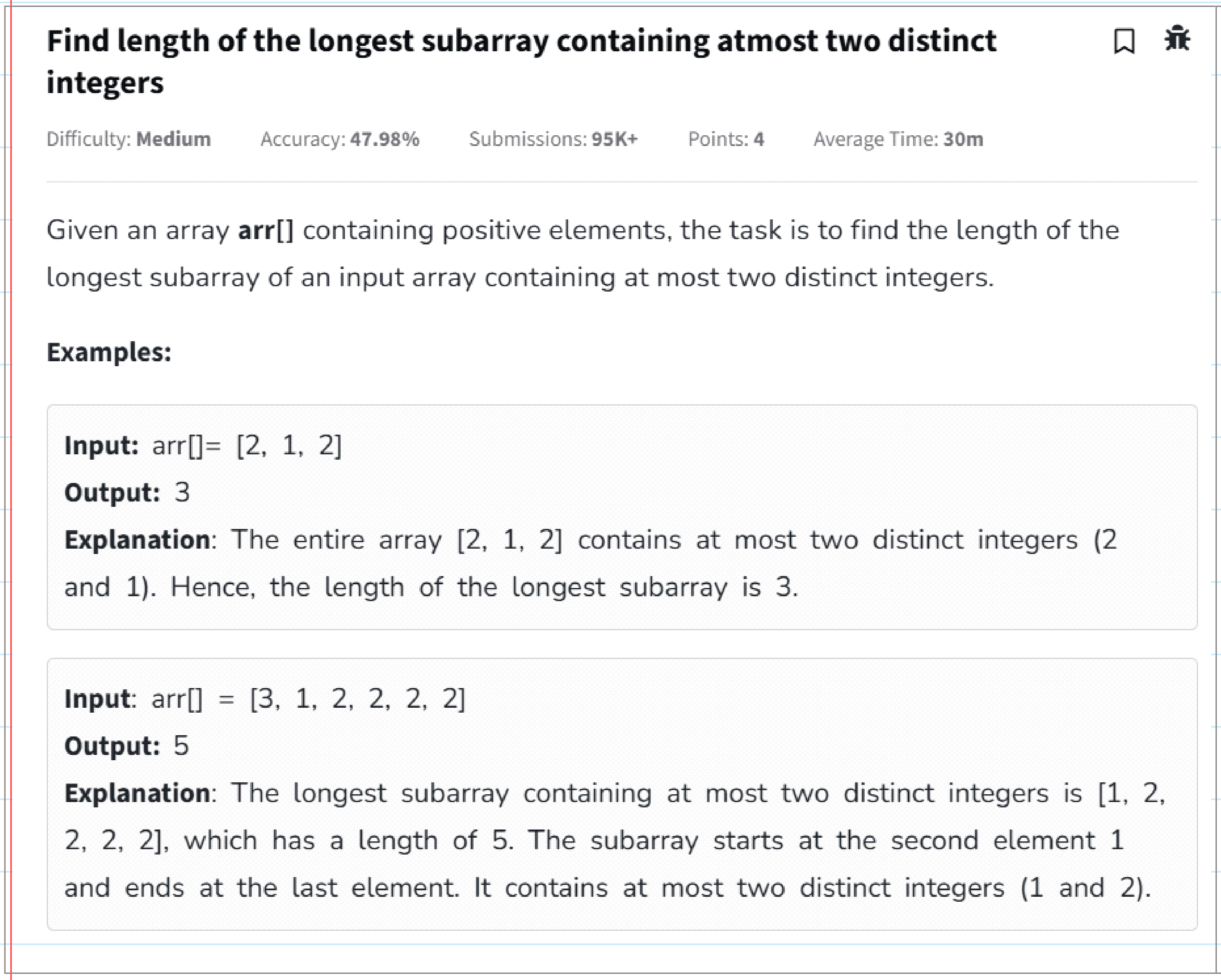
05 March 2025 12:17



Find length of the longest subarray containing atmost two distinct integers

Difficulty: **Medium** Accuracy: **47.98%** Submissions: **95K+** Points: **4** Average Time: **30m**

Given an array **arr[]** containing positive elements, the task is to find the length of the longest subarray of an input array containing at most two distinct integers.

Examples:

Input: arr[]= [2, 1, 2]
Output: 3
Explanation: The entire array [2, 1, 2] contains at most two distinct integers (2 and 1). Hence, the length of the longest subarray is 3.

Input: arr[] = [3, 1, 2, 2, 2, 2]
Output: 5
Explanation: The longest subarray containing at most two distinct integers is [1, 2, 2, 2, 2], which has a length of 5. The subarray starts at the second element 1 and ends at the last element. It contains at most two distinct integers (1 and 2).

