Q:-1 EASSY Given an integer array nums of size n, find the maximum and minimum elements in the array and return them as a pair.

Write a function findMinMax that takes in an integer array nums and returns a pair of integers containing the maximum and minimum elements.

You can assume that nums will always have at least one element.

Input:-An integer array nums of size n (1 ≤ n ≤ 10^4) where each element is an integer (-10^9 ≤ nums[i] ≤ 10^9).

Output:-Return an integer array containing two elements: the maximum element followed by the minimum element from nums.

Example 1:

Input: nums = [4, 2, 5, 1, 6, 3]

Output: [6, 1]

Explanation: The maximum element is 6, and the minimum element is 1.

Example 2:

Input: nums = [1, 5, 7, 2, 9, 3]

Output: [9, 1]

Explanation: The maximum element is 9, and the minimum element is 1.

Q:-2 MEDIUM Rotate Array.

Given an integer array nums, rotate the array to the right by k steps, where k is non-negative.

Example 1:

Input: nums = [1,2,3,4,5,6,7], k = 3

Output: [5,6,7,1,2,3,4]

Explanation:

rotate 1 steps to the right: [7,1,2,3,4,5,6]

rotate 2 steps to the right: [6,7,1,2,3,4,5]

rotate 3 steps to the right: [5,6,7,1,2,3,4]

Example 2:

Input: nums = [-1,-100,3,99], k = 2

Output: [3,99,-1,-100]

Explanation:

rotate 1 steps to the right: [99,-1,-100,3]

rotate 2 steps to the right: [3,99,-1,-100]

Constraints:

1 <= nums.length <= 105

-231 <= nums[i] <= 231 - 1

0 <= k <= 105

Q:-3EASSY Array Reverse

Given an array arr[], the task is to reverse the array. Reversing an array means rearranging the elements such that the first element becomes the last, the second element becomes second last and so on.

Examples:1

Input: arr[] = {1, 4, 3, 2, 6, 5}

Output: {5, 6, 2, 3, 4, 1}

Explanation: The first element 1 moves to last position, the second element 4 moves to second-last and so on.

Examples:1

Input: arr[] = {4, 5, 1, 2}

Output: {2, 1, 5, 4}

Explanation: The first element 4 moves to last position, the second element 5 moves to second last and so on.

Q:-4 MEDIUM Reverse Integer.

Given a signed 32-bit integer x, return x with its digits reversed. If reversing x causes the value to go outside the signed 32-bit integer range [-231, 231 - 1], then return 0.

Assume the environment does not allow you to store 64-bit integers (signed or unsigned).

Example 1:

Input: x = 123

Output: 321

Example 2:

Input: x = -123

Output: -321

Example 3:

Input: x = 120

Output: 21

Constraints:

-231 <= x <= 231 – 1

Q:-5 MEDIUM Majority Element

Given an array nums of size n, return the majority element.

The majority element is the element that appears more than ⌊n / 2⌋ times. You may assume that the majority element always exists in the array.

Example 1:

Input: nums = [3,2,3]

Output: 3

Example 2:

Input: nums = [2,2,1,1,1,2,2]

Output: 2

Constraints:

n == nums.length

1 <= n <= 5 * 104

-109 <= nums[i] <= 109

Q:-6 EASSY Check if array is sorted.

Given an array arr[], check whether it is sorted in non-decreasing order. Return true if it is sorted otherwise false.

Examples:-1

Input: arr[] = [10, 20, 30, 40, 50]

Output: true

Example:-2

Explanation: The given array is sorted.

Input: arr[] = [90, 80, 100, 70, 40, 30]

Output: false

Explanation: The given array is not sorted.

Constraints:

1 ≤ arr.size ≤ 106

- 109 ≤ arr[i] ≤ 109

Q:-7 EASSY Second Largest Element in an Array.

Given an array of positive integers arr[] of size n, the task is to find second largest distinct element in the array.

Note: If the second largest element does not exist, return -1.

Examples:-1

Input: arr[] = [12, 35, 1, 10, 34, 1]

Output: 34

Explanation: The largest element of the array is 35 and the second largest element is 34.

Example:-2

Input: arr[] = [10, 5, 10]

Output: 5

Explanation: The largest element of the array is 10 and the second largest element is 5.

Input: arr[] = [10, 10, 10]

Output: -1

Explanation: The largest element of the array is 10 there is no second largest element.

Q:-8 MEDIUM Binary Search.

Given an array of integers nums which is sorted in ascending order, and an integer target, write a function to search target in nums. If target exists, then return its index. Otherwise, return -1.

You must write an algorithm with O(log n) runtime complexity.

Example 1:

Input: nums = [-1,0,3,5,9,12], target = 9

Output: 4

Explanation: 9 exists in nums and its index is 4

Example 2:

Input: nums = [-1,0,3,5,9,12], target = 2

Output: -1

Explanation: 2 does not exist in nums so return -1

Constraints:

1 <= nums.length <= 104

-104 < nums[i], target < 104

All the integers in nums are unique.

nums is sorted in ascending order.

Q:-9 EASSY Find First and Last Position of Element in Sorted Array.

Given an array of integers nums sorted in non-decreasing order, find the starting and ending position of a given target value.

If target is not found in the array, return [-1, -1].

You must write an algorithm with O(log n) runtime complexity.

Example 1:

Input: nums = [5,7,7,8,8,10], target = 8

Output: [3,4]

Example 2:

Input: nums = [5,7,7,8,8,10], target = 6

Output: [-1,-1]

Example 3:

Input: nums = [], target = 0

Output: [-1,-1]

Constraints:

0 <= nums.length <= 105

-109 <= nums[i] <= 109

nums is a non-decreasing array.

-109 <= target <= 109

Q:-10 EASSY Sort an Array.

Given an array of integers nums, sort the array in ascending order and return it.

You must solve the problem without using any built-in functions in O(nlog(n)) time complexity and with the smallest space complexity possible.

Example 1:

Input: nums = [5,2,3,1]

Output: [1,2,3,5]

Explanation: After sorting the array, the positions of some numbers are not changed (for example, 2 and 3), while the positions of other numbers are changed (for example, 1 and 5).

Example 2:

Input: nums = [5,1,1,2,0,0]

Output: [0,0,1,1,2,5]

Explanation: Note that the values of nums are not necessarily unique.

Constraints:

1 <= nums.length <= 5 * 104

-5 * 104 <= nums[i] <= 5 * 104

Q:-11 HARD Valid Palindrome.

A phrase is a palindrome if, after converting all uppercase letters into lowercase letters and removing all non-alphanumeric characters, it reads the same forward and backward. Alphanumeric characters include letters and numbers.

Given a string s, return true if it is a palindrome, or false otherwise.

Example 1:

Input: s = "A man, a plan, a canal: Panama"

Output: true

Explanation: "amanaplanacanalpanama" is a palindrome.

