1. Two Sum - 17/08/24 - (Easy)

```
/*
    COMPANIES LIST :

    Accolite, Amazon, Microsoft, Infosys, Zoho, Flipkart, Morgan Stanley,
    FactSet, Hike, ABCO, Wipro, SAP Labs, CarWale
*/
```

1. Two Sum

Easy | Topics | Companies | Hint

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:

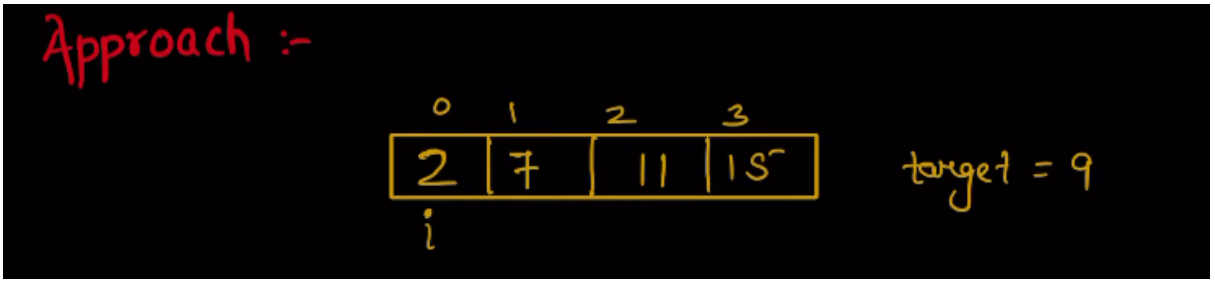
> **Input:** nums = [2,7,11,15], target = 9
> **Output:** [0,1]
> **Explanation:** Because nums[0] + nums[1] == 9, we return [0, 1].

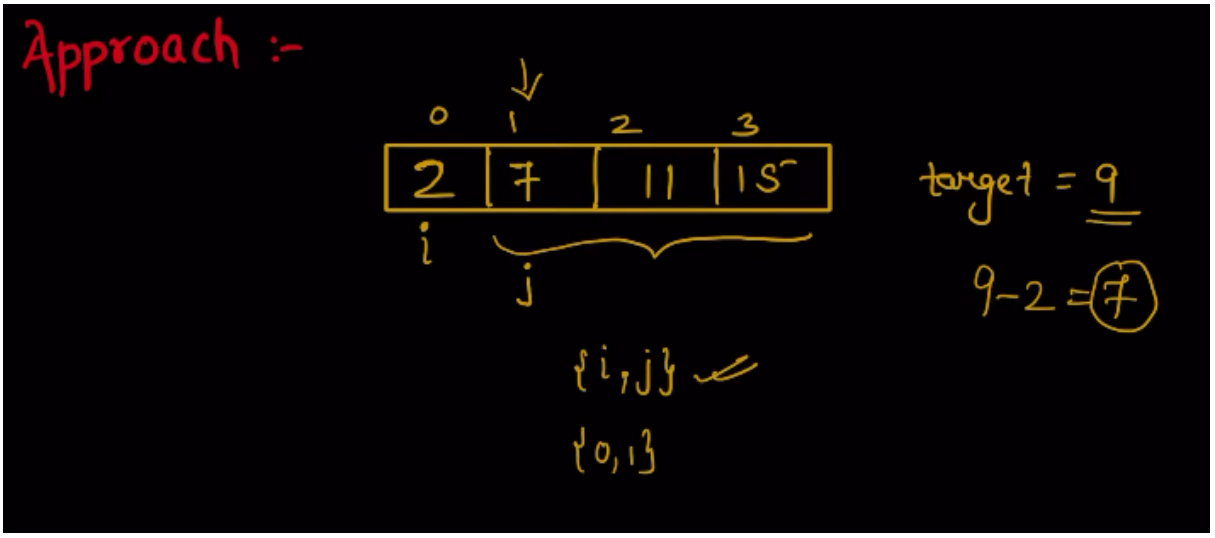
Example 2:

