Given an array of integers nums and an integer target, return *indices of the two numbers such that they add up to target*.

You may assume that each input would have ***exactly* one solution**, and you may not use the *same* element twice.

You can return the answer in any order.

**Example 1:**

**Input:** nums = [2,7,11,15], target = 9

**Output:** [0,1]

**Output:** Because nums[0] + nums[1] == 9, we return [0, 1].

**Example 2:**

**Input:** nums = [3,2,4], target = 6

**Output:** [1,2]

**Example 3:**

**Input:** nums = [3,3], target = 6

**Output:** [0,1]

**Constraints:**

* 2 <= nums.length <= 103
* -109 <= nums[i] <= 109
* -109 <= target <= 109
* **Only one valid answer exists.**

A really brute force way would be to search for all possible pairs of numbers but that would be too slow. Again, it's best to try out brute force solutions for just for completeness. It is from these brute force solutions that you can come up with optimizations.

Hide Hint 2

So, if we fix one of the numbers, say

x

, we have to scan the entire array to find the next number

y

which is

value - x

where value is the input parameter. Can we change our array somehow so that this search becomes faster?

Hide Hint 3

The second train of thought is, without changing the array, can we use additional space somehow? Like maybe a hash map to speed up the search?