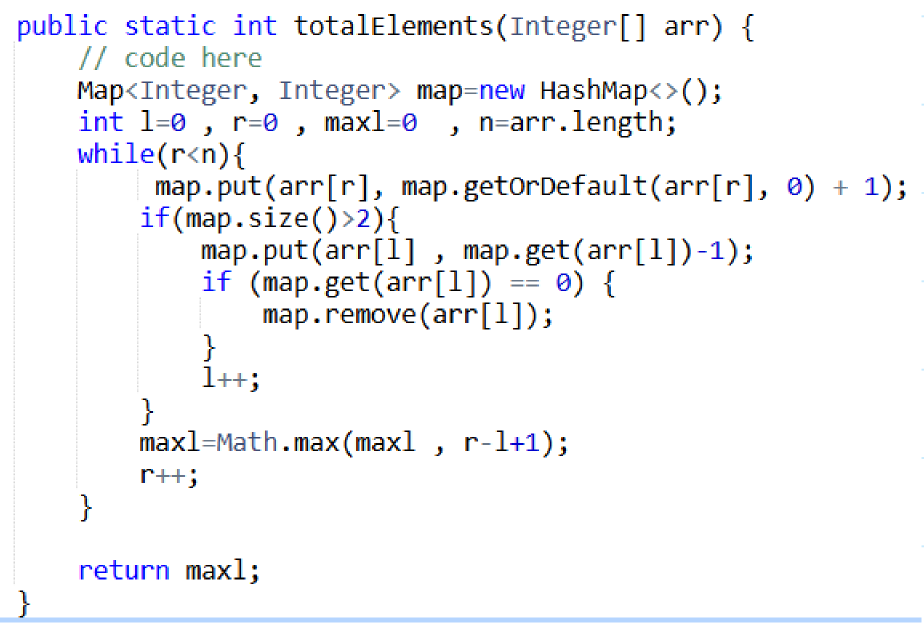
```
class Solution {
    public static int totalElements(Integer[] arr) {
        // code here
        Set<Integer> set =new HashSet<>();
        int n=arr.length , maxl=0;

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

            for(int j=i;j<n;j++){
                set.add(arr[j]);
                if(set.size()>2) break;
                maxl=Math.max(maxl,j-i+1);
            }
            set.clear();
        }

        return maxl;
    }
}
```

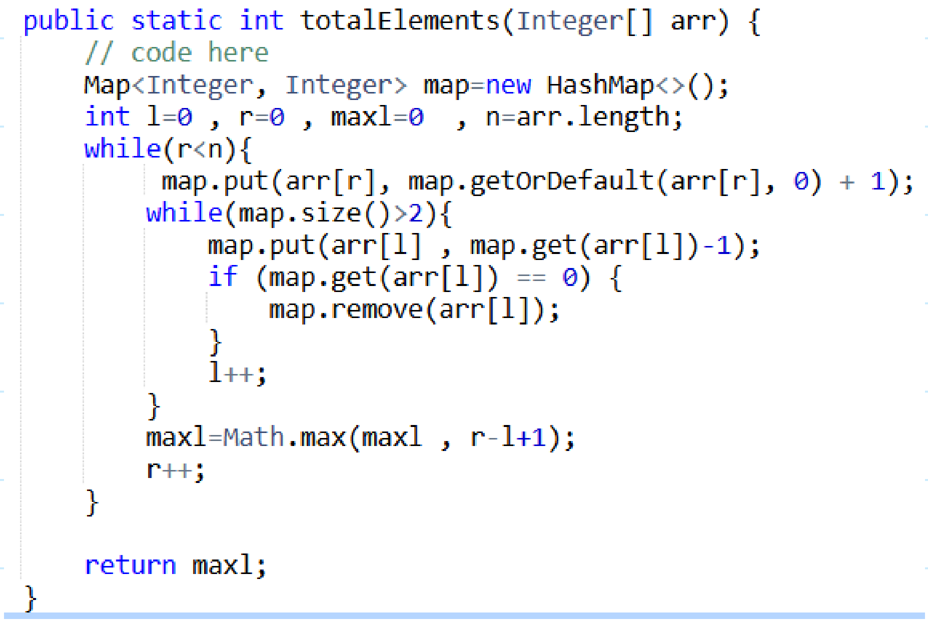
Optimal



```
public static int totalElements(Integer[] arr) {
    // code here
    Map<Integer, Integer> map=new HashMap<>();
    int l=0 , r=0 , maxl=0  , n=arr.length;
    while(r<n){
        map.put(arr[r], map.getOrDefault(arr[r], 0) + 1);
        if(map.size()>2){
            map.put(arr[l] , map.get(arr[l])-1);
            if (map.get(arr[l]) == 0) {
                map.remove(arr[l]);
            }
            l++;
        }
        maxl=Math.max(maxl , r-l+1);
        r++;
    }

    return maxl;
}
```

Better



```
public static int totalElements(Integer[] arr) {
    // code here
    Map<Integer, Integer> map=new HashMap<>();
    int l=0 , r=0 , maxl=0  , n=arr.length;
    while(r<n){
        map.put(arr[r], map.getOrDefault(arr[r], 0) + 1);
        while(map.size()>2){
            map.put(arr[l] , map.get(arr[l])-1);
            if (map.get(arr[l]) == 0) {
                map.remove(arr[l]);
            }
            l++;
        }
        maxl=Math.max(maxl , r-l+1);
        r++;
    }

    return maxl;
}
```