> **Input:** nums = [3,2,4], target = 6
> **Output:** [1,2]

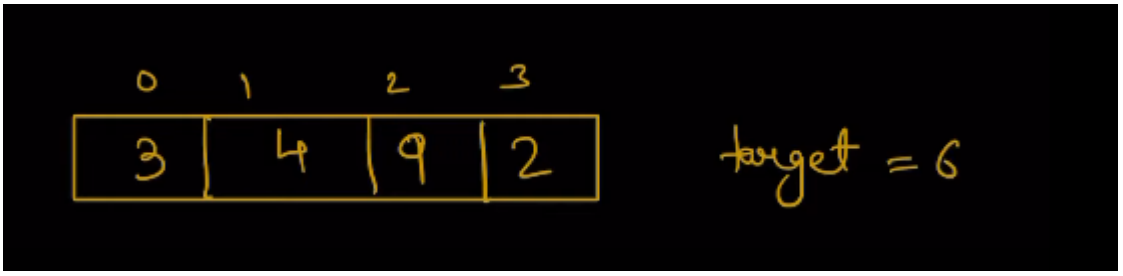
Example 3:

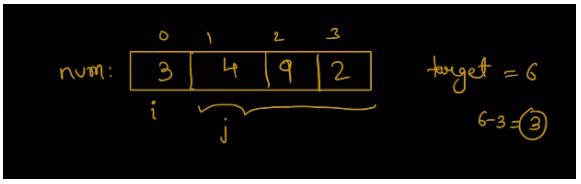
> **Input:** nums = [3,3], target = 6
> **Output:** [0,1]



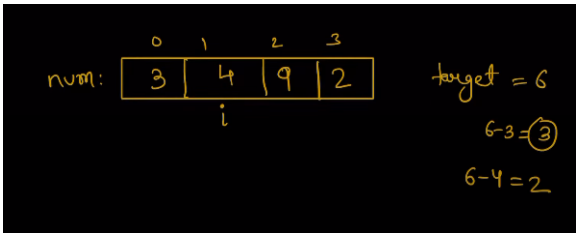


another one

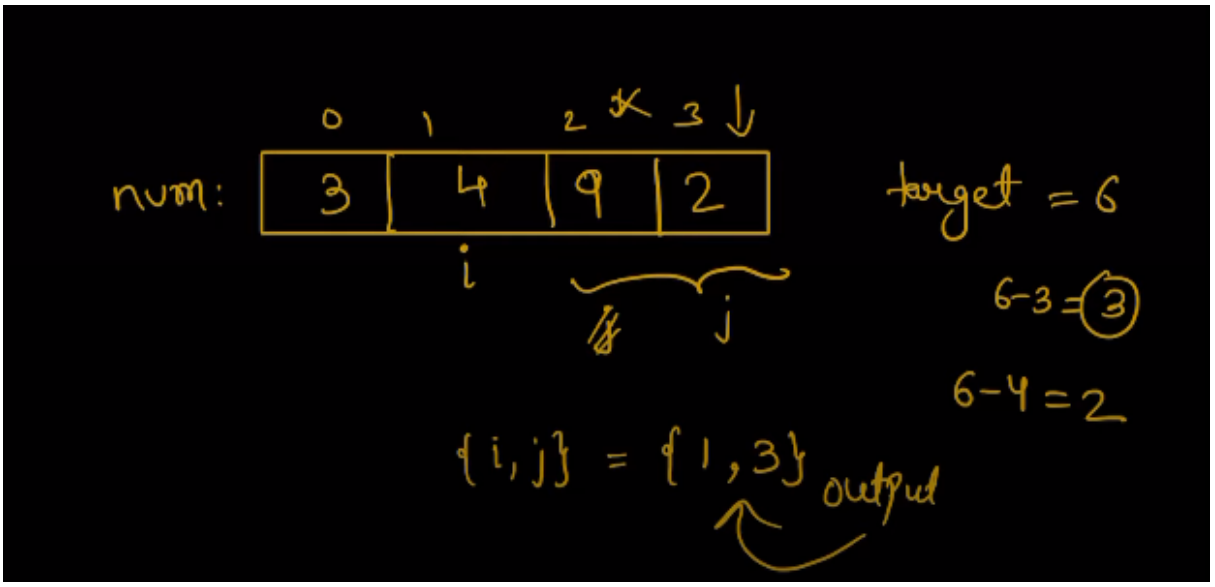




this is not correct as therer is no value 3 so we incrmenet i



now will chekc j if there is any target-nums[i] ,, yes we have 2 so we will find the answer

