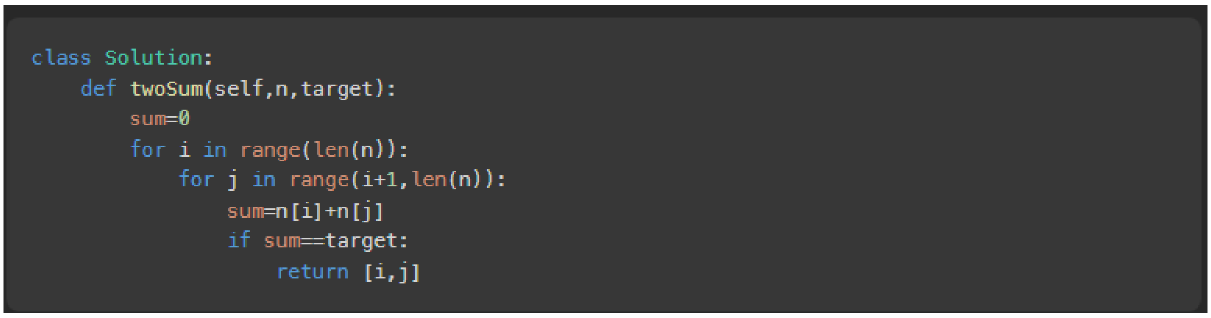
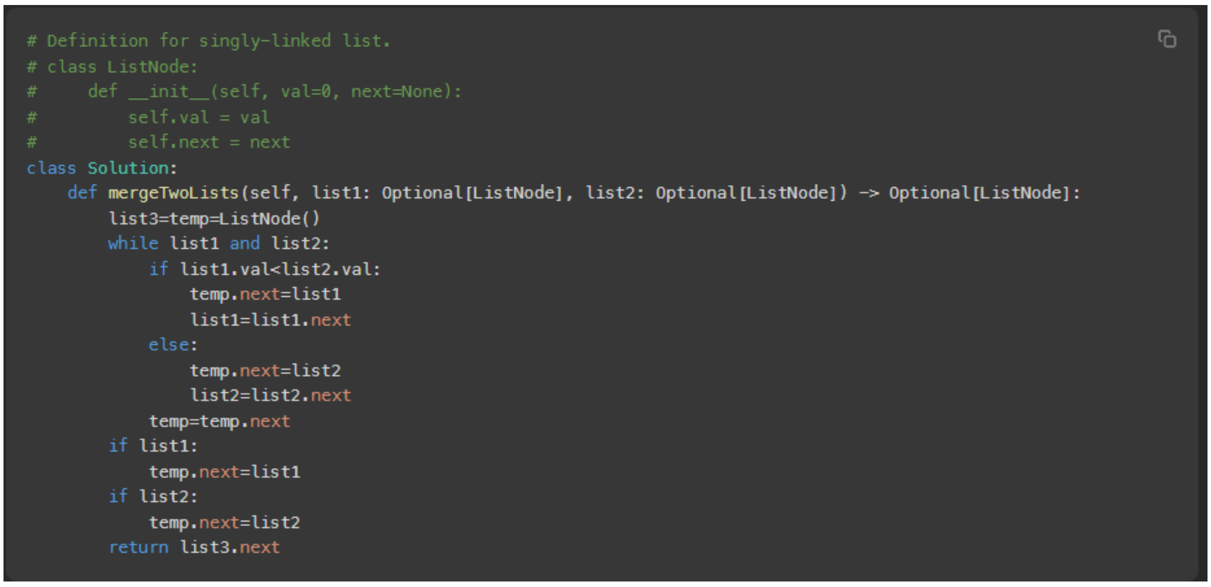
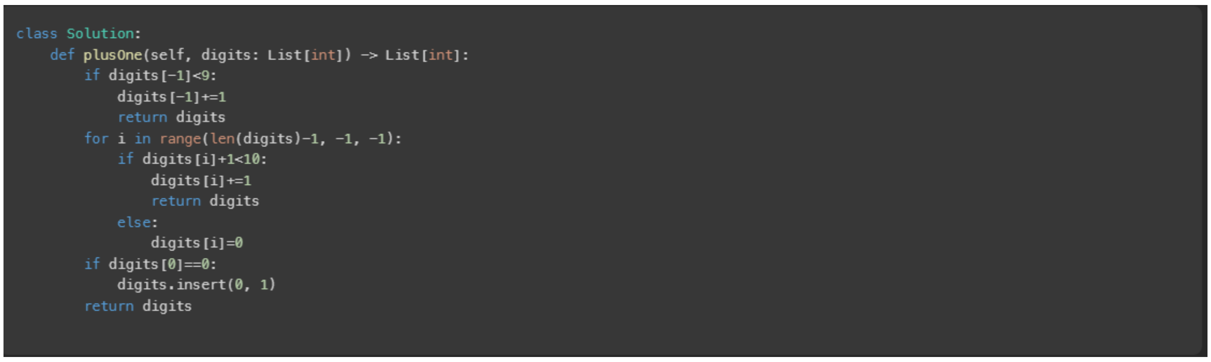


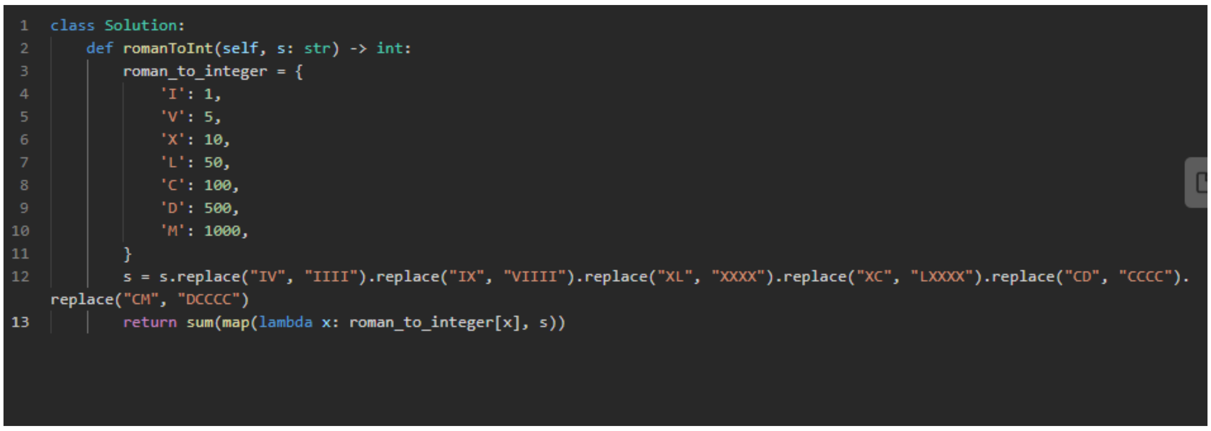
```python
class Solution:
    def twoSum(self,n,target):
        sum=0
        for i in range(len(n)):
            for j in range(i+1,len(n)):
                sum=n[i]+n[j]
                if sum==target:
                    return [i,j]
```



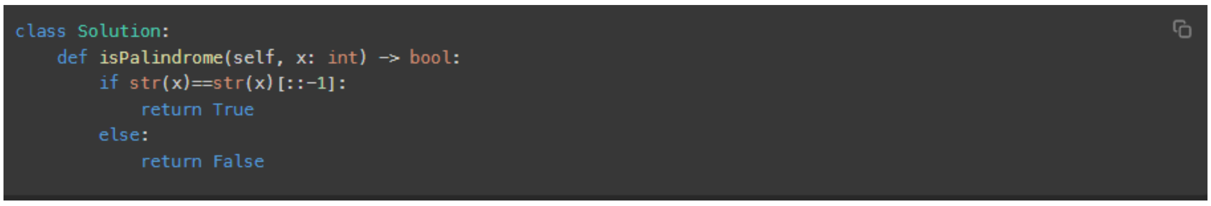
```python
# Definition for singly-linked list.
# class ListNode:
#     def __init__(self, val=0, next=None):
#         self.val = val
#         self.next = next
class Solution:
    def mergeTwoLists(self, list1: Optional[ListNode], list2: Optional[ListNode]) -> Optional[ListNode]:
        list3=temp=ListNode()
        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
```



```python
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
```



```python
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))
```



```python
class Solution:
    def isPalindrome(self, x: int) -> bool:
        if str(x)==str(x)[::-1]:
            return True
        else:
            return False
```