Leaders in an array

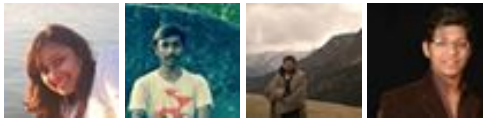
Write a program to print all the LEADERS in the array. An element is leader if it is greater than all the elements to its right side. And the rightmost element is always a leader. For example int the array {16, 17, 4, 3, 5, 2}, leaders are 17, 5 and 2.

Let the input array be arr[] and size of the array be *size*.

Method 1 (Simple)
Use two loops. The outer loop runs from 0 to size – 1 and one by one picks all elements from left to right. The inner loop compares the picked element to all the elements to its right side. If the picked element is greater than all the elements to its right side, then the picked element is the leader.

```
/*Function to print leaders in an array */
void printLeaders(int arr[], int size)
{
  int i, j;

  for (i = 0; i < size; i++)
  {
    for (j = i+1; j < size; j++)
    {
        if(arr[i] <= arr[j])
          break;
    }
    if(j == size) // the loop didn't break
    {
        printf("%d ", arr[i]);
    }
  }
}
```

Interview Experiences
Advanced Data Structures
Dynamic Programming
Greedy Algorithms
Backtracking
Pattern Searching
Divide & Conquer
Mathematical Algorithms
Recursion
Geometric Algorithms

```
/*Driver program to test above function*/
int main()
{
  int arr[] = {16, 17, 4, 3, 5, 2};
  printLeaders(arr, 6);
  getchar();
}
// Output:  17 5 2
```

Time Complexity: O(n*n)

Method 2 (Scan from right)
Scan all the elements from right to left in array and keep track of maximum till now. When maximum changes it's value, print it.

```
/*Function to print leaders in an array */
void printLeaders(int arr[], int size)
{
  int max_from_right =  arr[size-1];
  int i;

  /* Rightmost element is always leader */
  printf("%d ", max_from_right);

  for(i = size-2; i >= 0; i--)
  {
    if(max_from_right < arr[i])
    {
       printf("%d ", arr[i]);
       max_from_right = arr[i];
    }
  }
}

/*Driver program to test above function*/
int main()
{
  int arr[] = {16, 17, 4, 3, 5, 2};
  printLeaders(arr, 6);
  getchar();
}
// Output:  2 5 17
```

Geometric Algorithms

Popular Posts

All permutations of a given string
Memory Layout of C Programs
Understanding "extern" keyword in C
Median of two sorted arrays
Tree traversal without recursion and without stack!
Structure Member Alignment, Padding and Data Packing
Intersection point of two Linked Lists
Lowest Common Ancestor in a BST.
Check if a binary tree is BST or not
Sorted Linked List to Balanced BST

Time Complexity: O(n)

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

Related Tpoics:

- Remove minimum elements from either side such that 2*min becomes more than max
- Divide and Conquer | Set 6 (Search in a Row-wise and Column-wise Sorted 2D Array)
- Bucket Sort
- Kth smallest element in a row-wise and column-wise sorted 2D array | Set 1
- Find the number of zeroes
- Find if there is a subarray with 0 sum
- Divide and Conquer | Set 5 (Strassen's Matrix Multiplication)
- Count all possible groups of size 2 or 3 that have sum as multiple of 3

Writing code in comment? Please use **ideone.com** and share the link here.

Join the discussion…

wrestler • 4 days ago

def printLeadingElement(arr):

max=-999

for x in arr[::-1]:

if x>max:

print(x,end=" ")

max=x

print()

• Reply • Share ›

HRISHIKESH • 2 months ago

//another method (using recursion)

#include <iostream>
using namespace std;

int printLeader (int arr[],int str ,int end) {
if (str==end) {

Recent Comments

Aman Hi, Why arent we checking for conditions...

Write a C program to Delete a Tree. · 22 minutes ago

kzs please provide solution for the problem...