Example 2:

Input: s = "race a car"

Output: false

Explanation: "raceacar" is not a palindrome.

Example 3:

Input: s = " "

Output: true

Explanation: s is an empty string "" after removing non-alphanumeric characters.

Since an empty string reads the same forward and backward, it is a palindrome.

Constraints:

1 <= s.length <= 2 * 105

s consists only of printable ASCII characters.

Q:-12 EASSY Reverse Words in a String.

Given an input string s, reverse the order of the words.

A word is defined as a sequence of non-space characters. The words in s will be separated by at least one space.

Return a string of the words in reverse order concatenated by a single space.

Note that s may contain leading or trailing spaces or multiple spaces between two words. The returned string should only have a single space separating the words. Do not include any extra spaces.

Example 1:

Input: s = "the sky is blue"

Output: "blue is sky the"

Example 2:

Input: s = " hello world "

Output: "world hello"

Explanation: Your reversed string should not contain leading or trailing spaces.

Example 3:

Input: s = "a good   example"

Output: "example good a"

Explanation: You need to reduce multiple spaces between two words to a single space in the reversed string.

Constraints:

1 <= s.length <= 104

s contains English letters (upper-case and lower-case), digits, and spaces ' '.

There is at least one word in s.

Q:-13 MEDIUM Count the Number of Vowel Strings in Range.

You are given a 0-indexed array of string words and two integers left and right.

A string is called a vowel string if it starts with a vowel character and ends with a vowel character where vowel characters are 'a', 'e', 'i', 'o', and 'u'.

Return the number of vowel strings words[i] where i belongs to the inclusive range [left, right].

Example 1:

Input: words = ["are","amy","u"], left = 0, right = 2

Output: 2

Explanation:

- "are" is a vowel string because it starts with 'a' and ends with 'e'.

- "amy" is not a vowel string because it does not end with a vowel.

- "u" is a vowel string because it starts with 'u' and ends with 'u'.

The number of vowel strings in the mentioned range is 2.

Example 2:

Input: words = ["hey","aeo","mu","ooo","artro"], left = 1, right = 4

Output: 3

Explanation:

- "aeo" is a vowel string because it starts with 'a' and ends with 'o'.

- "mu" is not a vowel string because it does not start with a vowel.

- "ooo" is a vowel string because it starts with 'o' and ends with 'o'.

- "artro" is a vowel string because it starts with 'a' and ends with 'o'.

The number of vowel strings in the mentioned range is 3.

Constraints:

1 <= words.length <= 1000

1 <= words[i].length <= 10

words[i] consists of only lowercase English letters.

0 <= left <= right < words.length.

Q:-14.EASSY Valid Anagram.

Given two strings s and t, return true if t is an anagram of s, and false otherwise.

Example 1:

Input: s = "anagram", t = "nagaram"

Output: true

Example 2:

Input: s = "rat", t = "car"

Output: false

Constraints:

1 <= s.length, t.length <= 5 * 104

s and t consist of lowercase English letters.

Q:-15 HARD Longest Common Prefix

Write a function to find the longest common prefix string amongst an array of strings.

If there is no common prefix, return an empty string "".

Example 1:

Input: strs = ["flower","flow","flight"]

Output: "fl"

Example 2:

Input: strs = ["dog","racecar","car"]

Output: ""

Explanation: There is no common prefix among the input strings.

Constraints:

1 <= strs.length <= 200

0 <= strs[i].length <= 200

strs[i] consists of only lowercase English letters if it is non-empty.

Q:-16 HARD Count Number of Homogenous Substrings.

Given a string s, return the number of homogenous substrings of s. Since the answer may be too large, return it modulo 109 + 7.

A string is homogenous if all the characters of the string are the same.

A substring is a contiguous sequence of characters within a string.

Example 1:

Input: s = "abbcccaa"

Output: 13

Explanation: The homogenous substrings are listed as below:

"a"   appears 3 times.

"aa" appears 1 time.

"b"   appears 2 times.

"bb" appears 1 time.

"c"   appears 3 times.

"cc" appears 2 times.

"ccc" appears 1 time.

3 + 1 + 2 + 1 + 3 + 2 + 1 = 13.

Example 2:

Input: s = "xy"

Output: 2

Explanation: The homogenous substrings are "x" and "y".

Example 3:

Input: s = "zzzzz"

Output: 15

Constraints:

1 <= s.length <= 105

s consists of lowercase letters.

Q:-17 MEDIUM Climbing Stairs.

You are climbing a staircase. It takes n steps to reach the top.

Each time you can either climb 1 or 2 steps. In how many distinct ways can you climb to the top?

Example 1:

Input: n = 2

Output: 2

Explanation: There are two ways to climb to the top.

1. 1 step + 1 step

2. 2 steps

Example 2:

Input: n = 3

Output: 3

Explanation: There are three ways to climb to the top.

1. 1 step + 1 step + 1 step

2. 1 step + 2 steps

3. 2 steps + 1 step

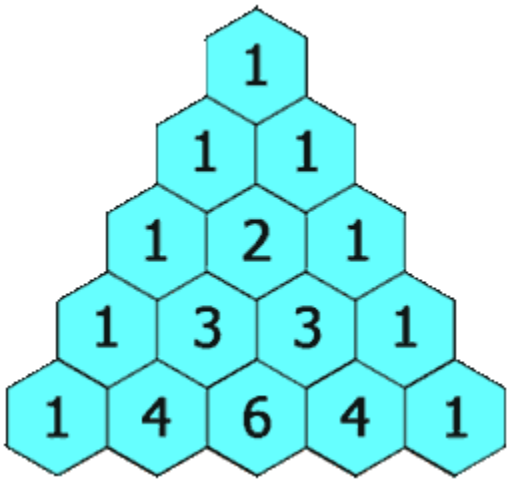
Constraints:

1 <= n <= 45

Q:-18 HARD Pascal's Triangle.

Given an integer numRows, return the first numRows of Pascal's triangle.

In Pascal's triangle, each number is the sum of the two numbers directly above it as shown:



Example 1:

Input: numRows = 5

Output: [[1],[1,1],[1,2,1],[1,3,3,1],[1,4,6,4,1]]

Example 2:

Input: numRows = 1

Output: [[1]]

Constraints:

1 <= numRows <= 30

Q:-19 MEDIUM Best Time to Buy and Sell Stock.

You are given an array prices where prices[i] is the price of a given stock on the ith day.

You want to maximize your profit by choosing a single day to buy one stock and choosing a different day in the future to sell that stock.

Return the maximum profit you can achieve from this transaction. If you cannot achieve any profit, return 0.

Example 1:

Input: prices = [7,1,5,3,6,4]

Output: 5

Explanation: Buy on day 2 (price = 1) and sell on day 5 (price = 6), profit = 6-1 = 5.

Note that buying on day 2 and selling on day 1 is not allowed because you must buy before you sell.

