

**Ex. No. : 59****Date : 20.12.2024****Balanced Array****Problem Statement:**

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**

- $3 \leq n \leq 105$
- $1 \leq \text{arr}[i] \leq 2 \times 10^4$ , where  $0 \leq i < n$
- 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  $0 \leq i < n$ .

**Sample Input**

STDIN		Function Parameters
-----		-----
4	→	arr[] size n = 4
1	→	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.

**Program:**

```

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

```

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

Passed all tests! ✓

**Ex. No. : 60****Date : 20.12.2024****Sum Them All****Problem Statement:**

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**

$1 \leq n \leq 104$

$1 \leq \text{numbers}[i] \leq 104$

**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  $0 \leq i < n$ .

**Sample Input**

STDIN

-----

5      →

1      →

2

3

4

5

Function

-----

numbers[] size n = 5

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

**Sample Output**

15

**Explanation**

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

**Program:**

```

1  /*
2   * Complete the 'arraySum' function below.
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;
11     for(int i=0;i<numbers_count;i++)
12     {
13         sum+=numbers[i];
14     }
15     return sum;
16 }
17
18

```

	Test	Expected	Got	
✓	int arr[] = {1,2,3,4,5}; printf("%d", arraySum(5, arr))	15	15	✓

Passed all tests! ✓

**Ex. No. : 61****Date : 20.12.2024****Minimum Difference Sum****Problem Statement:**

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**

$2 \leq n \leq 105$

$0 \leq arr[i] \leq 109$ , where  $0 \leq i < n$

**Input Format For Custom Testing**

The first line of input contains an integer,  $n$ , the size of `arr`.

Each of the following  $n$  lines contains an integer that describes `arr[i]` (where  $0 \leq i < n$ ).

**Sample Input For Custom Testing**

STDIN		Function
-----		-----
5	→	<code>arr[]</code> size $n = 5$
5	→	<code>arr[] = [5, 1, 3, 7, 3]</code>
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  $|1 - 3| + |3 - 3| + |3 - 5| + |5 - 7| = 6$ .

**Program:**

```

1  /*
2   * Complete the 'minDiff' function below.
3   *
4   * The function is expected to return an INTEGER.
5   * The function accepts INTEGER_ARRAY arr as parameter.
6   */
7
8  int minDiff(int arr_count, int* arr)
9  {
10     for(int i=0;i<arr_count-1;i++)
11     {
12         for(int j=0;j<arr_count-i-1;j++)
13         {
14             if(arr[j]>arr[j+1])
15             {
16                 int temp=arr[j];
17                 arr[j]=arr[j+1];
18                 arr[j+1]=temp;
19             }
20         }
21     }
22     int sum=0;
23     for(int i=0;i<arr_count-1;i++)
24     {
25         sum+= abs(arr[i]-arr[i+1]);
26     }
27     return sum;
28 }
29

```

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

Passed all tests! ✓