**Partition Point in the Array**

Show Topic Tags   

Given an unsorted array of integers. Find an element such that all the elements to its left are smaller and to its right are greater. Print -1 if no such element exists.Note that there can be more than one such element. In that case print the first such number occurring in the array.

**Input:**  
The first line of input contains an integer T denoting the number of test cases. Each test case contains an integer n which denotes the number of elements in the array a[]. Next line contains space separated n elements in the array a[].  
  
**Output:**  
Print an integer which is the required partition point.(-1 if no such partition exists)  
  
**Constraints:**  
1<=T<=100  
1<=n<=1000  
1<=a[i]<=10000  
  
**Example:  
Input:**  
2  
7  
4 3 2 5 8 6 7  
7  
5 6 2 8 10 9 8  
  
**Output:**  
5  
-1

\*\*For More Examples Use Expected Output\*\*

Contributor: Shashwat Jain

<http://practice.geeksforgeeks.org/problems/partition-point-in-the-array/0>

/\*

\* To change this template, choose Tools | Templates

\* and open the template in the editor.

\*/

package javaapplication250;

import java.io.\*;

import java.math.\*;

import java.util.\*;

/\*\*

\*

\* @author Administrador

\*/

public class JavaApplication250 {

public static void main(String[] args) throws IOException {

// TODO code application logic here

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

int t = Integer.parseInt(br.readLine());

while(t-- > 0) {

int n = Integer.parseInt(br.readLine().trim());

String[] input = br.readLine().trim().split(" ");

int[] a = new int[n];

for(int i =0;i<n; i++) {

a[i] = Integer.parseInt(input[i]);

}

int ans = -1;

for (int i = 0; i < n; i++)

{

int max = 0;

for (int j = 0; j < i; j++)

{

if (a[j] > max)

{

max = a[j];

}

}

int min = Integer.MAX\_VALUE;

for (int j = i + 1; j < a.length; j++)

{

if (a[j] < min)

{

min = a[j];

}

}

if (a[i] > max && a[i] < min)

{

ans = a[i];

break;

}

}

System.out.println(ans);

}

}

}

-------EDITORIAL------------

<http://www.geeksforgeeks.org/find-a-partition-point-in-array/>

**Simple solution** takes O(n2). Idea is to pick each array element one by one and for each element we have to check it is greater than all the elements to its left side and smaller than all the elements to its right side.

Below c++ implementation of above idea :

|  |
| --- |
| // Simple C++ program to find a partition point in  // an array  #include<bits/stdc++.h>  using namespace std;    // Prints an element such than all elements on left  // are smaller and all elements on right are greater.  int FindElement( int A[] , int n )  {       // traverse array elements      for (int i=0; i < n ; i++)      {          // If we found that such number          int flag = 0 ;            // check All the elements on its left are smaller          for (int j = 0 ; j < i ; j++ )              if (A[j] >= A[i] )              {                  flag = 1 ;                  break;              }            // check All the elements on its right are Greater          for( int j = i + 1 ; j < n; j++ )              if( A[j] <= A[i] )              {                  flag = 1 ;                  break;              }            // If flag == 0 indicates we found that number          if (flag == 0)              return A[i];      }      return -1;  }    // driver program to test above function  int main()  {      int A[] = {4 ,3 ,2 , 5 , 8 ,6 , 7 }   ;      int n = sizeof(A)/sizeof(A[0]);      cout << FindElement ( A , n ) << endl;      return 0;  } |

Run on IDE

Output:

5

Time complexity: O(n2)

**Efficient solution** take O(n) time.

1. Create an auxiliary array ‘GE[]’. GE[] should store the element which is greater than A[i] and is on left side of A[i].
2. Create an another Auxliary array ‘SE[]’. SE[i] should store the element which is smaller than A[i] and is on right side of A[i].
3. Find element in array that hold condition GE[i-1] < A[i] < SE[i+1].

Below c++ implementation of above idea :

|  |
| --- |
| // Simple C++ program to find a partition point in  // an array  #include<bits/stdc++.h>  using namespace std;    // Returns an element that has all the element to its  // left smaller and to its right greater  int FindElement(int A[], int n)  {      // Create an array 'SE[]' that will store smaller      // element on right side.      int SE[n];        // Create an another array 'GE[]' that will store      // greatest element on left side.      int GE[n];        // initalize first and last index of SE[] , GE[]      GE[0] = A[0];      SE[n-1] = A[n-1];        // store greatest element from left to right      for (int i=1; i<n ; i++)      {          if (GE[i-1] < A[i])              GE[i] = A[i];          else              GE[i] = GE[i-1];      }        // store smallest element from right to left      for (int i = n-2 ; i >= 0 ; i-- )      {          if (A[i] < SE[i+1])              SE[i] = A[i];          else              SE[i] = SE[i+1];      }        // Now find a number which is greater then all      // elements at it's left and smaller the all      // then elements to it's right      for (int j =0 ; j <n ; j++)      {          if ( ( j == 0  && A[j] < SE[j+1] ) ||              ( j == n-1 && A[j] > GE[j-1] ) ||              ( A[j] < SE[j+1] && A[j] > GE[j-1] ) )              return A[j];      }        return -1;  }    // driver program to test above function  int main()  {      int A[] = {4, 3, 2, 5 , 8 ,6 , 7 } ;      int n = sizeof(A)/sizeof(A[0]);      cout << FindElement ( A , n ) << endl;      return 0;  } |

Run on IDE

Output:

5

Time complexity: O(n)  
Auxiliary Space: O(n)

This article is contributed by **Nishant Singh**. If you like GeeksforGeeks and would like to contribute, you can also write an article using [contribute.geeksforgeeks.org](http://www.contribute.geeksforgeeks.org/)or mail your article to contribute@geeksforgeeks.org. See your article appearing on the GeeksforGeeks main page and help other Geeks.

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