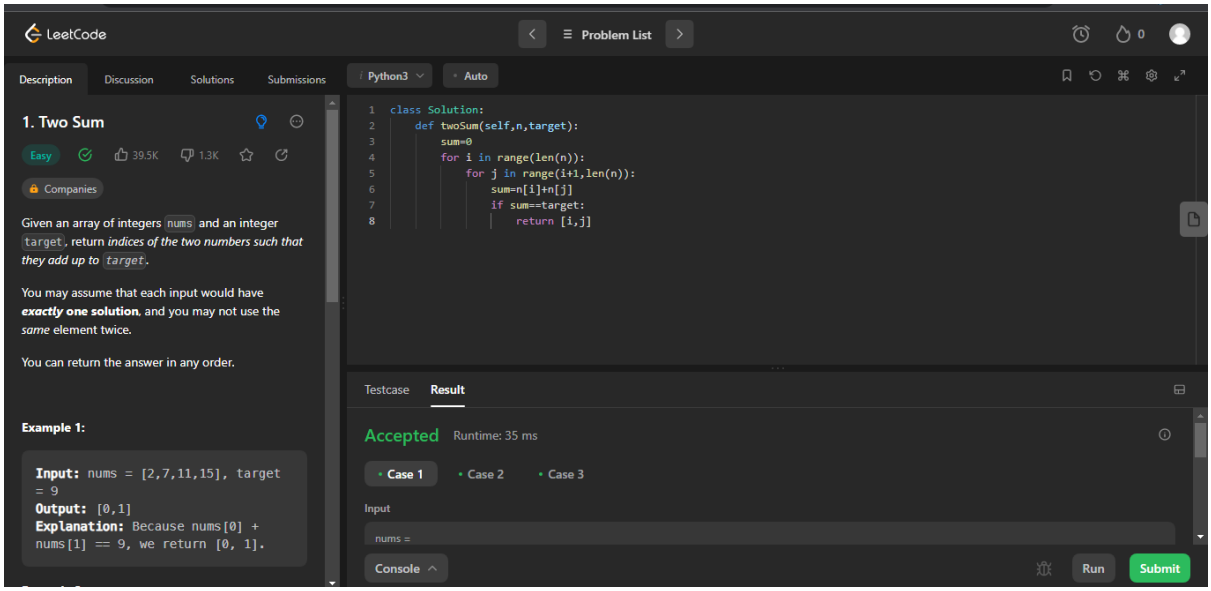
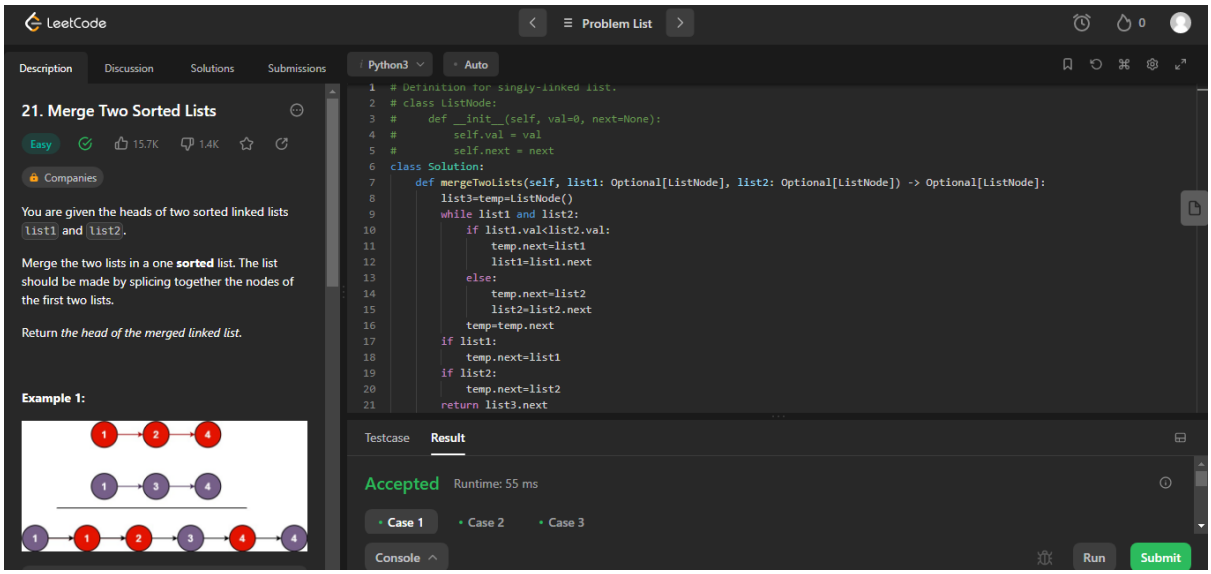
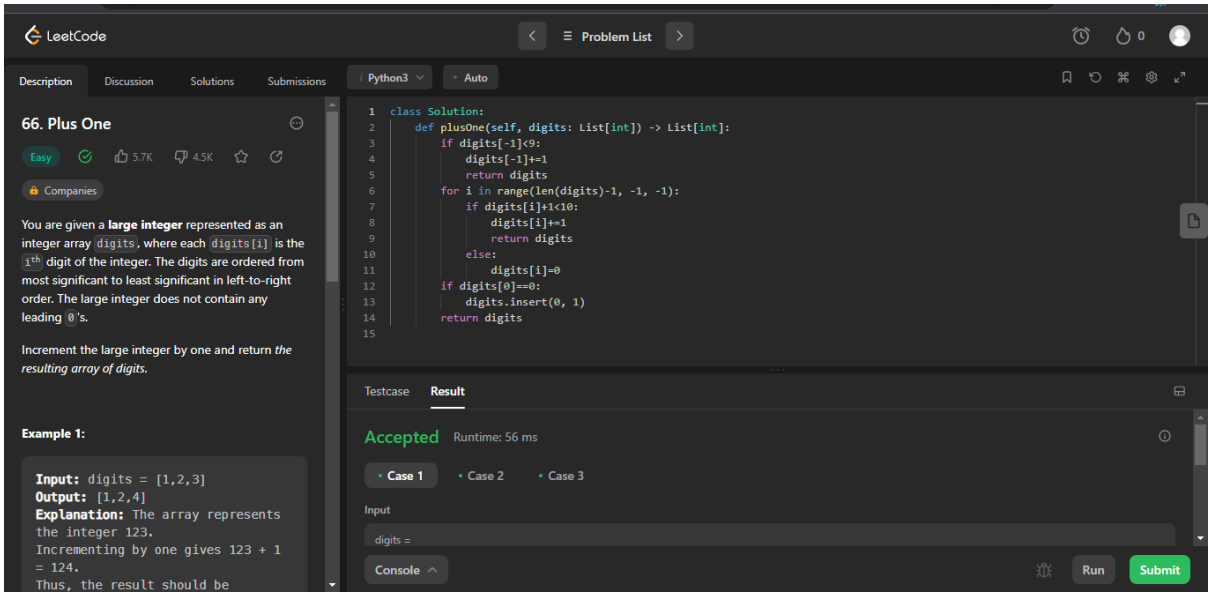