Example 2:

Input: prices = [7,6,4,3,1]

Output: 0

Explanation: In this case, no transactions are done and the max profit = 0.

Constraints:

1 <= prices.length <= 105

0 <= prices[i] <= 104

Q;-20 EASSY Move Zeroes

Given an integer array nums, move all 0's to the end of it while maintaining the relative order of the non-zero elements.

Note that you must do this in-place without making a copy of the array.

Example 1:

Input: nums = [0,1,0,3,12]

Output: [1,3,12,0,0]

Example 2:

Input: nums = [0]

Output: [0]

Constraints:

1 <= nums.length <= 104

-231 <= nums[i] <= 231 – 1

Q21:-HARD 3Sum

Given an integer array nums, return all the triplets [nums[i], nums[j], nums[k]] such that i != j, i != k, and j != k, and nums[i] + nums[j] + nums[k] == 0.

Notice that the solution set must not contain duplicate triplets.

Example 1:

Input: nums = [-1,0,1,2,-1,-4]

Output: [[-1,-1,2],[-1,0,1]]

Explanation:

nums[0] + nums[1] + nums[2] = (-1) + 0 + 1 = 0.

nums[1] + nums[2] + nums[4] = 0 + 1 + (-1) = 0.

nums[0] + nums[3] + nums[4] = (-1) + 2 + (-1) = 0.

The distinct triplets are [-1,0,1] and [-1,-1,2].

Notice that the order of the output and the order of the triplets does not matter.

Example 2:

Input: nums = [0,1,1]

Output: []

Explanation: The only possible triplet does not sum up to 0.

Example 3:

Input: nums = [0,0,0]

Output: [[0,0,0]]

Explanation: The only possible triplet sums up to 0.

Constraints:

3 <= nums.length <= 3000

-105 <= nums[i] <= 105

Q:-22 EASSY Remove Duplicates from Sorted Array.

Given an integer array nums sorted in non-decreasing order, remove the duplicates in-place such that each unique element appears only once. The relative order of the elements should be kept the same. Then return the number of unique elements in nums.

Consider the number of unique elements of nums to be k, to get accepted, you need to do the following things:

Change the array nums such that the first k elements of nums contain the unique elements in the order they were present in nums initially. The remaining elements of nums are not important as well as the size of nums.

Return k.

Custom Judge:

The judge will test your solution with the following code:

```
int[] nums = [...]; // Input array

int[] expectedNums = [...]; // The expected answer with correct length

int k = removeDuplicates(nums); // Calls your implementation

assert k == expectedNums.length;

for (int i = 0; i < k; i++) {

    assert nums[i] == expectedNums[i];

}
```

If all assertions pass, then your solution will be accepted.

Example 1:

Input: nums = [1,1,2]

Output: 2, nums = [1,2,_]

Explanation: Your function should return k = 2, with the first two elements of nums being 1 and 2 respectively.

It does not matter what you leave beyond the returned k (hence they are underscores).

Example 2:

Input: nums = [0,0,1,1,1,2,2,3,3,4]

Output: 5, nums = [0,1,2,3,4,_,_,_,_,_]

Explanation: Your function should return k = 5, with the first five elements of nums being 0, 1, 2, 3, and 4 respectively.

It does not matter what you leave beyond the returned k (hence they are underscores).

Constraints:

1 <= nums.length <= 3 * 104

-100 <= nums[i] <= 100

nums is sorted in non-decreasing order.

Q:-23 HARD Container With Most Water.

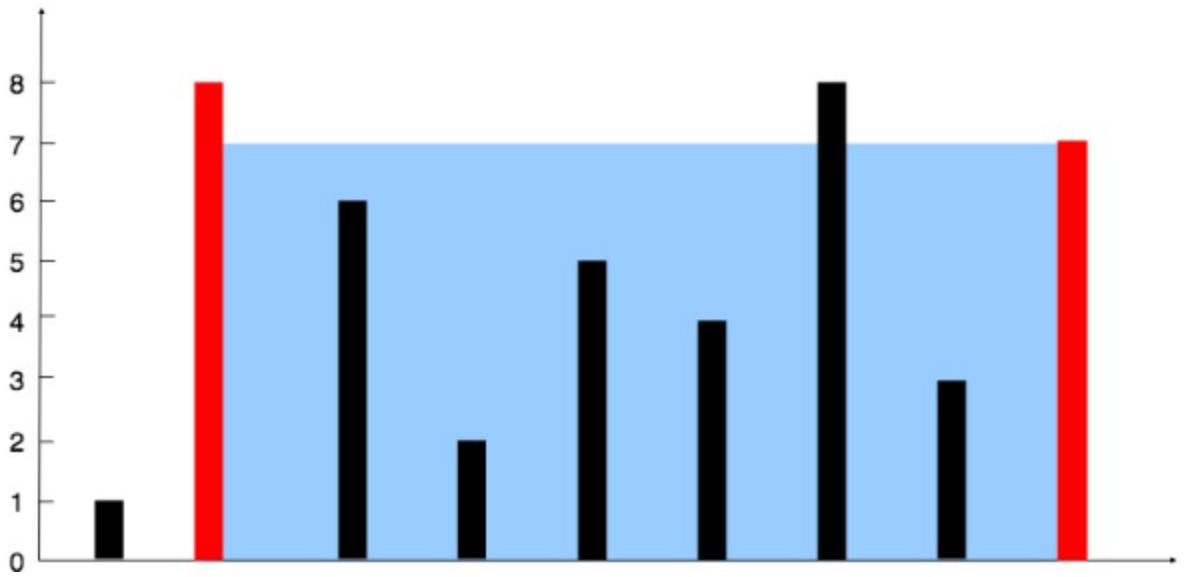
You are given an integer array height of length n. There are n vertical lines drawn such that the two endpoints of the ith line are (i, 0) and (i, height[i]).

Find two lines that together with the x-axis form a container, such that the container contains the most water.

Return the maximum amount of water a container can store.

Notice that you may not slant the container.

Example 1:



Input: height = [1,8,6,2,5,4,8,3,7]

Output: 49

Explanation: The above vertical lines are represented by array [1,8,6,2,5,4,8,3,7]. In this case, the max area of water (blue section) the container can contain is 49.

Example 2:

Input: height = [1,1]

Output: 1

Constraints:

n == height.length

2 <= n <= 105

0 <= height[i] <= 104

Q:-24EASSY Reverse String

Write a function that reverses a string. The input string is given as an array of characters s.

You must do this by modifying the input array in-place with O(1) extra memory.

Example 1:

Input: s = ["h","e","l","l","o"]

Output: ["o","l","l","e","h"]

Example 2:

Input: s = ["H","a","n","n","a","h"]

Output: ["h","a","n","n","a","H"]

Constraints:

1 <= s.length <= 105

s[i] is a printable ascii character.

