SOLVE LATER

Monk Celebrating Checkpoint

Attempted by: **260** / Accuracy: **51%** / Maximum Score: **30** /
4.0 ★★★★☆ 7 Votes

Tag(s): Data Structures, Medium, Stack, approved

There are a variety of approaches to solve this problem. 3 of them broadly include :

1. Finding the previous/next greater/smaller element of each array element using a stack.
2. Binary search over any RMQ data structure
3. Divide and conquer + any data structure supporting RMQ.

As stacks have already been considered in previous list of Code Monk topics, we shall consider this approach here

It is possible to find the previous/next greater/smaller element of each array element in $O(N)$ using a stack. Using this information, we can find for each array element the number of sub arrays it appears in as the maximum/minimum element. The pseudo code to find the next greater element of each array element would be something like this :

```
        for(int i=1;i<=n;i++)
        {
            if(i==0)
                st.push(i);
            else
            {
                while(!st.empty() and
a[i] > a[st.top()])
                {
```

BEST SUBMISSIONS

LANGUAGE: C

TIME (sec)	MEMORY (KiB)
2.62563	64

by LIT2016003

VIEW BEST SUBMISSION VIEW ALL SUBMISSION

CONTRIBUTOR

AUTHOR
Prateek Garg

TESTER
Anand Jaisingh

THIS PROBLEM WAS ASKED IN

CHALLENGE NAME
CodeMonk (Checkpoint - I)

SOCIAL SHARE

```
                    mar[st.top()] = i;
                    st.pop();
                }
                st.push(i);
            }
        }
        while(!st.empty())
        {
            mar[st.top()]=n+1;
            st.pop();
        }
```

One observation to make here is that we can performing the operations after computing the above for each array element. The maximum sum can be computed on the basis of the info obtained above.

For each array element, we know the number of sub arrays it appears in as the max/min. Initially we need to calculate the summation of the difference between the max and min of each sub array. Lets call this variable ans. If an element x appears in i sub arrays as the maximum and j sub arrays as the minimum, then on performing an operation on this element, it shall contribute $i - j$ to the variable ans.

Thus, we need to create an array $arr[]$ where $arr[i]$ indicates the difference between the number of sub arrays in which element i appears as the max-the number of sub arrays in which element i appear as the min.

We can then sort this array and add to answer the highest value x elements from $arr[]$. However, there is a corner case here when $arr[i] < 0$. This case needs to be handled separately.

