sum[rt]=A[l]; //储存数组值
24             return;
25         }
26         int m=(l+r)>>1;
27         //左右递归
28         Build(l,m,ls);
29         Build(m+1,r,rs);
30         PushUp(rt);
31     }
32
33     void Update(int L,LL C,int l,int r,int rt){
34         //l,r:当前节点区间,rt:当前节点编号
35         if(l==r){ //到叶节点, 修改
36             Sum[rt]+=C;
37             return;
38         }
39         int m=(l+r)>>1;
40         //根据条件判断往左子树调用还是往右
41         if(L<=m) Update(L,C,l,m,ls);
42         else Update(L,C,m+1,r,rs);
43         PushUp(rt); //子节点更新了, 所以本节点也需要更新信息
44     }
```

整

```
45 void PushDown(int rt,int ln,int rn){
46     //ln,rn为左子树, 右子树的数字数量。
47     if(Add[rt]){
48         //下推标记
49         Add[ls]+=Add[rt];
50         Add[rs]+=Add[rt];
51         //修改子节点的Sum使之与对应的Add相对应
52         Sum[ls]+=Add[rt]*ln;
53         Sum[rs]+=Add[rt]*rn;
54         //清除本节点标记
55         Add[rt]=0;
56     }
57 }
58
59 void Update(int L,int R,LL C,int l,int r,int rt){
60     //L,R:操作区间, l,r:当前节点区间, rt:当前节点编号
61     if(L<=l and r<=R){//如果本区间完全在操作区间[L,R]以内
62         Sum[rt]+=C*(r-l+1);//更新数字和, 向上保持正确
63         Add[rt]+=C;//增加Add标记, 表示本区间的Sum正确, 子区间的Sum仍需要根据Add的值来调
64         整
65         return ;
66     }
67     int mid=(l+r)>>1;
68     PushDown(rt,mid-l+1,r-mid);//下推标记
69     //这里判断左右子树跟[L,R]有无交集, 有交集才递归
70     if(L<=mid) Update(L,R,C,l,mid,ls);
71     if(R>mid) Update(L,R,C,mid+1,r,rs);
72     PushUp(rt);//更新本节点信息
73 }
74
75 LL Query(int L,int R,int l,int r,int rt){
76     //L,R:操作区间,l,r:当前节点区间, rt:当前节点编号
77     if(L<=l and r<=R){
78         //在区间内, 直接返回
79         return Sum[rt];
80     }
81     int mid=(l+r)>>1;
82     //下推标记, 否则Sum可能不正确
83     PushDown(rt,mid-l+1,r-mid);
84
85     //累计答案
86     LL ANS=0;
87     if(L<=mid) ANS+=Query(L,R,l,mid,ls);
88     if(R>mid) ANS+=Query(L,R,mid+1,r,rs);
89     return ANS;
90 }
91 };
92
```

```

93  int main(){
94      ios_base::sync_with_stdio(0);
95      cin.tie(0);
96      SegmentTree st;
97      int n,m;
98      cin>>n>>m;
99      FOR(i,1,n)
100         cin>>st.A[i];
101      st.Build(1,n,1);
102      int o,l,r,k;
103      FOR(i,1,m){
104         cin>>o;
105         if(o==1){
106             cin>>l>>r>>k; //[l,r]+=k
107             st.Update(l,r,k,1,n,1);
108         }
109         elif(o==2){
110             cin>>l>>r;
111             cout<<st.Query(l,r,1,n,1)<<endl; //sum[l->r]
112         }
113     }
114     return 0;
115 }

```

区间最大值

```

1  #include<bits/stdc++.h>
2  #define FOR(i,a,b) for(int i=(a);i<=(b);++i)
3  using namespace std;
4
5  const int maxn=5e5+7;
6
7  int a[maxn];
8  struct node{
9      int l,r;
10     int ma,cnt;
11 }t[maxn];
12
13 int ans1,n,m;
14
15 void push_up(int p){
16     t[p].ma=max(t[2*p].ma,t[2*p+1].ma);
17     if(t[2*p].ma==t[2*p+1].ma) t[p].cnt=t[2*p].cnt+t[2*p+1].cnt;
18     else if(t[2*p].ma>t[2*p+1].ma) t[p].cnt=t[2*p].cnt;
19     else t[p].cnt=t[2*p+1].cnt;
20 }
21
22 void build(int p,int l,int r){

```



```

23     t[p].l=l;t[p].r=r;t[p].cnt=0;
24     if(l==r){t[p].ma=a[l];t[p].cnt=1;return;}
25     int mid=(l+r)/2;
26     build(2*p,l,mid);
27     build(2*p+1,mid+1,r);
28     push_up(p);
29 }
30
31 void update(int p,int x,int k){
32     if(t[p].l==t[p].r){t[p].ma=k;t[p].cnt=1;return;}
33     int mid=(t[p].l+t[p].r)/2;
34     if(x<=mid) update(2*p,x,k);
35     else update(2*p+1,x,k);
36     push_up(p);
37 }
38
39 void query(int p,int l,int r){
40     int L=t[p].l,R=t[p].r;
41     if(L>=l and R<=r){ans1=max(ans1,t[p].ma);return;}
42     int mid=(L+R)/2;
43     if(l<=mid) query(2*p,l,r);
44     if(r>mid) query(2*p+1,l,r);
45     push_up(p);
46 }
47
48 int main(){
49     cin.tie(0)->sync_with_stdio(0);
50     cin>>n>>m;
51     FOR(i,1,n){
52         cin>>a[i];
53     }
54     build(1,1,n);
55     int x,y;
56     FOR(i,1,m){
57         cin>>x>>y;
58         ans1=0;
59         query(1,x,y);
60         cout<<ans1<<'\\n';
61     }
62     return 0;
63 }

```