Q:-25 MEDIUM Reverse Vowels of a String.

Given a string s, reverse only all the vowels in the string and return it.

The vowels are 'a', 'e', 'i', 'o', and 'u', and they can appear in both lower and upper cases, more than once.

Example 1:

Input: s = "IceCreAm"

Output: "AceCreIm"

Explanation:

The vowels in s are ['I', 'e', 'e', 'A']. On reversing the vowels, s becomes "AceCreIm".

Example 2:

Input: s = "leetcode"

Output: "leotcede"

Constraints:

1 <= s.length <= 3 * 105

s consist of printable ASCII characters.

Q:-26 MEDIUM First Unique Character in a String.

Given a string s, find the first non-repeating character in it and return its index. If it does not exist, return -1.

Example 1:

Input: s = "leetcode"

Output: 0

Explanation:

The character 'l' at index 0 is the first character that does not occur at any other index.

Example 2:

Input: s = "loveleetcode"

Output: 2

Example 3:

Input: s = "aabb"

Output: -1

Constraints:

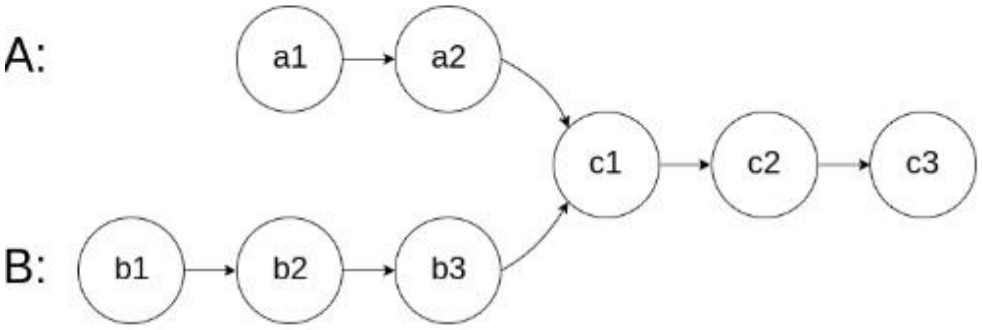
1 <= s.length <= 105

s consists of only lowercase English letters.

Q:-27 MEDIUM Intersection of Two Linked Lists

Given the heads of two singly linked-lists headA and headB, return the node at which the two lists intersect. If the two linked lists have no intersection at all, return null.

For example, the following two linked lists begin to intersect at node c1:



The test cases are generated such that there are no cycles anywhere in the entire linked structure.

Note that the linked lists must retain their original structure after the function returns.

Custom Judge:

The inputs to the judge are given as follows (your program is not given these inputs):

intersectVal - The value of the node where the intersection occurs. This is 0 if there is no intersected node.

listA - The first linked list.

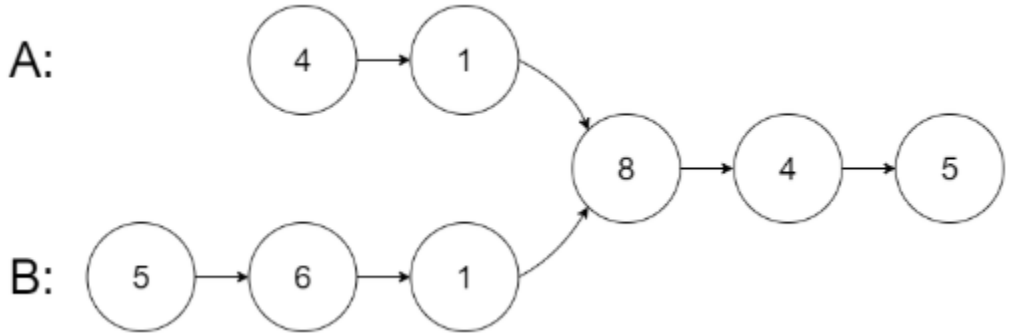
listB - The second linked list.

skipA - The number of nodes to skip ahead in listA (starting from the head) to get to the intersected node.

skipB - The number of nodes to skip ahead in listB (starting from the head) to get to the intersected node.

The judge will then create the linked structure based on these inputs and pass the two heads, headA and headB to your program. If you correctly return the intersected node, then your solution will be accepted.

Example 1:



Input: intersectVal = 8, listA = [4,1,8,4,5], listB = [5,6,1,8,4,5], skipA = 2, skipB = 3

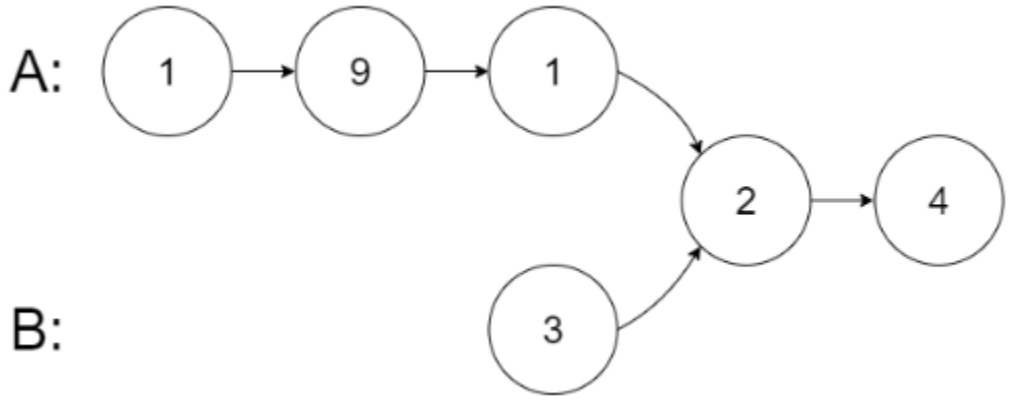
Output: Intersected at '8'

Explanation: The intersected node's value is 8 (note that this must not be 0 if the two lists intersect).

From the head of A, it reads as [4,1,8,4,5]. From the head of B, it reads as [5,6,1,8,4,5]. There are 2 nodes before the intersected node in A; There are 3 nodes before the intersected node in B.

- Note that the intersected node's value is not 1 because the nodes with value 1 in A and B (2nd node in A and 3rd node in B) are different node references. In other words, they point to two different locations in memory, while the nodes with value 8 in A and B (3rd node in A and 4th node in B) point to the same location in memory.