Tester Solution :

```
// divide and conquer approach with
segment tree...

import java.io.*;
import java.util.*;
public final class sub_max
{
    static BufferedReader br;
    static FastScanner sc;
    static PrintWriter out;
    static Random rnd=new Random();
    static List<Pair> list=new
```

```
ArrayList<Pair>();
    static int[] a,tree1,tree2;
    static int n,x,k;
    static long[] s1,s2;

    static long solve1(int s,int e)
    {
        if(s>e)
        {
            return 0;
        }
        if(s==e)
        {
            s1[s]=1;return a[s];
        }
        else
        {
            int
max_pos=getMax(1,0,n-1,s,e);long
curr1=a[max_pos],curr2=((long)(max_pos-s+1
))*((long)(e-max_pos+1));s1[max_pos]=curr2
;
            return
(curr1*curr2)+solve1(s,max_pos-1)+solve1(m
ax_pos+1,e);
        }
    }

    static long solve2(int s,int e)
    {
        if(s>e)
        {
            return 0;
        }
        if(s==e)
        {
            s2[s]=1;return a[s];
        }
        else
        {
            int
min_pos=getMin(1,0,n-1,s,e);long
curr1=a[min_pos],curr2=((long)(min_pos-s+1
))*((long)(e-min_pos+1));s2[min_pos]=curr2
;
            return
(curr1*curr2)+solve2(s,min_pos-1)+solve2(m
in_pos+1,e);
        }
    }
```

```
    static void build1(int node,int s,int
e)
    {
        if(s>e)
        {
            return;
        }
        if(s==e)
        {
            tree1[node]=s;
        }
        else
        {
            int mid=(s+e)>>1;

build1(node<<1,s,mid);build1(node<<1|1,mid
+1,e);

tree1[node]=resolve1(tree1[node<<1],tree1[
node<<1|1]);
        }
    }

    static void build2(int node,int s,int
e)
    {
        if(s>e)
        {
            return;
        }
        if(s==e)
        {
            tree2[node]=s;
        }
        else
        {
            int mid=(s+e)>>1;

build2(node<<1,s,mid);build2(node<<1|1,mid
+1,e);

tree2[node]=resolve2(tree2[node<<1],tree2[
node<<1|1]);
        }
    }

    static int getMax(int node,int s,int
e,int l,int r)
    {
```

```
        if(s>e || l>e || r<s)
        {
            return -1;
        }
        if(l<=s && r>=e)
        {
            return tree1[node];
        }
        else
        {
            int mid=(s+e)>>1;
            int
q1=getMax(node<<1,s,mid,l,r),q2=getMax(nod
e<<1|1,mid+1,e,l,r);
            return resolve1(q1,q2);
        }
    }

    static int getMin(int node,int s,int
e,int l,int r)
    {
        if(s>e || l>e || r<s)
        {
            return -1;
        }
        if(l<=s && r>=e)
        {
            return tree2[node];
        }
        else
        {
            int mid=(s+e)>>1;
            int
q1=getMin(node<<1,s,mid,l,r),q2=getMin(nod
e<<1|1,mid+1,e,l,r);
            return resolve2(q1,q2);
        }
    }

    static int resolve1(int idx1,int idx2)
    {
        if(idx1==-1 || idx2==-1)
        {
            return Math.max(idx1,idx2);
        }
        return
(a[idx1]>a[idx2]?idx1:idx2);
    }

    static int resolve2(int idx1,int idx2)
```

```
    {
        if(idx1==-1 || idx2==-1)
        {
            return Math.max(idx1,idx2);
        }
        return
(a[idx1]<a[idx2]?idx1:idx2);
    }

    static void brute(int[] a)
    {
        long sum1=0,sum2=0;
        for(int i=0;i<n;i++)
        {
            int max=-1;
            for(int j=i;j<n;j++)
            {

max=Math.max(max,a[j]);sum1+=max;
            }
        }
        for(int i=0;i<n;i++)
        {
            long min=Long.MAX_VALUE;
            for(int j=i;j<n;j++)
            {

min=Math.min(min,a[j]);sum2+=min;
            }
        }
        out.println(sum1+" "+sum2+"
"+(sum1-sum2));
    }

    static void init(String curr) throws
Exception
    {
        br=new BufferedReader(new
FileReader(new File("in"+curr+".txt")));
        sc=new FastScanner(br);
        out=new PrintWriter(new
FileWriter("out"+curr+".txt"));
    }

    public static void run() throws
Exception
    {

init("00");n=sc.nextInt();x=sc.nextInt();k
=sc.nextInt();a=new int[n];tree1=new
```

```
int[4*n];tree2=new int[4*n];s1=new
long[n];s2=new long[n];
        for(int i=0;i<n;i++)
        {
            a[i]=sc.nextInt();
        }

build1(1,0,n-1);build2(1,0,n-1);long
val1=solve1(0,n-1),val2=solve2(0,n-1);
        for(int i=0;i<n;i++)
        {
            if(s1[i]-s2[i]>0)
            {
                list.add(new
Pair(i,s1[i]-s2[i]));
            }
        }
        Collections.sort(list);
        for(int i=list.size()-1,j=x;j>0
&& i>=0;j--,i--)
        {

val1+=list.get(i).cnt;a[list.get(i).idx]++
;
        }
        out.println(val1-val2);
        //brute(a);
        out.close();
    }

    public static void main(String args[])
throws Exception
    {
        new Thread(null,new Runnable(){
            public void run()
            {
                try
                {
                    new sub_max().run();
                }
                catch(Exception e)
                {

                }
            }
        },"1",1<<26).start();
    }
}
class Pair implements Comparable<Pair>
{
```

```
    int idx;long cnt;
    public Pair(int idx,long cnt)
    {
        this.idx=idx;this.cnt=cnt;
    }
    public int compareTo(Pair x)
    {
        return
Long.compare(this.cnt,x.cnt);
    }
}
class FastScanner
{
    BufferedReader in;
    StringTokenizer st;

    public FastScanner(BufferedReader in)
{
        this.in = in;
    }

    public String nextToken() throws
Exception {
        while (st == null ||
!st.hasMoreTokens()) {
            st = new
StringTokenizer(in.readLine());
        }
        return st.nextToken();
    }

    public String next() throws Exception
{
        return nextToken().toString();
    }

    public int nextInt() throws Exception
{
        return
Integer.parseInt(nextToken());
    }

    public long nextLong() throws
Exception {
        return
Long.parseLong(nextToken());
    }

    public double nextDouble() throws
Exception {
```

```
        return
Double.parseDouble(nextToken());
    }
}
```

