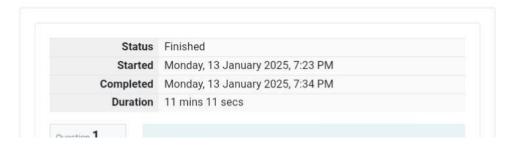
Week 13:

ROLL NO:240801186

Name: MAGESH BAPU P



Q1) Given an array of numbers, find the index of the smallest array element (the pivot), for which the sums of all elements to the left and to the right are equal. The array may not be reordered.

Example

arr=[1,2,3,4,6]

- the sum of the first three elements, 1+2+3=6. The value of the last element is 6.
- · Using zero based indexing, arr[3]=4 is the pivot between the two subarrays.
- · The index of the pivot is 3.

Function Description

Complete the function balancedSum in the editor below.

balancedSum has the following parameter(s):

int arr[n]: an array of integers

Returns:

int: an integer representing the index of the pivot

Constraints

- 3 ≤ n ≤ 105
- · $1 \le arr[i] \le 2 \times 104$, 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 → arr[] size n = 4

1 → arr = [1, 2, 3, 3]

2

3

Sample Output 0
```

Explanation 0

2

- 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

$$3 \rightarrow arr[] size n = 3$$

$$1 \rightarrow arr = [1, 2, 1]$$

2

1

Sample Output 1

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.

Code:

```
1 \text{ T} /* 2 \text{ } /* Complete the 'balancedSum' function below.
      * The function is expected to return an INTEGER.

* The function accepts INTEGER_ARRAY arr as parameter.
4
5
     int balancedSum(int arr_count, int* arr)
8
9 ₹ {
           int totalsum = 0;
for (int i =0;i<arr_count;i++){</pre>
10
11
                 totalsum += arr[i];
12
13
           int leftsum =0;
14
           for(int i =0;icarr_count;i++){
  int rightsum = totalsum - leftsum -arr[i];
  if(leftsum==rightsum){
15 1
16
17 ,
                     `return i;
18
19
20
                leftsum +=arr[i];
21
22
           return 1;
23
     }
24
```

OUTPUT:

```
Test

int arr[] = {1,2,3,3};
printf("%d", balancedSum(4, arr))

Passed all tests! ✓
```

```
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 104
1 \le numbers[i] \le 104
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
```

Q2) Calculate the sum of an array of integers.

```
15
```

Explanation 0

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

Sample Case 1

Sample Input 1

STDIN Function

- 2 \rightarrow numbers[] size n = 2
- 12 \rightarrow numbers = [12, 12]

12

Sample Output 1

24

Explanation 1

```
12 + 12 = 24.
```

Code:

OUTPUT:

```
Test Expected Got

int arr[] = {1,2,3,4,5};
printf("%d", arraySum(5, arr))

Passed all tests! ✓
```

Code:

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

OUTPUT:

	Test	Expected	Got	
~	<pre>int arr[] = {5, 1, 3, 7, 3}; printf("%d", minDiff(5, arr))</pre>	6	6	~

Passed all tests! 🗸