Example 2:



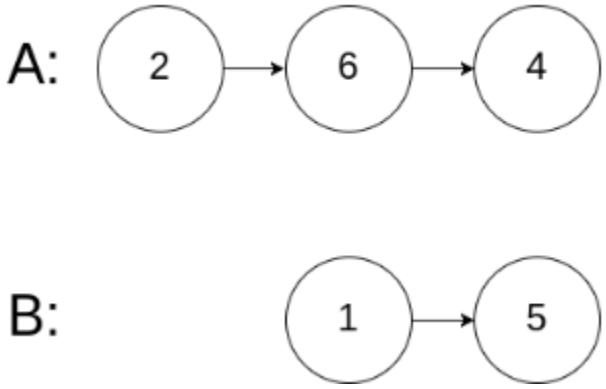
Input: intersectVal = 2, listA = [1,9,1,2,4], listB = [3,2,4], skipA = 3, skipB = 1

Output: Intersected at '2'

Explanation: The intersected node's value is 2 (note that this must not be 0 if the two lists intersect).

From the head of A, it reads as [1,9,1,2,4]. From the head of B, it reads as [3,2,4]. There are 3 nodes before the intersected node in A; There are 1 node before the intersected node in B.

Example 3:



Input: intersectVal = 0, listA = [2,6,4], listB = [1,5], skipA = 3, skipB = 2

Output: No intersection

Explanation: From the head of A, it reads as [2,6,4]. From the head of B, it reads as [1,5]. Since the two lists do not intersect, intersectVal must be 0, while skipA and skipB can be arbitrary values.

Explanation: The two lists do not intersect, so return null.

Constraints:

The number of nodes of listA is in the m.

The number of nodes of listB is in the n.

1 <= m, n <= 3 * 104

1 <= Node.val <= 105

0 <= skipA <= m

0 <= skipB <= n

intersectVal is 0 if listA and listB do not intersect.

intersectVal == listA[skipA] == listB[skipB] if listA and listB intersect.

Q:-28 MEDIUM Middle of the Linked List.

Given the head of a singly linked list, return the middle node of the linked list.

If there are two middle nodes, return the second middle node.

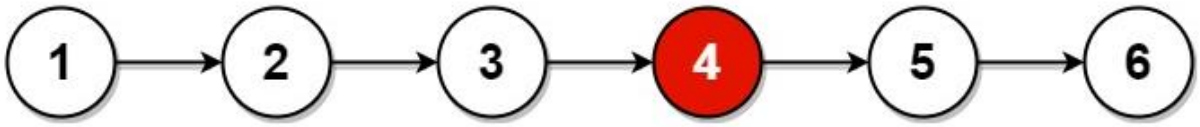
Example 1:

Input: head = [1,2,3,4,5]

Output: [3,4,5]

Explanation: The middle node of the list is node 3.

Example 2:



Input: head = [1,2,3,4,5,6]

Output: [4,5,6]

Explanation: Since the list has two middle nodes with values 3 and 4, we return the second one.

Constraints:

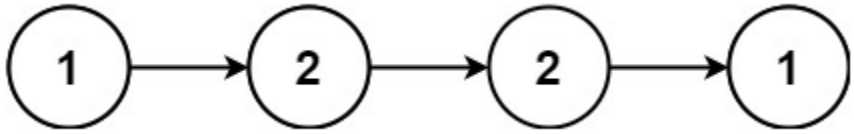
The number of nodes in the list is in the range [1, 100].

1 <= Node.val <= 100

Q:-29 HARD Palindrome Linked List

Given the head of a singly linked list, return true if it is a palindrome or false otherwise.

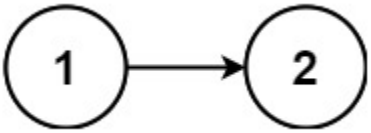
Example 1:



Input: head = [1,2,2,1]

Output: true

Example 2:



Input: head = [1,2]

Output: false

Constraints:

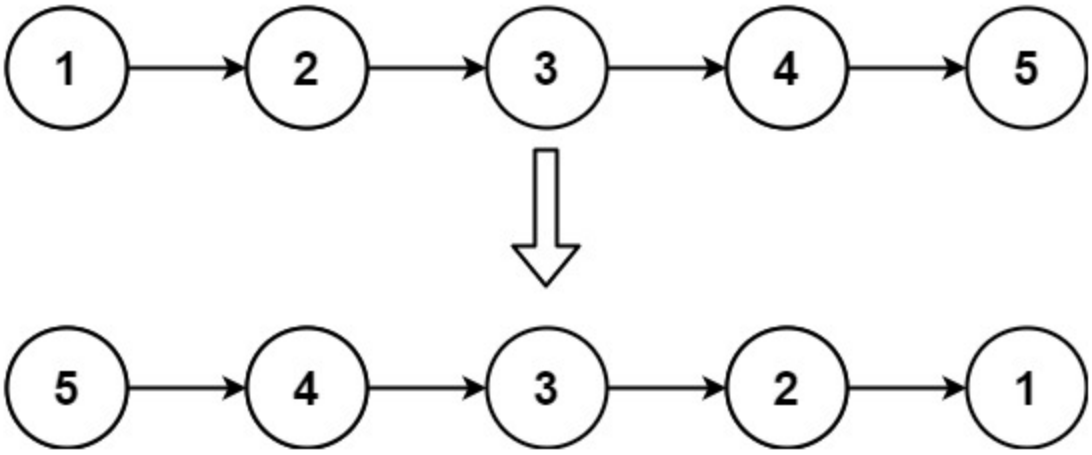
The number of nodes in the list is in the range [1, 105].

0 <= Node.val <= 9

Q:-30 HARD Reverse Linked List

Given the head of a singly linked list, reverse the list, and return the reversed list.

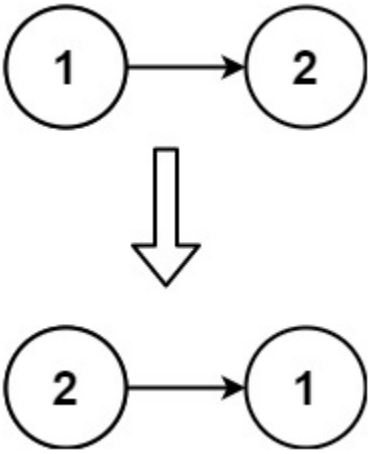
Example 1:



Input: head = [1,2,3,4,5]

Output: [5,4,3,2,1]

Example 2:



Input: head = [1,2]

Output: [2,1]

Example 3:

Input: head = []

Output: []

Constraints:

The number of nodes in the list is the range [0, 5000].

-5000 <= Node.val <= 5000

Q:-31 ESAY Armstrong Numbers

You are given a 3-digit number n, Find whether it is an Armstrong number or not.

An Armstrong number of three digits is a number such that the sum of the cubes of its digits is equal to the number itself. 371 is an Armstrong number since 33 + 73 + 13 = 371.

Examples

Input: n = 153

Output: true

Explanation: 153 is an Armstrong number since $1^3 + 5^3 + 3^3 = 153$.

Input: n = 372

Output: false

Explanation: 372 is not an Armstrong number since $3^3 + 7^3 + 2^3 = 378$.

