Mock Test > codeks7407@gmail.com

Full Name:	Yanga Mdede
Email:	codeks7407@gmail.com
Test Name:	**Mock Test**
Taken On:	4 Sep 2024 03:12:51 IST
Time Taken:	9 min 56 sec/ 10 min
Invited by:	Ankush
Invited on:	4 Sep 2024 03:02:56 IST
Skills Score:	
Tags Score:	

100%

105/105

scored in **Mock Test** in 9 min 56 sec on 4 Sep 2024 03:12:51 IST

Algorithms	105/105
Core CS	105/105
Easy	105/105
Problem Solving	105/105
Search	105/105
Sorting	105/105
problem-solving	105/105

Recruiter/Team Comments:

No Comments.

	Question Description	Time Taken	Score	Status
Q1	**Find the Median** > **Coding**	9 min 44 sec	105/ 105	✓

QUESTION 1

✓

Correct Answer

Score 105

Find the Median > Coding

Sorting | Search | Algorithms | Easy | problem-solving | Core CS | Problem Solving

QUESTION DESCRIPTION

The median of a list of numbers is essentially its middle element after sorting. The same number of elements occur after it as before. Given a list of numbers with an odd number of elements, find the median?

Example

$arr = [5, 3, 1, 2, 4]$

The sorted array $arr' = [1, 2, 3, 4, 5]$. The middle element and the median is 3.

Function Description

Complete the *findMedian* function in the editor below.

findMedian has the following parameter(s):

- *int arr[n]:* an unsorted array of integers

Returns

- *int:* the median of the array

Input Format

The first line contains the integer n, the size of arr.
The second line contains n space-separated integers $arr[i]$

Constraints

- $1 \leq n \leq 1000001$
- n is odd
- $-10000 \leq arr[i] \leq 10000$

Sample Input 0

```
7
0 1 2 4 6 5 3
```

Sample Output 0

```
3
```

Explanation 0

The sorted $arr = [0, 1, 2, 3, 4, 5, 6]$. It's middle element is at $arr[3] = 3$.

CANDIDATE ANSWER

Language used: **Java 8**

```

class Result
{

    /*
     * Complete the 'findMedian' function below.
     *
     * The function is expected to return an INTEGER.
     * The function accepts INTEGER_ARRAY arr as parameter.
     */

     public static int findMedian(List<Integer> arr)
     {
         // Write your code here

         Collections.sort(arr);
         int arrSize = arr.size();

         int median = 0;

         if(arrSize % 2 != 0)
         {
             median = arr.get(arrSize / 2);
         }

         else
         {
            int median1 = (arrSize) / (2 - 1);
             int median2 = (arrSize) / (2);

            median = (arr.get(median1) + arr.get(median2)) / 2;
        }

        return median;

    }
}

```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 1	Easy	Sample case	✓ Success	0	0.1072 sec	30.7 KB
Testcase 2	Easy	Hidden case	✓ Success	35	0.1524 sec	32.7 KB
Testcase 3	Easy	Hidden case	✓ Success	35	0.136 sec	33.6 KB
Testcase 4	Easy	Hidden case	✓ Success	35	0.2256 sec	45.6 KB

No Comments

PDF generated at: 3 Sep 2024 21:54:34 UTC