Full Name:	Sam Seda
Email:	zyxsamys@gmail.com
Test Name:	**Mock Test**
Taken On:	27 Nov 2023 06:55:57 IST
Time Taken:	6 min 5 sec/ 10 min
Resume:	https://hackerrank-resumes.s3.amazonaws.com/7505911/orhriQxfhSXb9mP_TAqThADGaXRuQ9z9u_Yz0yh4C-NgvwxfG02wYPirrz8wf17UBA/CVSamuelSedaLegariaEN.pdf
Linkedin:	https://www.linkedin.com/in/samuel-seda/
Invited by:	Ankush
Invited on:	27 Nov 2023 06:55:47 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 6 min 5 sec on 27 Nov 2023 06:55:57 IST

Recruiter/Team Comments:

No Comments.

	Question Description	Time Taken	Score	Status
Q1	**Find the Median** > **Coding**	5 min 58 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: **Go**

```go
package main

import (
    "bufio"
    "fmt"
    "io"
    "os"
    "sort"
    "strconv"
    "strings"
)



/*
 * Complete the 'findMedian' function below.
 *
```

```
 * The function is expected to return an INTEGER.
 * The function accepts INTEGER_ARRAY arr as parameter.
 */

func findMedian(arr []int32) int32 {
    // Write your code here
    // Sort Slice
    sort.Slice(arr, func(i,j int) bool{return arr[i]<arr[j]})
    // Get Mid Index
    midIndex := len(arr)/2
    return arr[midIndex]
}

func main() {
    reader := bufio.NewReaderSize(os.Stdin, 16 * 1024 * 1024)

    stdout, err := os.Create(os.Getenv("OUTPUT_PATH"))
    checkError(err)

    defer stdout.Close()

    writer := bufio.NewWriterSize(stdout, 16 * 1024 * 1024)

    nTemp, err := strconv.ParseInt(strings.TrimSpace(readLine(reader)), 10,
64)
    checkError(err)
    n := int32(nTemp)

    arrTemp := strings.Split(strings.TrimSpace(readLine(reader)), " ")

    var arr []int32

    for i := 0; i < int(n); i++ {
        arrItemTemp, err := strconv.ParseInt(arrTemp[i], 10, 64)
        checkError(err)
        arrItem := int32(arrItemTemp)
        arr = append(arr, arrItem)
    }

    result := findMedian(arr)

    fmt.Fprintf(writer, "%d\n", result)

    writer.Flush()
}

func readLine(reader *bufio.Reader) string {
    str, _, err := reader.ReadLine()
    if err == io.EOF {
        return ""
    }

    return strings.TrimRight(string(str), "\r\n")
}

func checkError(err error) {
    if err != nil {
        panic(err)
    }
}
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED

Testcase 1	Easy	Sample case	Success	0	0.0024 sec	3.09 KB
Testcase 2	Easy	Hidden case	Success	35	0.0051 sec	3.52 KB
Testcase 3	Easy	Hidden case	Success	35	0.0047 sec	5.52 KB
Testcase 4	Easy	Hidden case	Success	35	0.018 sec	7.57 KB

No Comments

PDF generated at: 27 Nov 2023 01:33:32 UTC