Input: n = 100

Output: false

Explanation: 100 is not an Armstrong number since $1^3 + 0^3 + 0^3 = 1$.
Constraints:

100 ≤ n <1000

Q:-32 MEDIUM Linked List Cycle

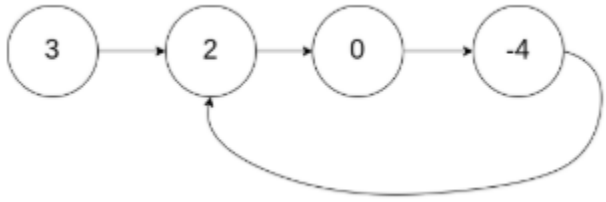
Given head, the head of a linked list, determine if the linked list has a cycle in it.

There is a cycle in a linked list if there is some node in the list that can be reached again by continuously following the next pointer.

Internally, pos is used to denote the index of the node that tail's next pointer is connected to. Note that pos is not passed as a parameter.

Return true if there is a cycle in the linked list. Otherwise, return false.

Example 1:

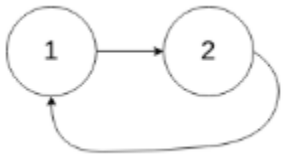


Input: head = [3,2,0,-4], pos = 1

Output: true

Explanation: There is a cycle in the linked list, where the tail connects to the 1st node (0-indexed).

Example 2:



Input: head = [1,2], pos = 0

Output: true

Explanation: There is a cycle in the linked list, where the tail connects to the 0th node.

Example 3:

Input: head = [1], pos = -1

Output: false

Explanation: There is no cycle in the linked list.

Constraints:

The number of the nodes in the list is in the range [0, 104].

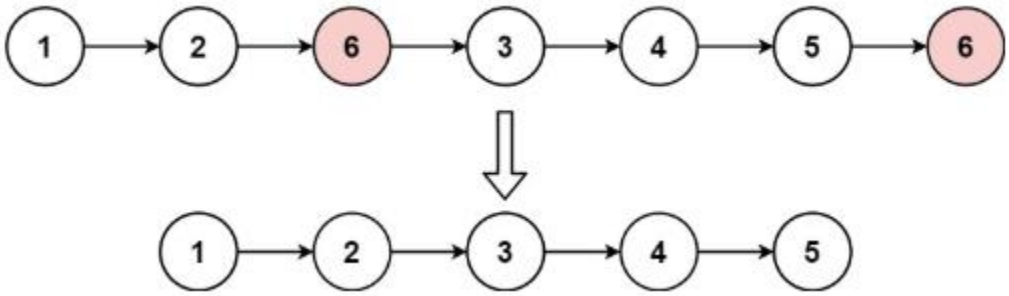
-105 <= Node.val <= 105

pos is -1 or a valid index in the linked-list.

Q:-33 MEDIUM Remove Linked List Elements

Given the head of a linked list and an integer val, remove all the nodes of the linked list that has Node.val == val, and return the new head.

Example 1:



Input: head = [1,2,6,3,4,5,6], val = 6

Output: [1,2,3,4,5]

Example 2:

Input: head = [], val = 1

Output: []

Example 3:

Input: head = [7,7,7,7], val = 7

Output: []

Constraints:

The number of nodes in the list is in the range [0, 104].

1 <= Node.val <= 50

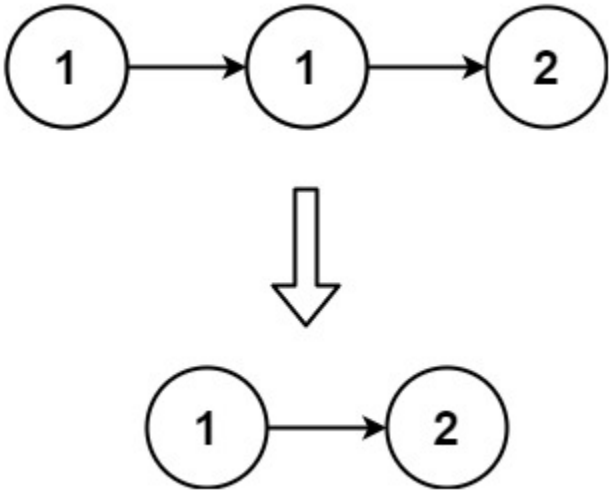
0 <= val <= 50

Q:-34 HARD

Remove Duplicates from Sorted List.

Given the head of a sorted linked list, delete all duplicates such that each element appears only once. Return the linked list sorted as well.

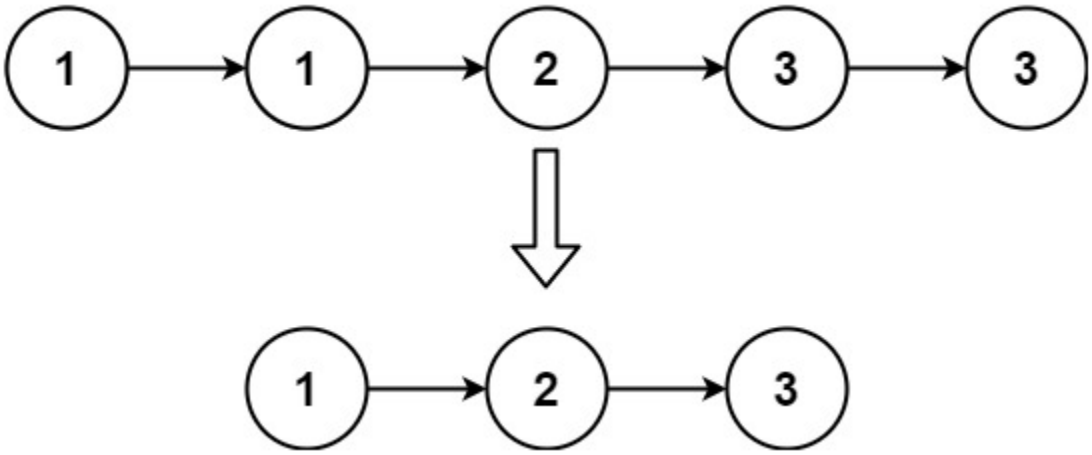
Example 1:



Input: head = [1,1,2]

Output: [1,2]

Example 2:



Input: head = [1,1,2,3,3]

Output: [1,2,3]

Constraints:

The number of nodes in the list is in the range [0, 300].

-100 <= Node.val <= 100

The list is guaranteed to be sorted in ascending order.

