Sherlock and Array

Watson gives Sherlock an array of integers. His challenge is to find an element of the array such that the sum of all elements to the left is equal to the sum of all elements to the right.

Example

$arr = [5, 6, 8, 11]$

8 is between two subarrays that sum to 11.

$arr = [1]$

The answer is $[1]$ since left and right sum to 0.

You will be given arrays of integers and must determine whether there is an element that meets the criterion. If there is, return `YES`. Otherwise, return `NO`.

Function Description

Complete the *balancedSums* function in the editor below.

balancedSums has the following parameter(s):

- *int arr[n]:* an array of integers

Returns

- *string:* either `YES` or `NO`

Input Format

The first line contains T, the number of test cases.

The next T pairs of lines each represent a test case.
- The first line contains n, the number of elements in the array arr.
- The second line contains n space-separated integers $arr[i]$ where $0 \leq i < n$.

Constraints

$1 \leq T \leq 10$
$1 \leq n \leq 10^5$
$1 \leq arr[i] \leq 2 \times 10^4$
$0 \leq i < n$

Sample Input 0

```
2
3
1 2 3
```

```
4
1 2 3 3
```

Sample Output 0

```
NO
YES
```

Explanation 0

For the first test case, no such index exists.
For the second test case, $arr[0] + arr[1] = arr[3]$, therefore index 2 satisfies the given conditions.

Sample Input 1

```
3
5
1 1 4 1 1
4
2 0 0 0
4
0 0 2 0
```

Sample Output 1

```
YES
YES
YES
```

Explanation 1

In the first test case, $arr[2] = 4$ is between two subarrays summing to 2.
In the second case, $arr[0] = 2$ is between two subarrays summing to 0.
In the third case, $arr[2] = 2$ is between two subarrays summing to 0.