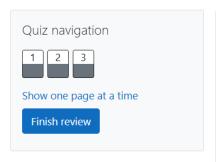
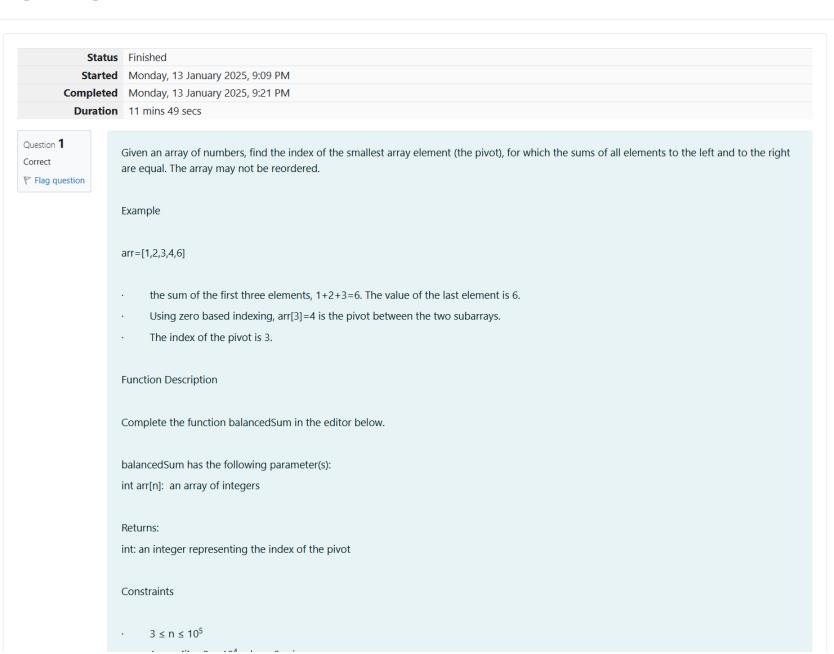
GE23131-Programming Using C-2024





 $1 \le arr[i] \le 2 \times 10^4$, where $0 \le i < n$ It is guaranteed that a solution always exists. Input Format for Custom Testing Input from stdin will be processed as follows and passed to the function. The first line contains an integer n, the size of the array arr. Each of the next n lines contains an integer, arr[i], where $0 \le i < n$. Sample Case 0 Sample Input 0 STDIN Function Parameters $4 \rightarrow arr[] size n = 4$ 1 \rightarrow arr = [1, 2, 3, 3] 2 3 3 Sample Output 0 2 Explanation 0 The sum of the first two elements, 1+2=3. The value of the last element is 3. Using zero based indexing, arr[2]=3 is the pivot between the two subarrays. The index of the pivot is 2. Sample Case 1 Sample Input 1

STDIN Function Parameters

```
3 → arr[] size n = 3
1 → arr = [1, 2, 1]
2
1
Sample Output 1
```

Explanation 1

- · The first and last elements are equal to 1.
- · Using zero based indexing, arr[1]=2 is the pivot between the two subarrays.
- · The index of the pivot is 1.

Answer: (penalty regime: 0 %)

Reset answer

```
* Complete the 'balancedSum' function below.
 3
     * The function is expected to return an INTEGER.
 4
     * The function accepts INTEGER_ARRAY arr as parameter.
 5
 6
 7
    int balancedSum(int arr_count, int* arr)
 8
 9 ,
        int totalsum=0;
10
        for(int i=0;i<arr_count;i++){</pre>
11 1
            totalsum+=arr[i];
12
13
14
        int leftsum=0;
        for(int i=0;i<arr_count;i++){</pre>
15
            int rightsum=totalsum-leftsum-arr[i];
16
            if(leftsum==rightsum){
17
18
                return i;
19
            leftsum+=arr[i];
20
21
22
        return 1;
23
24
25
```

	Test	Expected	Got	
~	<pre>int arr[] = {1,2,3,3}; printf("%d", balancedSum(4, arr))</pre>	2	2	~

Passed all tests! <

Question **2**Correct

Flag question

Calculate the sum of an array of integers.

Example

numbers = [3, 13, 4, 11, 9]

The sum is 3 + 13 + 4 + 11 + 9 = 40.

Function Description

Complete the function arraySum in the editor below.

arraySum has the following parameter(s):

int numbers[n]: an array of integers

Returns

int: integer sum of the numbers array

Constraints

 $1 \le n \le 10^4$

 $1 \le numbers[i] \le 10^4$

Input Format for Custom Testing

Input from stdin will be processed as follows and passed to the function.

The first line contains an integer n, the size of the array numbers.

Each of the next n lines contains an integer numbers[i] where $0 \le i < n$.