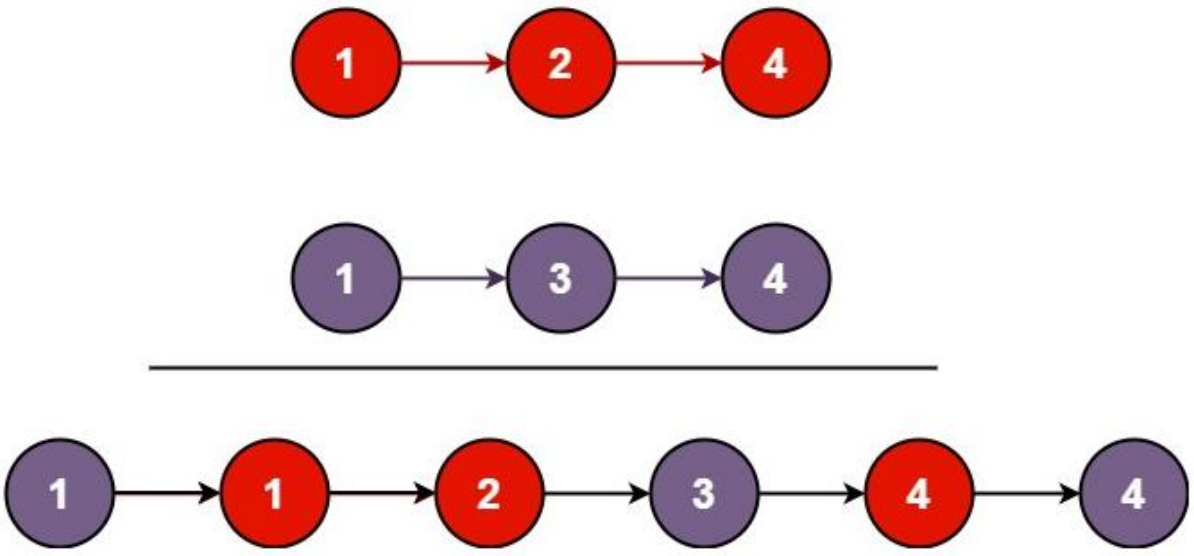
Q:-35 HARD Merge Two Sorted Lists

You are given the heads of two sorted linked lists list1 and list2.

Merge the two lists into one sorted list. The list should be made by splicing together the nodes of the first two lists.

Return the head of the merged linked list.

Example 1:



Input: list1 = [1,2,4], list2 = [1,3,4]

Output: [1,1,2,3,4,4]

Example 2:

Input: list1 = [], list2 = []

Output: []

Example 3:

Input: list1 = [], list2 = [0]

Output: [0]

Constraints:

The number of nodes in both lists is in the range [0, 50].

-100 <= Node.val <= 100

Both list1 and list2 are sorted in non-decreasing order.

Q:-36 HARD Long Pressed Name

Your friend is typing his name into a keyboard. Sometimes, when typing a character c, the key might get long pressed, and the character will be typed 1 or more times.

You examine the typed characters of the keyboard. Return True if it is possible that it was your friends name, with some characters (possibly none) being long pressed.

Example 1:

Input: name = "alex", typed = "aaleex"

Output: true

Explanation: 'a' and 'e' in 'alex' were long pressed.

Example 2:

Input: name = "saeed", typed = "ssaaedd"

Output: false

Explanation: 'e' must have been pressed twice, but it was not in the typed output.

Constraints:

1 <= name.length, typed.length <= 1000

name and typed consist of only lowercase English letters.

Q:-37 MEDIUM Isomorphic Strings.

Given two strings s and t, determine if they are isomorphic.

Two strings s and t are isomorphic if the characters in s can be replaced to get t.

All occurrences of a character must be replaced with another character while preserving the order of characters. No two characters may map to the same character, but a character may map to itself.

Example 1:

Input: s = "egg", t = "add"

Output: true

Explanation:

The strings s and t can be made identical by:

Mapping 'e' to 'a'.

Mapping 'g' to 'd'.

Example 2:

Input: s = "foo", t = "bar"

Output: false

Explanation:

The strings s and t can not be made identical as 'o' needs to be mapped to both 'a' and 'r'.

Example 3:

Input: s = "paper", t = "title"

Output: true

Constraints:

1 <= s.length <= 5 * 104

t.length == s.length

s and t consist of any valid ascii character.

Q:-38 EASSY Is Subsequence.

Given two strings s and t, return true if s is a subsequence of t, or false otherwise.

A subsequence of a string is a new string that is formed from the original string by deleting some (can be none) of the characters without disturbing the relative positions of the remaining characters. (i.e., "ace" is a subsequence of "abcde" while "aec" is not).

Example 1:

Input: s = "abc", t = "ahbgdc"

Output: true

Example 2:

Input: s = "axc", t = "ahbgdc"

Output: false

Constraints:

0 <= s.length <= 100

0 <= t.length <= 104

s and t consist only of lowercase English letters.

Q:-39 HARD Generate Parentheses

Given n pairs of parentheses, write a function to generate all combinations of well-formed parentheses.

Example 1:

Input: n = 3

Output: ["((()))","(()())","(())()","()(())","()()()"]

Example 2:

Input: n = 1

Output: ["()"]

Constraints:

1 <= n <= 8

Q:-40 MEDIUM Merge Sorted Array

You are given two integer arrays nums1 and nums2, sorted in non-decreasing order, and two integers m and n, representing the number of elements in nums1 and nums2 respectively.

Merge nums1 and nums2 into a single array sorted in non-decreasing order.

The final sorted array should not be returned by the function, but instead be stored inside the array nums1. To accommodate this, nums1 has a length of m + n, where the first m elements denote the elements that should be merged, and the last n elements are set to 0 and should be ignored. nums2 has a length of n.

Example 1:

Input: nums1 = [1,2,3,0,0,0], m = 3, nums2 = [2,5,6], n = 3

Output: [1,2,2,3,5,6]

Explanation: The arrays we are merging are [1,2,3] and [2,5,6].

The result of the merge is [1,2,2,3,5,6] with the underlined elements coming from nums1.

Example 2:

Input: nums1 = [1], m = 1, nums2 = [], n = 0

Output: [1]

Explanation: The arrays we are merging are [1] and [].

The result of the merge is [1].

Example 3:

Input: nums1 = [0], m = 0, nums2 = [1], n = 1

Output: [1]

Explanation: The arrays we are merging are [] and [1].

The result of the merge is [1].

Note that because m = 0, there are no elements in nums1. The 0 is only there to ensure the merge result can fit in nums1.

Constraints:

nums1.length == m + n

nums2.length == n

0 <= m, n <= 200

1 <= m + n <= 200

-109 <= nums1[i], nums2[j] <= 109

Q:-41 HARD Roman to Integer

Roman numerals are represented by seven different symbols: I, V, X, L, C, D and M.

Symbol	Value
I	1
V	5
X	10

L	50
C	100
D	500
M	1000

For example, 2 is written as II in Roman numeral, just two ones added together. 12 is written as XII, which is simply X + II. The number 27 is written as XXVII, which is XX + V + II.

Roman numerals are usually written largest to smallest from left to right. However, the numeral for four is not IIII. Instead, the number four is written as IV. Because the one is before the five we subtract it making four. The same principle applies to the number nine, which is written as IX. There are six instances where subtraction is used:

I can be placed before V (5) and X (10) to make 4 and 9.