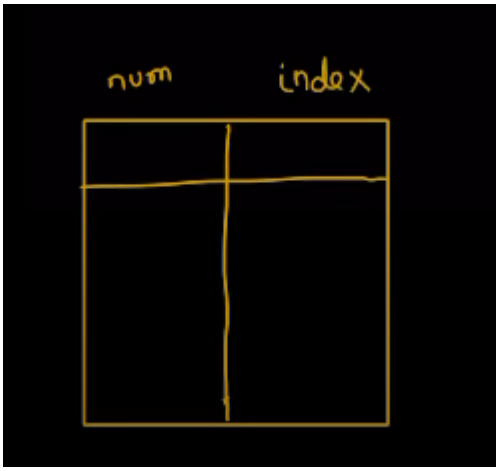


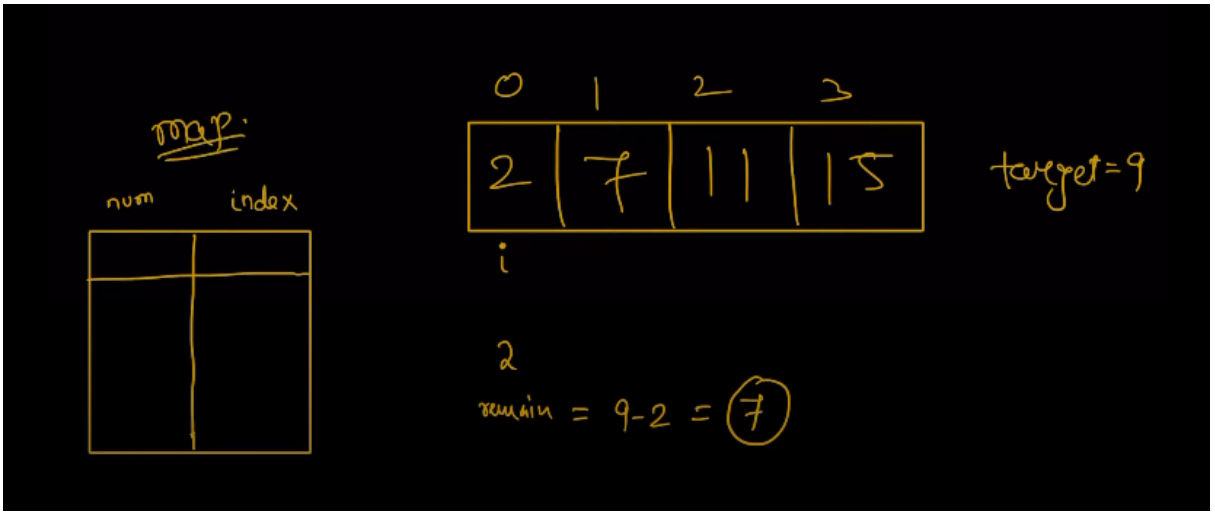
T.C : O(n^2)

now a improve one :

with T.C : O (n)

