Two sum problem :

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]
Explanation: 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:

- $2 <= nums.length <= 10^4$
- $-10^9 <= nums[i] <= 10^9$
- $-10^9 <= target <= 10^9$
- Only one valid answer exists.

Used approach : unordered hashmap

Step - 1 : first we created hashmap and start iterating through the Array.

Step - 2 : we will check for each element y, if there exists a target-y in the map

If there exists a target-y in the map. If it exists, we will return the indices of both integers. Otherwise, We will store that element and it’s index as key and mapped value in the map. This is done to further check for that element and also get it’s index if required.