Mock Test > yehdan@msn.com

Full Name:	Jianping Ye
Email:	yehdan@msn.com
Test Name:	**Mock Test**
Taken On:	30 Jul 2025 22:08:13 IST
Time Taken:	1 min 31 sec/ 10 min
Invited by:	Ankush
Invited on:	30 Jul 2025 22:08:06 IST
Skills Score:	
Tags Score:	Algorithms 105/105 Core CS 105/105 Easy 105/105 Problem Solving 105/105 Search 105/105 Sorting 105/105 problem-solving 105/105

100%

105/105

scored in **Mock Test** in 1 min 31 sec on 30 Jul 2025 22:08:13 IST

Recruiter/Team Comments:

No Comments.

	Question Description	Time Taken	Score	Status
Q1	**Find the Median** > **Coding**	1 min 15 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: **C**

```c

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

int findMedian(int n, int* m) {
    int  i, less, greater, equal;
    int  min, max, guess, maxltguess, mingtguess;

    min = max = m[0] ;
    for (i=1 ; i<n ; i++) {
        if (m[i]<min) min=m[i];
        if (m[i]>max) max=m[i];
    }

    while (1) {
        guess = (min+max)/2;
        less = 0; greater = 0; equal = 0;
        maxltguess = min ;
        mingtguess = max ;
        for (i=0; i<n; i++) {
            if (m[i]<guess) {
                less++;
                if (m[i]>maxltguess) maxltguess = m[i] ;
            } else if (m[i]>guess) {
                greater++;
                if (m[i]<mingtguess) mingtguess = m[i] ;
            } else equal++;
        }
        if (less <= (n+1)/2 && greater <= (n+1)/2) break ;
        else if (less>greater) max = maxltguess ;
        else min = mingtguess;
    }
    if (less >= (n+1)/2) return maxltguess;
    else if (less+equal >= (n+1)/2) return guess;
    else return mingtguess;
}
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 1	Easy	Sample case	✓ Success	0	0.0083 sec	7.13 KB
Testcase 2	Easy	Hidden case	✓ Success	35	0.008 sec	7.38 KB
Testcase 3	Easy	Hidden case	✓ Success	35	0.0099 sec	7.38 KB
Testcase 4	Easy	Hidden case	✓ Success	35	0.0237 sec	8.63 KB

No Comments