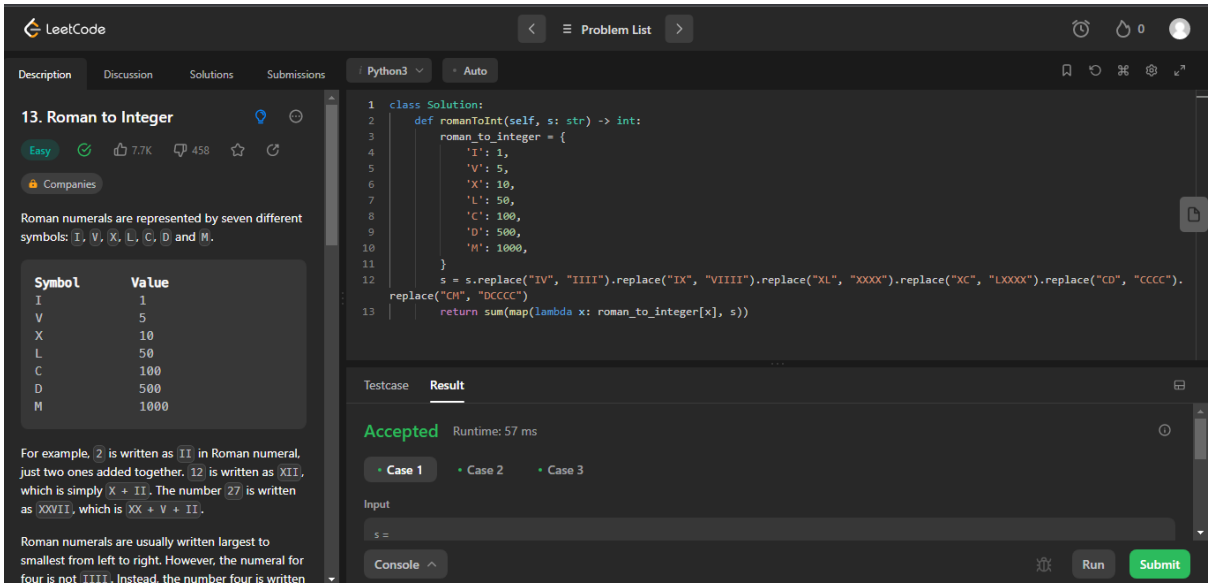
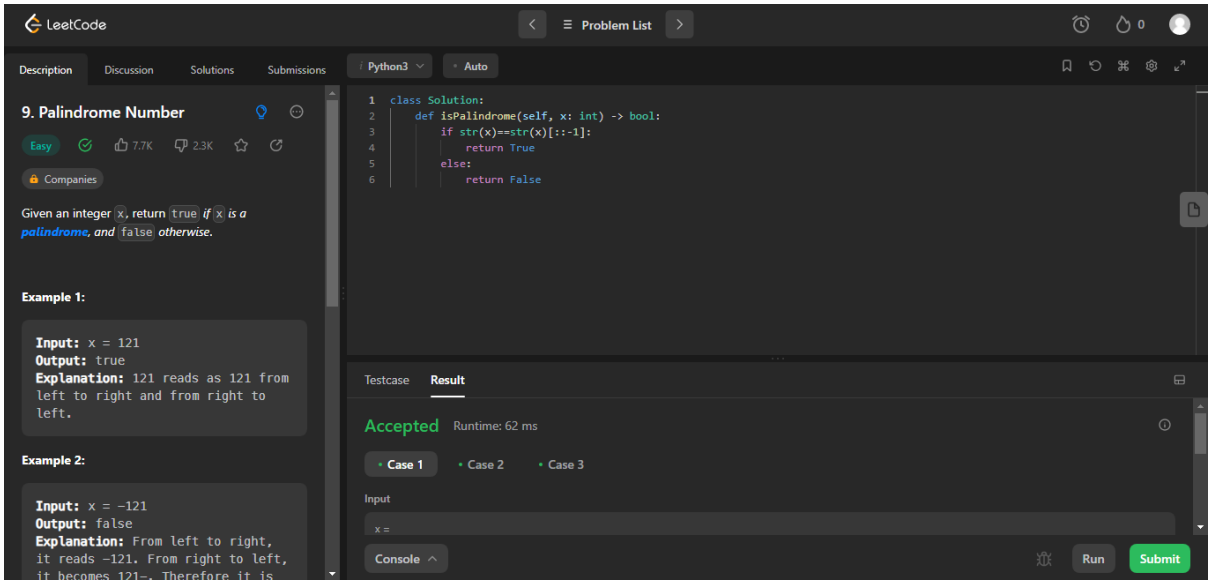