X can be placed before L (50) and C (100) to make 40 and 90.

C can be placed before D (500) and M (1000) to make 400 and 900.

Given a roman numeral, convert it to an integer.

Example 1:

Input: s = "III"

Output: 3

Explanation: III = 3.

Example 2:

Input: s = "LVIII"

Output: 58

Explanation: L = 50, V= 5, III = 3.

Example 3:

Input: s = "MCMXCIV"

Output: 1994

Explanation: M = 1000, CM = 900, XC = 90 and IV = 4.

Constraints:

1 <= s.length <= 15

s contains only the characters ('I', 'V', 'X', 'L', 'C', 'D', 'M').

It is guaranteed that s is a valid roman numeral in the range [1, 3999].

Q:-42 MEDIUM Plus One

You are given a large integer represented as an integer array digits, where each digits[i] is the ith digit of the integer. The digits are ordered from most significant to least significant in left-to-right order. The large integer does not contain any leading 0's.

Increment the large integer by one and return the resulting array of digits.

Example 1:

Input: digits = [1,2,3]

Output: [1,2,4]

Explanation: The array represents the integer 123.

Incrementing by one gives 123 + 1 = 124.

Thus, the result should be [1,2,4].

Example 2:

Input: digits = [4,3,2,1]

Output: [4,3,2,2]

Explanation: The array represents the integer 4321.

Incrementing by one gives 4321 + 1 = 4322.

Thus, the result should be [4,3,2,2].

Example 3:

Input: digits = [9]

Output: [1,0]

Explanation: The array represents the integer 9.

Incrementing by one gives 9 + 1 = 10.

Thus, the result should be [1,0].

Constraints:

1 <= digits.length <= 100

0 <= digits[i] <= 9

digits does not contain any leading 0's.

Q:-43 MEDIUM Missing Number

Given an array nums containing n distinct numbers in the range [0, n], return the only number in the range that is missing from the array.

Example 1:

Input: nums = [3,0,1]

Output: 2

Explanation:

n = 3 since there are 3 numbers, so all numbers are in the range [0,3].
2 is the missing number in the range since it does not appear in nums.

Example 2:

Input: nums = [0,1]

Output: 2

Explanation:

n = 2 since there are 2 numbers, so all numbers are in the range [0,2].
2 is the missing number in the range since it does not appear in nums.

Example 3:

Input: nums = [9,6,4,2,3,5,7,0,1]

Output: 8

Explanation:

n = 9 since there are 9 numbers, so all numbers are in the range [0,9]. 8 is the missing number in the range since it does not appear in nums.

Constraints:

n == nums.length

1 <= n <= 104

0 <= nums[i] <= n

All the numbers of nums are unique.

Q:-44 MEDIUM Add Strings

Given two non-negative integers, num1 and num2 represented as string, return the sum of num1 and num2 as a string.

You must solve the problem without using any built-in library for handling large integers (such as BigInteger). You must also not convert the inputs to integers directly.

Example 1:

Input: num1 = "11", num2 = "123"

Output: "134"

Example 2:

Input: num1 = "456", num2 = "77"

Output: "533"

Example 3:

Input: num1 = "0", num2 = "0"

Output: "0"

Constraints:

1 <= num1.length, num2.length <= 104

num1 and num2 consist of only digits.

num1 and num2 don't have any leading zeros except for the zero itself.

Q:-45 EASSY Maximum Product of Three Numbers

Given an integer array nums, find three numbers whose product is maximum and return the maximum product.

Example 1:

Input: nums = [1,2,3]

Output: 6

Example 2:

Input: nums = [1,2,3,4]

Output: 24

Example 3:

Input: nums = [-1,-2,-3]

Output: -6

Constraints:

3 <= nums.length <= 104

-1000 <= nums[i] <= 1000

Q:-46 MEDIUM Happy Number

Write an algorithm to determine if a number n is happy.

A happy number is a number defined by the following process:

Starting with any positive integer, replace the number by the sum of the squares of its digits.

Repeat the process until the number equals 1 (where it will stay), or it loops endlessly in a cycle which does not include 1.

Those numbers for which this process ends in 1 are happy.

Return true if n is a happy number, and false if not.

Example 1:

Input: n = 19

Output: true

Explanation:

12 + 92 = 82

82 + 22 = 68

62 + 82 = 100

12 + 02 + 02 = 1

Example 2:

Input: n = 2

Output: false

Constraints:

1 <= n <= 231 - 1

Q:-47 MEDIUM Search Insert Position

Given a sorted array of distinct integers and a target value, return the index if the target is found. If not, return the index where it would be if it were inserted in order.

You must write an algorithm with O(log n) runtime complexity.

Example 1:

Input: nums = [1,3,5,6], target = 5

Output: 2

Example 2:

Input: nums = [1,3,5,6], target = 2

Output: 1

Example 3:

Input: nums = [1,3,5,6], target = 7

Output: 4

Constraints:

1 <= nums.length <= 104

-104 <= nums[i] <= 104

nums contains distinct values sorted in ascending order.

-104 <= target <= 104

Q:-48 EASSY Add Digits

Given an integer num, repeatedly add all its digits until the result has only one digit, and return it.

Example 1:

Input: num = 38

Output: 2

Explanation: The process is

38 --> 3 + 8 --> 11

11 --> 1 + 1 --> 2

Since 2 has only one digit, return it.

Example 2:

Input: num = 0

Output: 0

Constraints:

0 <= num <= 231 – 1

Q:-49.EASSY Length of Last Word

Given a string s consisting of words and spaces, return the length of the last word in the string.

A word is a maximal substring consisting of non-space characters only.

Example 1:

Input: s = "Hello World"

Output: 5

Explanation: The last word is "World" with length 5.

Example 2:

Input: s = "   fly me   to   the moon "

Output: 4

Explanation: The last word is "moon" with length 4.

Example 3:

Input: s = "luffy is still joyboy"

Output: 6

Explanation: The last word is "joyboy" with length 6.

Constraints:

1 <= s.length <= 104

s consists of only English letters and spaces ' '.

There will be at least one word in s.

Q:-50 EASSY Find Numbers with Even Number of Digits.

Given an array nums of integers, return how many of them contain an even number of digits.

Example 1:

Input: nums = [12,345,2,6,7896]

Output: 2

Explanation:

12 contains 2 digits (even number of digits).

345 contains 3 digits (odd number of digits).

2 contains 1 digit (odd number of digits).

6 contains 1 digit (odd number of digits).

7896 contains 4 digits (even number of digits).

Therefore only 12 and 7896 contain an even number of digits.

Example 2:

Input: nums = [555,901,482,1771]

Output: 1

Explanation:

Only 1771 contains an even number of digits.

Constraints:

1 <= nums.length <= 500

1 <= nums[i] <= 105