Sample Case 0

Sample Input 0

```
STDIN Function
```

 $5 \rightarrow numbers[] size n = 5$

1 \rightarrow numbers = [1, 2, 3, 4, 5]

2

3

4

5

Sample Output 0

15

Explanation 0

$$1 + 2 + 3 + 4 + 5 = 15$$
.

Sample Case 1

Sample Input 1

STDIN Function

 $2 \rightarrow numbers[] size n = 2$

12 → numbers = [12, 12]

12

Sample Output 1

```
Explanation 1
12 + 12 = 24.
Answer: (penalty regime: 0 %)
 Reset answer
   1 v
        * Complete the 'arraySum' function below.
   2
   3
   4
        * The function is expected to return an INTEGER.
   5
        * The function accepts INTEGER ARRAY numbers as parameter.
   6
   7
   8
       int arraySum(int numbers count, int *numbers)
   9
  10
           int sum=0;
           for(int i=0;i<numbers count;i++){</pre>
  11
  12
               sum=sum+numbers[i];
  13
  14
           return sum;
  15
  16
```

I		Test	Expected	Got	
	~	<pre>int arr[] = {1,2,3,4,5}; printf("%d", arraySum(5, arr))</pre>	15	15	~
Pa	asse	d all tests! 🗸			

arr. Each of the following n lines contains an integer that describes arr[i] (where 0 ≤ i < n). Sample Case 0 Sample Input For Custom Testing

Question **3**Incorrect

Flag question

Given an array of n integers, rearrange them so that the sum of the absolute differences of all adjacent elements is minimized. Then, compute the sum of those absolute differences. Example n = 5 arr = [1, 3, 3, 2, 4] If the list is rearranged as arr' = [1, 2, 3, 3, 4], the absolute differences are |1 - 2| = 1, |2 - 3| = 1, |3 - 3| = 0, |3 - 4| = 1. The sum of those differences is 1 + 1 + 0 + 1 = 3. Function Description Complete the function minDiff in the editor below. minDiff has the following parameter: arr: an integer array Returns: int: the sum of the absolute differences of adjacent elements Constraints $2 \le n \le 105$ $0 \le arr[i] \le 109$, where $0 \le i < n$ Input Format For Custom Testing The first line of input contains an integer, n, the size of

STDIN Function ---- 5 \rightarrow arr[] size n = 5 5 \rightarrow arr[] = [5, 1, 3, 7, 3] 1 3 7 3 Sample Output 6 Explanation n = 5 arr = [5, 1, 3, 7, 3] If arr is rearranged as arr' = [1, 3, 3, 5, 7], the differences are minimized. The final answer is |1 - 3| + |3 - 3| + |3 - 5| + |5 - 7| = 6. Sample Case 1 Sample Input For Custom Testing STDIN Function ---- 2 \rightarrow arr[] size n = 2 3 \rightarrow arr[] = [3, 2] 2 Sample Output 1 Explanation n = 2 arr = [3, 2] There is no need to rearrange because there are only two elements. The final answer is |3 - 2| = 1.

Answer: (penalty regime: 0 %)

```
Reset answer
```

```
2
     * Complete the 'minDiff' function below.
 3
     * The function is expected to return an INTEGER.
 4
     * The function accepts INTEGER ARRAY arr as parameter.
 6
    #include<stdio.h>
 7
    int minDiff(int arr count, int* arr) {
        return (*(int*)a - *(int*)b);
10
11
    int minDiff(int arr count,int* arr)
12
13
        qsort(arr,arr count,sizeof(int),compare);
14
        int totaldiff=0;
15
        for(int i=1;i<arr count;i++){</pre>
16
            totaldiff +=abs(arr[i]-arr[i-1]);
17
18
        return totaldiff;
19
20
```

```
Syntax Error(s)
tester .c: In function 'minDiff':
__tester__.c:i3:20: error: 'a' undeclared (Tirst use in this Tunction)
  13 | return (*(int*)a - *(int*)b);
__tester__.c:13:20: note: each undeclared identifier is reported only once for each function it appears in
__tester__.c:13:31: error: 'b' undeclared (first use in this function)
  13 | return (*(int*)a - *(int*)b);
__tester__.c: At top level:
__tester__.c:15:5: error: redefinition of 'minDiff'
  15 | int minDiff(int arr_count,int* arr)
__tester__.c:12:5: note: previous definition of 'minDiff' with type 'int(int, int *)'
  12 | int minDiff(int arr_count, int* arr) {
     ^~~~~~
__tester__.c: In function 'minDiff':
__tester__.c:17:37: error: 'compare' undeclared (first use in this function)
  17 | qsort(arr,arr_count,sizeof(int),compare);
```

Finish review