| | |
|---|---|
| Full Name: | ToYoNiX |
| Email: | toyonix.assemmohamed.2005@gmail.com |
| Test Name: | **Mock Test** |
| Taken On: | 6 Sep 2023 16:39:40 IST |
| Time Taken: | 5 min 54 sec/ 10 min |
| Invited by: | Ankush |
| Invited on: | 6 Sep 2023 16:29:07 IST |
| Skills Score: | |
| Tags Score: | Algorithms 105/105<br>Core CS 105/105<br>Easy 105/105<br>Problem Solving 105/105<br>Search 105/105<br>Sorting 105/105<br>problem-solving 105/105 |

**100%**

**105/105**

scored in **Mock Test** in 5 min 54 sec on 6 Sep 2023 16:39:40 IST

**Recruiter/Team Comments:**

*No Comments.*

| | Question Description | Time Taken | Score | Status |
|---|---|---|---|---|
| Q1 | Find the Median > **Coding** | 5 min 46 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++14**

```cpp
#include <bits/stdc++.h>
#include <algorithm>

using namespace std;

string ltrim(const string &);
string rtrim(const string &);
vector<string> split(const string &);



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

int findMedian(int *arr, int n) {
    sort(arr, arr + n);
    return arr[n / 2];
}

int main()
{
    ofstream fout(getenv("OUTPUT_PATH"));

    string n_temp;
    getline(cin, n_temp);

```

```
    int n = stoi(ltrim(rtrim(n_temp)));

    string arr_temp_temp;
    getline(cin, arr_temp_temp);

    vector<string> arr_temp = split(rtrim(arr_temp_temp));

    int arr[n];

    for (int i = 0; i < n; i++) {
        int arr_item = stoi(arr_temp[i]);

        arr[i] = arr_item;
    }

    int result = findMedian(arr, n);

    fout << result << "\n";

    fout.close();

    return 0;
}

string ltrim(const string &str) {
    string s(str);

    s.erase(
        s.begin(),
        find_if(s.begin(), s.end(), not1(ptr_fun<int, int>(isspace)))
    );

    return s;
}

string rtrim(const string &str) {
    string s(str);

    s.erase(
        find_if(s.rbegin(), s.rend(), not1(ptr_fun<int, int>
(isspace))).base(),
        s.end()
    );

    return s;
}

vector<string> split(const string &str) {
    vector<string> tokens;

    string::size_type start = 0;
    string::size_type end = 0;

    while ((end = str.find(" ", start)) != string::npos) {
        tokens.push_back(str.substr(start, end - start));

        start = end + 1;
    }

    tokens.push_back(str.substr(start));

    return tokens;
}
```

| TESTCASE | DIFFICULTY | TYPE | STATUS | SCORE | TIME TAKEN | MEMORY USED |
|---|---|---|---|---|---|---|
| Testcase 1 | Easy | Sample case | Success | 0 | 0.0267 sec | 8.88 KB |
| Testcase 2 | Easy | Hidden case | Success | 35 | 0.0359 sec | 9.07 KB |
| Testcase 3 | Easy | Hidden case | Success | 35 | 0.0339 sec | 9.14 KB |
| Testcase 4 | Easy | Hidden case | Success | 35 | 0.0444 sec | 13.1 KB |

No Comments

**PDF generated at: 6 Sep 2023 11:17:07 UTC**