



Full Name: Đào Nguyễn Huy Hoàng

Email: hoang.dnh210380@sis.hust.edu.vn

Test Name: Mock Test

Taken On: 2 Apr 2023 22:26:46 IST

Time Taken: 5 min 26 sec/ 15 min

Invited by: Ankush

Invited on: 2 Apr 2023 22:26:36 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 5 min 26 sec on 2 Apr 2023 22:26:46 IST

Recruiter/Team Comments:

No Comments.

	Question Description	Time Taken	Score	Status
Q1	Find the Median > Coding	5 min 17 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

```
1 #include <stdio.h>
2 void swap(int* a, int* b)
3 {
4     int temp = *a;
5     *a = *b;
6     *b = temp;
7 }
8
9 //Partition Function
10 int partition(int arr[], int low, int high)
11 {
12     int pivot = arr[high];
13     int i = (low - 1);
14     int j;
15     for (j = low; j <= high - 1; j++) {
16         if (arr[j] <= pivot) {
17             i++;
18             swap(&arr[i], &arr[j]);
19         }
20     }
21     swap(&arr[i + 1], &arr[high]);
22     return (i + 1);
23 }
24
25 // Quick Sort function
26 void quicksort(int Arr[], int low, int high)
27 {
28     if (low < high) {
29         // pi = Partition index
30         int pi = partition(Arr, low, high);
```

```

31     quicksort(Arr, low, pi - 1);
32     quicksort(Arr, pi + 1, high);
33 }
34 }
35 int main(){
36     int N,A[100005];
37     long sum=0;
38     scanf("%d",&N);
39     int t=0;
40     for(int i=1;i<=N;i++){
41         scanf("%d",&A[i]);
42     }
43     quicksort(A,1,N);
44     printf("%d",A[(N+1)/2]);
45 }

```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 1	Easy	Sample case	✔ Success	0	0.0469 sec	7.56 KB
Testcase 2	Easy	Hidden case	✔ Success	35	0.021 sec	7.46 KB
Testcase 3	Easy	Hidden case	✔ Success	35	0.0263 sec	7.46 KB
Testcase 4	Easy	Hidden case	✔ Success	35	0.0408 sec	7.88 KB

No Comments