1. Two Sum

Easy 39542 1275 Add to List Share

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]
```

```python
class Solution:
    def twoSum(self, nums: List[int], target: int) -> List[int]:

        lis = {}
        for i, num in enumerate(nums):
            n = target - num
            if n not in lis:
                lis[num] = i
            else:
                return [lis[n], i]
```

Testcase | Run Code Result | Debugger

Accepted Runtime: 56 ms

Your input
[2,7,11,15]
9

Output [0,1] Diff

Expected [0,1]

Problems | Pick One | Prev 1/603 Next | Console | Use Example Testcases | Run Code | Submit

27. Remove Element

Easy 4640 6359 Add to List Share

Given an integer array `nums` and an integer `val`, remove all occurrences of `val` in `nums` **in-place**. The relative order of the elements may be changed.

Since it is impossible to change the length of the array in some languages, you must instead have the result be placed in the **first part** of the array `nums`. More formally, if there are `k` elements after removing the duplicates, then the first `k` elements of `nums` should hold the final result. It does not matter what you leave beyond the first `k` elements.

Return `k` *after placing the final result in the first* `k` *slots of* `nums`.

Do **not** allocate extra space for another array. You must do this by **modifying the input array in-place** with O(1) extra memory.

Custom Judge:

The judge will test your solution with the following code:

```
int[] nums = [...]; // Input array
int val = ...; // Value to remove
int[] expectedNums = [...]; // The expected answer with correct length.
                            // It is sorted with no values equaling val
```

```python
class Solution:
    def removeElement(self, nums: List[int], val: int) -> int:
        count = 0
        for i in range(len(nums)):
            if nums[i] != val :
                nums[count] = nums[i]
                count +=1
        return count
```

Testcase | Run Code Result | Debugger

Accepted Runtime: 62 ms

Your input
[3,2,2,3]
3

Output [2,2] Diff

Expected [2,2]

Problems | Pick One | Prev 8/603 Next | Console | Use Example Testcases | Run Code | Submit

9. Palindrome Number

Easy 7678 2340 Add to List Share

Given an integer `x`, return `true` *if* `x` *is a* ***palindrome****, and* `false` *otherwise.*

Example 1:

```
Input: x = 121
Output: true
Explanation: 121 reads as 121 from left to right and from right to left.
```

Example 2:

```
Input: x = -121
Output: false
Explanation: From left to right, it reads -121. From right to left, it
becomes 121-. Therefore it is not a palindrome.
```

Example 3:

```
Input: x = 10
```

```python
class Solution:

    def isPalindrome(self, x):
        return str(x) == str(x)[::-1]

```

Testcase | Run Code Result | Debugger

Accepted Runtime: 32 ms

Your input	121 -121
Output	true false
Expected	true false

Diff

Problems | Pick One | Prev 2/603 Next | Console | Use Example Testcases | Run Code | Submit