SOLVE LATER

Mancunian And Fantabulous Pairs

Attempted by: **791** / Accuracy: **50%** / Maximum Score: **30** /
8 Votes

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

In given array A, suppose for an element at index curr (A[curr]), first element greater than it after it is A[next] and first element greater than it and before it is A[prev]. Observe that for all subarray starting from any index in range [prev+1,curr] and ending at index next, A[curr] is second maximum and A[next] is maximum. Which generate (curr-prev+1) pairs in total with difference of (next-curr+1) in maximum and second maximum.

If we get next and prev greater element for every element in an array, and keep track of maximum number of pairs possible for any difference (of maximum and second maximum). We will just need to add all these numbers (for each difference).

Now only left job is to get greater element after (and before, which is similar problem) any element. This can be naively done in **O(n)** by traversing whole array, which will make our algorithm **O(n^2)**, which will only generate partial points.

This can easily be done using a stack. Traverse from the starting element in an array, keeping track of all numbers in the stack in decreasing order. After arriving at any number, pop all elements from stack which are less than current element to get location of number bigger than it and push current element on it. This generates required value for all numbers in the array. Making our algorithm run in **O(n)**.

BEST SUBMISSIONS

LANGUAGE: C

TIME (sec) 1.07896

MEMORY (KiB) 23628

by Hong Han

VIEW BEST SUBMISSION VIEW ALL SUBMISSION

CONTRIBUTOR

AUTHOR
Satyaki Upadhyay

TESTER
Mitesh Agrawal

THIS PROBLEM WAS ASKED IN

CHALLENGE NAME
April Easy '17

SOCIAL SHARE

IS THIS EDITORIAL HELPFUL?

Yes, it's helpful

No, it's not helpful

37 developer(s) found this editorial helpful.

Author Solution by Satyaki Upadhyay

```
1. //satyaki3794
2. #include <bits/stdc++.h>
3. #define ff first
4. #define ss second
5. #define pb push_back
6. #define MOD (1000000007LL)
7. #define LEFT(n) (2*(n))
8. #define RIGHT(n) (2*(n)+1)
9.
10. using namespace std;
11. typedef long long ll;
12. typedef pair<int, int> ii;
13. typedef pair<int, ii> iii;
14.
15. ll pwr(ll base, ll p, ll mod = MOD){
16. ll ans =
    1;while(p){if(p&1)ans=(ans*base)%mod;
    base=(base*base)%mod;p/=2;}return
    ans;
17. }
18.
19.
20.
21. const int N = 1000*1000+5;
22. int n, arr[N], next_right[N],
    maxlen[N], next_left[N];
23. set<int> distinct_;
24.
25.
26.
27. int main(){
28.
29.     ios_base::sync_with_stdio(0);
30.     cin.tie(0);
31.
```

```
    cin>>n;
    assert(n >= 1 && n <= 1000*1000);
    for(int i=1;i<=n;i++){
        cin>>arr[i];
        assert(arr[i] >= 1 && arr[i] <= 1000*1000*1000);
        distinct_.insert(arr[i]);
    }
    assert((int)distinct_.size() == n);

    stack<int> st;
    for(int i=n;i>=1;i--){
        next_right[i] = -1;
        while(!st.empty() && arr[st.top()] < arr[i])
                st.pop();
        if(!st.empty())
                next_right[i] = st.top();
        st.push(i);
    }

    while(!st.empty())
        st.pop();
    for(int i=1;i<=n;i++){
        next_left[i] = 0;
        while(!st.empty() && arr[st.top()] < arr[i])
                st.pop();
        if(!st.empty())
                next_left[i] = st.top();
        st.push(i);
    }

    for(int i=1;i<=n;i++){
        if(next_right[i] != -1)

maxlen[next_right[i]-i] = max(maxlen[next_right[i]-i], i - next_left[i]);
    }

    ll ans = 0;
    for(int i=1;i<=n;i++)
        ans += maxlen[i];
    cout<<ans;
    return 0;
}
```

```
73.
74.
75.
76.
77.
78.
79.
80.
```

Tester Solution by Mitesh Agrawal

```
1. //Mitesh Agrawal
2. #include <bits/stdc++.h>
3. using namespace std;
4.
5. #define MAXN 1000005
6. #define ii pair<int,int>
7. #define ff first
8. #define ss second
9. #define ll long long
10.
11. int arr[MAXN];
12. int nextBig[MAXN];
13. int prevBig[MAXN];
14. int maxi[MAXN];
15. int n;
16. ll ans;
17.
18. void makeNext(){
19.         int i;
20.         stack<ii> s;
21.         for(i=n-1;i>=0;i--){
22.                 nextBig[i] = i;
23.                 while(!s.empty() &&
    s.top().ff < arr[i])
24.                         s.pop();
25.                 if(!s.empty())
26.                         nextBig[i] =
    s.top().ss;
27.                 s.push(ii(arr[i],i));
28.         }
29. }
30.
31. void makePrev(){
32.         int i;
33.         stack<ii> s;
34.         for(i=0;i<n;i++){
35.                 prevBig[i] = -1;
36.                 while(!s.empty() &&
```

```
    s.top().ff < arr[i])
37.                         s.pop();
38.                 if(!s.empty())
39.                         prevBig[i] =
    s.top().ss;
40.                 s.push(ii(arr[i],i));
41.         }
42. }
43.
44. int main ()
45. {
46.         int i,j;
47.         scanf("%d",&n);
48.         for(i=0;i<n;i++)
49.                 scanf("%d",&arr[i]);
50.
51.         makePrev();
52.         makeNext();
53.
54.         for(i=0;i<n;i++)
55.                 if(nextBig[i]!=i)
56.
    maxi[nextBig[i]-i] =
    max(maxi[nextBig[i]-i], i -
    prevBig[i]);
57.
58.         for(i=0;i<n;i++)
59.                 ans+=maxi[i];
60.
61.         printf("%lld\n",ans);
62.     return 0;
63. }
64.
65.
```

COMMENTS (10) SORT BY: Relevance

Join Discussion...

Cancel Post

Ashwani Gautam Edited 9 months ago

The problem you did using stack, what is it called ?