Author Solution by Prateek Garg

```
#include <bits/stdc++.h>
#define ll long long

using namespace std;
const int ma = 1e5+5;
ll mal[ma], mar[ma] , mil[ma],
mir[ma], ar[ma], a[ma],
mins[ma],mp[ma], fre[1000005];
int n,x,k;
stack <int> st;
void great(bool f)
{
        if(f)
        {
                for(int i=1;i<=n;i++)
                {
                        if(i==0)

st.push(i);
                        else
                        {

while(!st.empty() and a[i] >
a[st.top()])
                                {

mar[st.top()] = i;

st.pop();
                                }

st.push(i);
                        }
                }
                while(!st.empty())
                {

mar[st.top()]=n+1;
                        st.pop();
```

```
                }
        }
        else
        {
                for(int i=n;i>=1;i--)
                {
                        if(i==n)
st.push(i);
                        else
                        {
while(!st.empty() and a[i] >
a[st.top()])
                                {
mal[st.top()] = i;
st.pop();
                                }
st.push(i);
                        }
                }
                while(!st.empty())
                {
mal[st.top()]=0;
                        st.pop();
                }
        }
}
void smal(bool f)
{
        if(f)
        {
                for(int i=1;i<=n;i++)
                {
                        if(i==0)
st.push(i);
                        else
                        {
while(!st.empty() and a[i] <
a[st.top()])
                                {
mir[st.top()] = i;
```

```
    st.pop();
70.                                 }
71.
    st.push(i);
72.                         }
73.                 }
74.                 while(!st.empty())
75.                 {
76.
    mir[st.top()]=n+1;
77.                         st.pop();
78.                 }
79.         }
80.         else
81.         {
82.                 for(int i=n;i>=1;i--)
83.                 {
84.                         if(i==n)
85.
    st.push(i);
86.                         else
87.                         {
88.
    while(!st.empty() and a[i] <
    a[st.top()])
89.                                 {
90.
    mil[st.top()] = i;
91.
    st.pop();
92.                                 }
93.
    st.push(i);
94.                         }
95.                 }
96.                 while(!st.empty())
97.                 {
98.
    mil[st.top()]=0;
99.                         st.pop();
100.                }
101.        }
102. }
103. bool cmp(ll a, ll b)
104. {
105.        return a > b;
106. }
107. int main(int argc, char* argv[])
108. {
109.        /*freopen(argv[1],"r",stdin);
```

```
110.
    freopen(argv[2],"w",stdout);*/
111.        cin>>n>>x>>k;
112.        assert(1<=n and n<=1e5);
113.        assert(1<=x and x<=n);
114.        for(int i=1;i<=n;i++)
115.
    cin>>a[i],fre[a[i]]++,assert(0<=a[i]
    and a[i]<=1e6);
116.        /*for(int i=0;i<n;i++)
117.        {
118.                for(int
    j=i+1;j<n;j++)
119.                {
120.
    //cout<<a[i]<<" "<<a[j]<<endl;
121.
    assert(abs(a[i]-a[j])>=2);
122.                }
123.        }*/
124.                for(int
    i=0;i<=1e6;i++)
125.
    assert(fre[i]<=1);
126.        great(true);
127.        great(false);
128.        smal(true);
129.        smal(false);
130.
131.        ll mas=0, mis=0;
132.        for(int i=1;i<=n;i++)
133.        {
134.                mp[i] =
    ((mar[i]-i)*(i-mal[i]))-((mir[i]-i)*(
    i-mil[i]));
135.                mas+=
    ((mar[i]-i)*(i-mal[i]))*a[i];
136.                mis+=
    (mir[i]-i)*(i-mil[i])*a[i];
137.        }
138.        sort(mp+1,mp+n+1, cmp);
139.        int c=0;
140.
141.        ll sum = mas-mis;
142.        for(int i=1;i<=x;i++)
143.        {
144.                        if(mp[i]>0)
145.
    sum+=mp[i];
146.        }
```

```
147.
148.            cout<<sum<<endl;
149.
150.
151.
152.            return 0;
153.    }
```