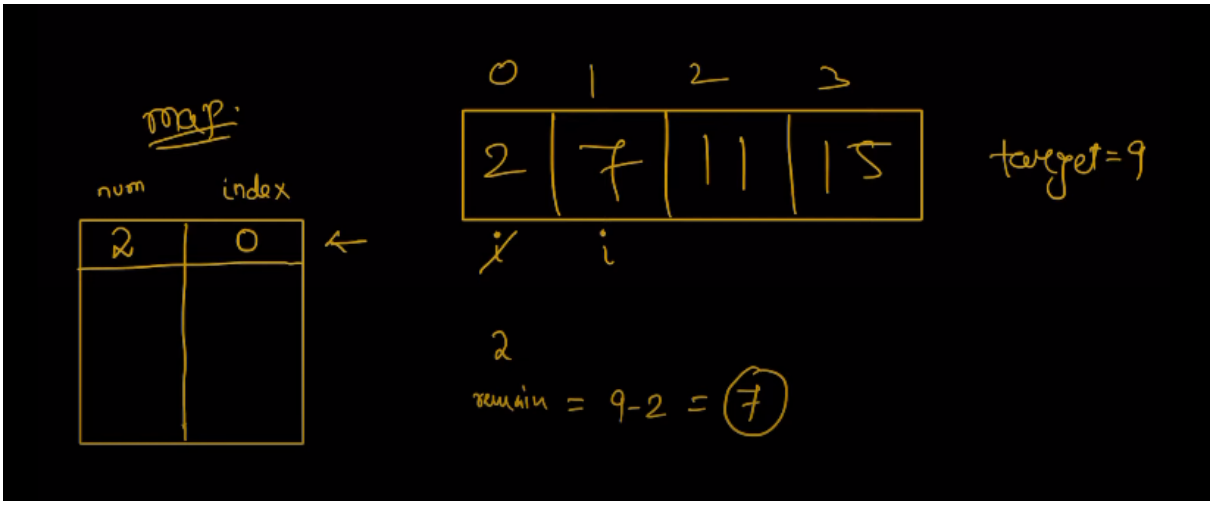
we will use a map



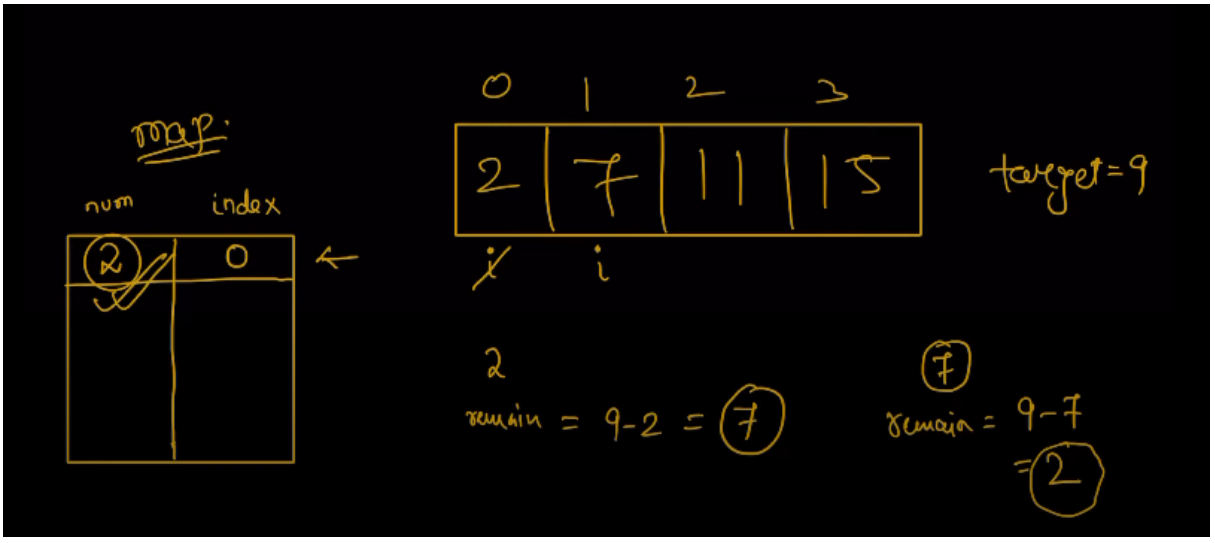


here will target-2 =7

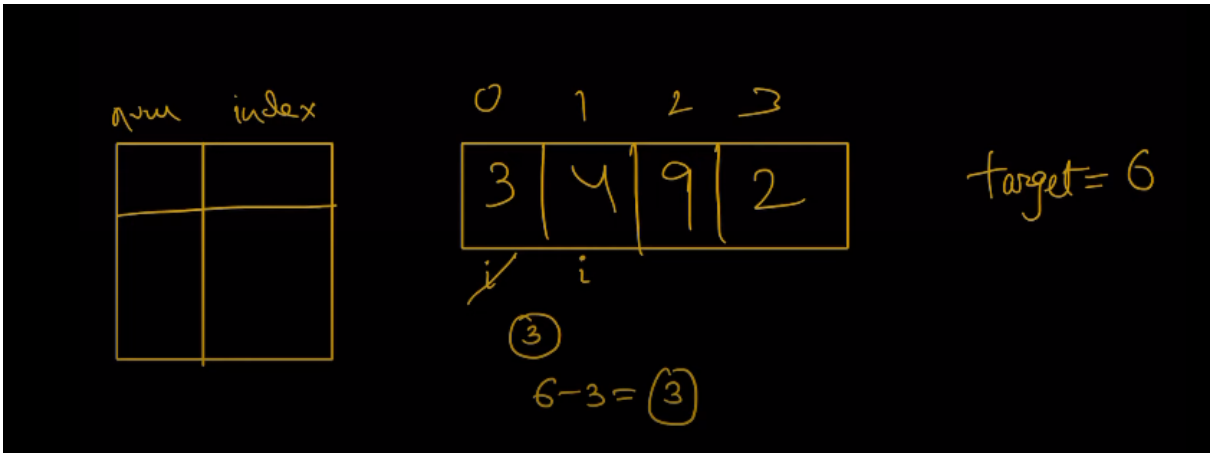
we will check if we have 7 in the map
