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.

```
int balancedSum(int arr_count, int* arr)
{
    total_sum=sum(arr)
    left_sum=0
    left_sum+=arr[i
    return 0;]

}
```

Syntax Error(s)

```
__tester__.c: In function ‘balancedSum’:
__tester__.c:14:5: error: ‘total_sum’ undeclared (first use in this function)
   14 |     total_sum=sum(arr)
      |     ^~~~~~~~~
__tester__.c:14:5: note: each undeclared identifier is reported only once for ea
__tester__.c:14:15: error: implicit declaration of function ‘sum’ [-Werror=impli
```

Calculate the sum of an array of integers.

Example

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

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

```
int arraySum(int numbers_count, int *numbers)
{
    int arr = [1,2,3,4,5];
    let arraysum =arr.reduce((acc,currentvalue) => acc +cutrrentvalue

}
```

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 ≤ n ≤105 0 ≤ arr[i] ≤ 109, where 0 ≤ 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 ≤ i < n) . Sample Case 0 Sample Input For Custom Testing STDIN Function ----- ----- --- 5 → arr[] size n = 5 5 → 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 → arr[] size n = 2 3 → arr[] = [3, 2] 2 Sample Output 1 Explanation n = 2 arr = [3, 2] There is no need to rearrange

```
int minDiff(int arr_count, int* arr)
{
    arr.sort()
    total_dif =0
    for i in range(1,len(arr));
    total_diff+=abs(arr[i] -arr[i-1])
    return total_diff

}
```