Problem | Editorial | Submissions | Comments

Rotate and delete

Difficulty: **Medium** Accuracy: **20.63%** Submissions: **56K+** Points: **4**

Given an array **arr** integers. Assume **sz** to be the initial size of the array. Do the following operations exactly **sz/2** times. In every **kth** (1<= k <= sz/2) operation:

- Right-rotate the array clockwise by **1**.
- Delete the **(n– k + 1)**th element from begin.

Now, Return the first element of the array.

Note: Here n keeps on decreasing with every operation.

Examples:

Input: arr = [1, 2, 3, 4, 5, 6]
Output: 3
Explanation: Rotate the array clockwise i.e. after rotation the array arr = [6, 1, 2, 3, 4, 5] and delete the last element that is 5 that will be arr = [6, 1, 2, 3, 4]. Again rotate the array for the second time and deletes the second last element that is 2 that will be A = [4, 6, 1, 3], doing similar operation when we perform 4th operation, 4th last element does not exist. Then we deletes 1st element ie 1 that will be arr = [3, 6]. So, continuing this procedure the last element in arr is 2. So, the output will be 3.

Input: arr = [1, 2, 3, 4]
Output: 2
Explanation: Rotate the vector clockwise i.e. after rotation the vector arr = [4, 1, 2, 3] and delete the last element that is 3 that will be arr = [4, 1, 2]. After doing all the operations, the output will be 2.

Expected Time Complexity: O(n^2)
Expected Auxiliary Space: O(1)

Constraints:
1 <= arr.size() <= 10^3
1 <= arr[i] <= 10^6

Try more examples

Seen this question in a real interview before ? Yes No

Topic Tags

Related Articles

Report An Issue
If you are facing any issue on this page. Please let us know.

Java (1.8) Start Timer

```java
//{ Driver Code Starts
// Initial Template for Java
import java.io.*;
import java.lang.*;
import java.util.*;


// } Driver Code Ends

class Solution {
    public static int rotateDelete(ArrayList<Integer> arr) {
        int n = arr.size();
        for (int k = 1; k <= n / 2; k++) {
            // Rotate the array by 1 (right rotation)
            int last = arr.remove(arr.size() - 1);  // Remove last element
            arr.add(0, last);                       // Add it to the front

            // Delete the (n - k + 1)th element
            int deleteIndex = arr.size() - k;       // Calculate the index for (n-k+1)th el
            arr.remove(deleteIndex);                // Remove the element at (n-k+1)th posi
        }
        return arr.get(0); // Return the first element
    }
}


//{ Driver Code Starts.

class GFG {
    public static void main(String[] args) throws IOException {
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        int t = Integer.parseInt(br.readLine());
        while (t-- > 0) {
            String line = br.readLine();
```

Custom Input Compile & Run Submit