if no then increment i and store 2 in map with index

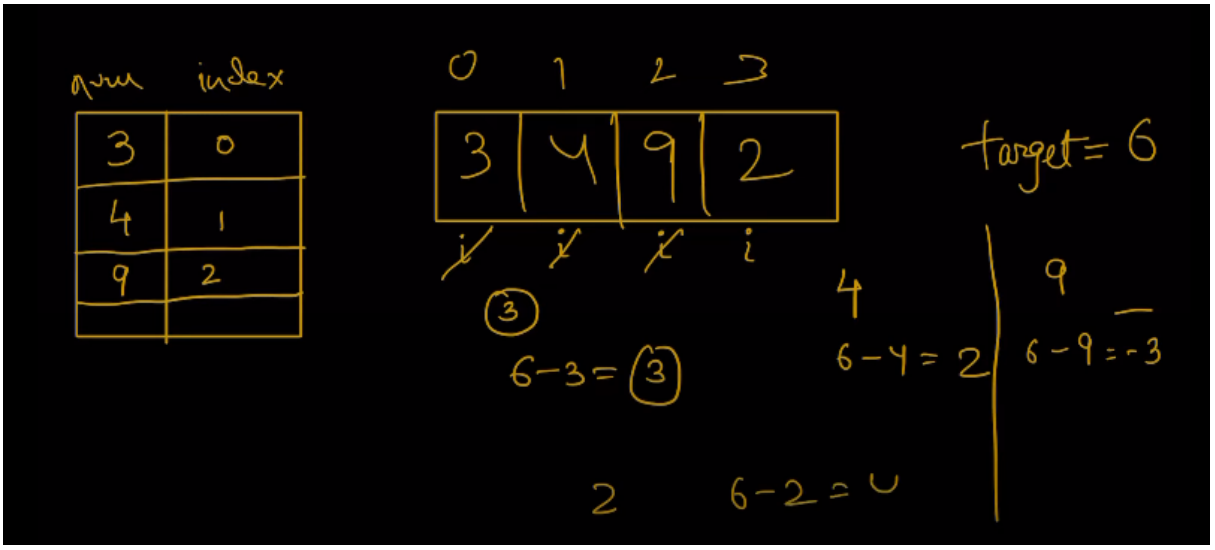


now we are at index 1: number 7
target-7 =2
now we will check iof there is 2 in map
yes there is so we print iondex of 7 and 2