LeetCode
Problem List
Description
Discussion
Solutions
Submissions
Python3
Auto
1. Two Sum
Easy
39.5K
1.3K
Companies
Given an array of integers nums and an integer target, return indices of the two numbers such that they add up to target.
You may assume that each input would have exactly one solution, 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].
class Solution:
def twoSum(self,n,target):
sum=0
for i in range(len(n)):
for j in range(i+1,len(n)):
sum=n[i]+n[j]
if sum==target:
return [i,j]
Testcase
Result
Accepted
Runtime: 35 ms
Case 1
Case 2
Case 3
Input
nums =
Console
Run
Submit


LeetCode
Problem List
Description
Discussion
Solutions
Submissions
Python3
Auto
21. Merge Two Sorted Lists
Easy
15.7K
1.4K
Companies
You are given the heads of two sorted linked lists list1 and list2.
Merge the two lists in a 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:
# Definition for singly-linked list.
# class ListNode:
# def __init__(self, val=0, next=None):
# self.val = val
# self.next = next
class Solution:
def mergeTwoLists(self, list1: Optional[ListNode], list2: Optional[ListNode]) -> Optional[ListNode]:
list3=temp=ListNode()
while list1 and list2:
if list1.val<list2.val:
temp.next=list1
list1=list1.next
else:
temp.next=list2
list2=list2.next
temp=temp.next
if list1:
temp.next=list1
if list2:
temp.next=list2
return list3.next
Testcase
Result
Accepted
Runtime: 55 ms
Case 1
Case 2
Case 3
Console
Run
Submit


LeetCode
Problem List
Description
Discussion
Solutions
Submissions
Python3
Auto
66. Plus One
Easy
5.7K
4.5K
Companies
You are given a large integer represented as an integer array digits, where each digits[i] is the 1th 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
class Solution:
def plusOne(self, digits: List[int]) -> List[int]:
if digits[-1]<9:
digits[-1]+=1
return digits
for i in range(len(digits)-1, -1, -1):
if digits[i]+1<10:
digits[i]+=1
return digits
else:
digits[i]=0
if digits[0]==0:
digits.insert(0, 1)
return digits
Testcase
Result
Accepted
Runtime: 56 ms
Case 1
Case 2
Case 3
Input
digits =
Console
Run
Submit


LeetCode
Problem List
Description
Discussion
Solutions
Submissions
Python3
Auto
13. Roman to Integer
Easy
7.7K
458
Companies
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
class Solution:
def romanToInt(self, s: str) -> int:
roman_to_integer = {
'I': 1,
'V': 5,
'X': 10,
'L': 50,
'C': 100,
'D': 500,
'M': 1000,
}
s = s.replace("IV", "IIII").replace("IX", "VIIII").replace("XL", "XXXX").replace("XC", "LXXXX").replace("CD", "CCCC").replace("CM", "DCCCC")
return sum(map(lambda x: roman_to_integer[x], s))
Testcase
Result
Accepted
Runtime: 57 ms
Case 1
Case 2
Case 3
Input
s =
Console
Run
Submit


LeetCode
Problem List
Description
Discussion
Solutions
Submissions
Python3
Auto
9. Palindrome Number
Easy
7.7K
2.3K
Companies
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
class Solution:
def isPalindrome(self, x: int) -> bool:
if str(x)==str(x)[::-1]:
return True
else:
return False
Testcase
Result
Accepted
Runtime: 62 ms
Case 1
Case 2
Case 3
Input
x =
Console
Run
Submit