Leet code

1.Two sum:

class Solution(object):

    def twoSum(self, nums, target):

        """

        :type nums: List[int]

        :type target: int

        :rtype: List[int]

        """

        num\_map = {}  # Create a hash map to store numbers and their indices

        for i, num in enumerate(nums):

            complement = target - num

            if complement in num\_map:

                # If the complement is found in the map, return its index and the current index

                return [num\_map[complement], i]

            else:

                # Otherwise, add the current number and its index to the map

                num\_map[num] = i

        # This part should ideally not be reached if a solution is guaranteed

        return []

2.add two numbers:

class Solution(object):  
 def addTwoNumbers(self, l1, l2):  
 dummy = ListNode(0)  
 current = dummy  
 carry = 0  
  
 while l1 is not None or l2 is not None:  
 x = l1.val if l1 is not None else 0  
 y = l2.val if l2 is not None else 0  
 sum = carry + x + y  
 carry = sum // 10  
 current.next = ListNode(sum % 10)  
 current = current.next  
  
 if l1 is not None: l1 = l1.next  
 if l2 is not None: l2 = l2.next  
  
 if carry > 0:  
 current.next = ListNode(carry)  
  
 return dummy.next