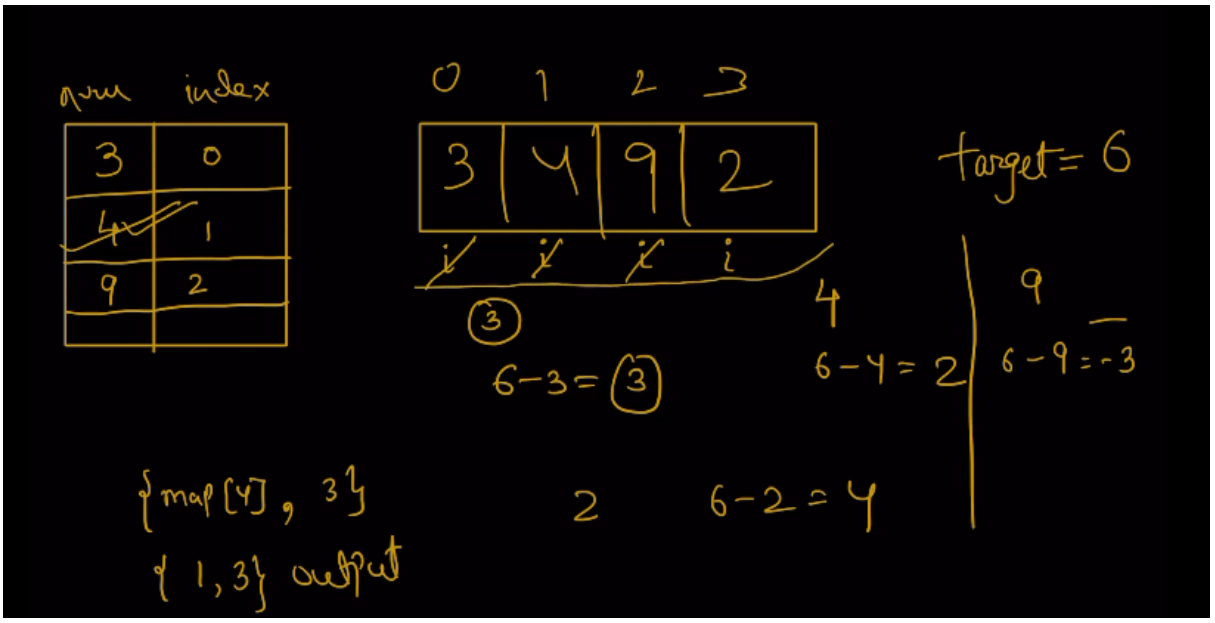

map:
num
index
2
0
0 1 2 3
2 7 11 15
target = 9
i
i
2
remain = 9-2 = 7
7
remain = 9-7
= 2

Another one


num
index
0 1 2 3
3 4 9 2
target = 6
i
i
3
6-3 = 3


num
index
3 0
4 1
9 2
0 1 2 3
3 4 9 2
target = 6
i
i
i
i
3
6-3 = 3
4
6-4 = 2
2
6-2 = 0
9
6-9 = -3

finally at last we at index 3 : 2 value



answer

Brute Force

```
class Solution {
public:
    vector<int> twoSum(vector<int>& nums, int target) {
        for(int i = 0; i < nums.size(); i++) {
            // For each number, iterate through the numbers t
            for(int j = i + 1; j < nums.size(); j++) {
                // Check if the sum of the two numbers equals
                if(nums[i] + nums[j] == target) {
                    // Return the indices of the two numbers
                    return {i, j};
                }
            }
        }
        // If no solution is found, return an empty vector
        return {};
```

```
    }
};
```

Optimal

```
class Solution {
public:
    vector<int> twoSum(vector<int>& nums, int target) {
        map <int,int> mp; //< number, index>
        int n = nums.size();
        for(int i=0;i<n;i++){
            int remain = target-nums[i];
            if(mp.find(remain)!= mp.end()){

                return {mp[remain],i};
            }
            mp[nums[i]] = i;
        }
        return {};
    }
};
```