Backtracking | Set 2 (Rat in a Maze) · 25 minutes ago

Sanjay Agarwal bool tree::Root_to_leaf_path_given_sum(tree...

Root to leaf path sum equal to a given number · 50 minutes ago

GOPI GOPINATH @admin Highlight this sentence "We can easily...

Count trailing zeroes in factorial of a number · 52 minutes ago

newCoder3006 If the array contains negative numbers also. We...

Find subarray with given sum · 1 hour ago

newCoder3006 Code without using while loop. We can do it...

Find subarray with given sum · 1 hour ago

```
if (str==end) {
cout<< arr[str] <<" ";
return arr[str];
}
int temp =printLeader( arr,str+1,end);
if (arr[str]>temp ) {
cout<< arr [str]<< " ";
return arr[str];
}
else return temp;
}
int main() {
int arr[] = {4,3,7,12,6,67,5,45,34,35,2,8};
printLeader(arr,0,11);
return 0;
}
```

• Reply • Share ›

dark_night • 9 months ago

We can use stack for left to right pass..

For each element .. before pushing it pop all the elements that are less than it left in it

3 • Reply • Share ›

bharath reddy dark_night • 7 months ago

But this takes O(n) extra space

4 • Reply • Share ›

geekyboy • a year ago

@geeksforgeeks
in the first method shouldn't it be

if(arr[i] <= arr[j])
break;

rather than
if(arr[i] < arr[j])
break;

because the condition says a leader must be greater than all the elements to i
condtion is violated.

If I am wrong do inform.
Any help and suggestions are welcomed

• Reply • Share ›

geekyboy ➚ geekyboy • a year ago

for eg
{12,10,12,8,7,6}

answer should be 12 8 7 6
rather than 12 12 8 7 6

• Reply • Share ›

GeeksforGeeks ➚ geekyboy • a year ago

Thanks for pointing this out. We have updated the post. Keep it

1 • Reply • Share ›

VD • 2 years ago

{
This can be also done will the help of stack.
push the first element in the stack // s-> 16
push another element,
if (upcoming element is smaller then the element of the top of the stack)
Push(element);

else
pop() till you find element bigger then the upcoming element.
}
left over elements in the stack will be leaders !!! :)
ex:
s-> 16 //pop 16 push 17
s-> 17 //push 4
s->17 ,4. // push 3
s->17 ,4,3, // pop till you element bigger than 5 then push
s-> 17,5// push 2
s->17,5,2
all elements left in the stack are leaders :)

• Reply • Share ›

shek8034 ➦ VD • 11 months ago
Good logic... Never thought about it :P
But space complexity is more.

1 • Reply • Share ›

Mayautobot ➦ VD • 2 years ago
Nice one!
But space complexity is more.

• Reply • Share ›

kg1020 • 2 years ago
a function in C++.

```
 void leaders(int ar[],int n)
{
    cout<<"\nLeaders : ";
    int lastLeader = ar[n-1];
    cout<<lastLeader<<'\t';
```

```
    for(int i=n-2; i>=0; i--)
    {
        if(ar[i] > lastLeader)
        {
            lastLeader = ar[i];
            cout<<lastLeader<<'\t';
        }
    }
}
```

Reply • Share ›

Amol Agarwal • 3 years ago

This will print 2,2,5,17. an extra 2.

Reply • Share ›

GeeksforGeeks • 4 years ago

@Rajeev & kp101090:

Thanks for pointing this optimization. We have made the suggested changes.

Reply • Share ›

kp101090 • 4 years ago

Agree with Rajeev...
for loop should be
for(i=size-2;i>=0;--i)

Reply • Share ›

Rajeev • 4 years ago

In the for loop inspite of size-1, we can start from size-2. I think the first compa

Reply • Share ›

ravikant • 4 years ago

int i;

is missing :P

• Reply • Share ›

GeeksforGeeks ravikant • 4 years ago

@ravikant: Thanks for pointing this out. We have added the line.

• Reply • Share ›

Subscribe Add Disqus to your site