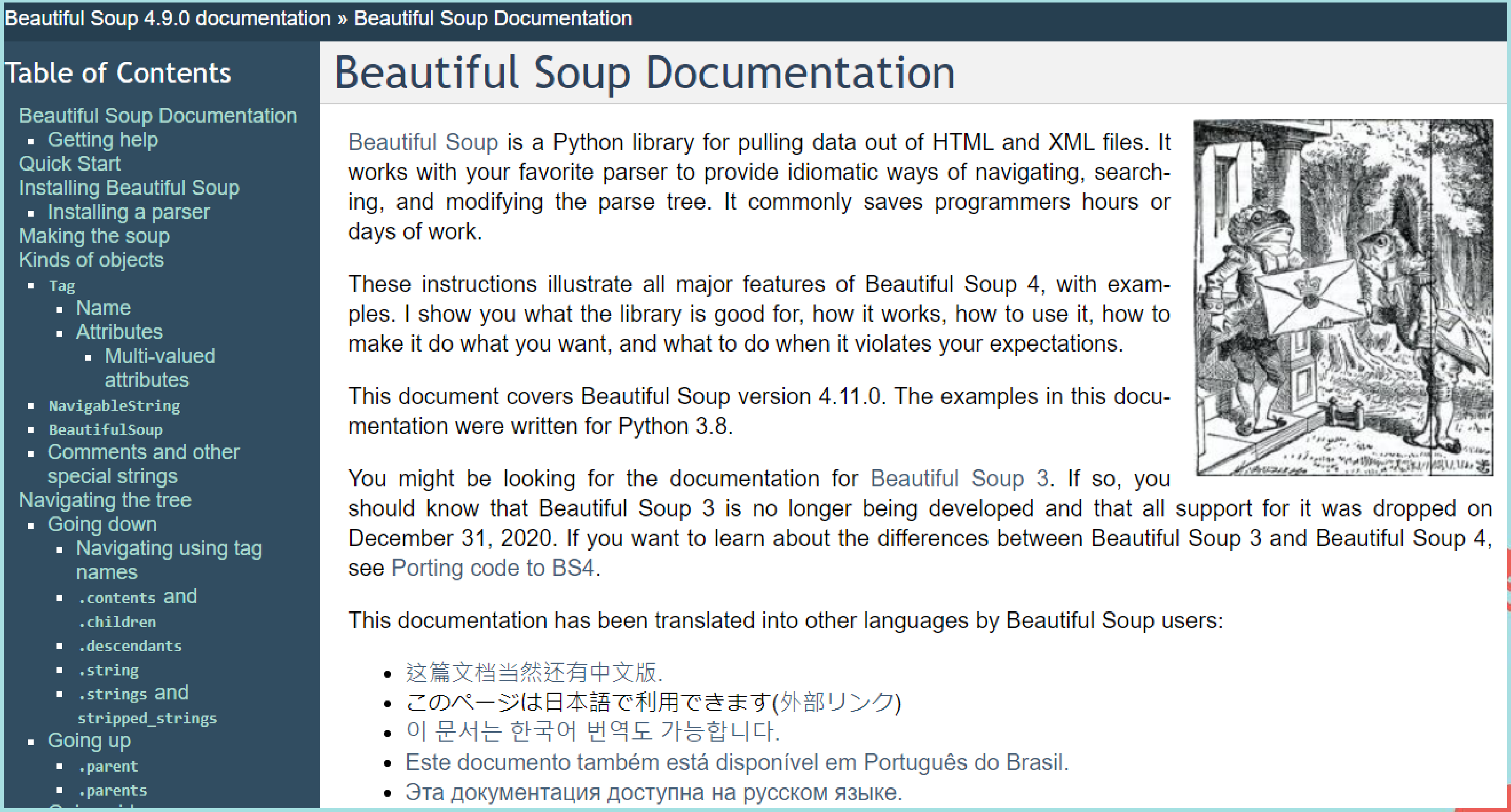
2023-Spring
Advanced Computer Programming (3)

CSIE, Asia Univ.

Course schedule

- W1-Introduction
- W2-Python libraries
- W3-BeautifulSoup(1)
- W4-BeautifulSoup(2)
- W5-Scrapy(1)
- W6-Scrapy(2)
- W7-Storing Data
- W8-Project development(1)
- W9-Midterm presentation
- W10-Web & HTTP
- W11-Flask
- W12-Flask Routes
- W13-Jinja template
- W14-Flask-form
- W15-Flask-mail
- W16-REST API
- W17-Project development(2)
- W18-Final presentation

Beautiful Soup



Beautiful Soup 4.9.0 documentation » Beautiful Soup Documentation

Table of Contents

Beautiful Soup Documentation
- Getting help

Quick Start
Installing Beautiful Soup
- Installing a parser

Making the soup
Kinds of objects
- `Tag`
  - Name
  - Attributes
    - Multi-valued attributes
- `NavigableString`
- `BeautifulSoup`
- Comments and other special strings

Navigating the tree
- Going down
  - Navigating using tag names
  - `.contents` and `.children`
  - `.descendants`
  - `.string`
  - `.strings` and `stripped_strings`
- Going up
  - `.parent`
  - `.parents`

Beautiful Soup Documentation

Beautiful Soup is a Python library for pulling data out of HTML and XML files. It works with your favorite parser to provide idiomatic ways of navigating, searching, and modifying the parse tree. It commonly saves programmers hours or days of work.

These instructions illustrate all major features of Beautiful Soup 4, with examples. I show you what the library is good for, how it works, how to use it, how to make it do what you want, and what to do when it violates your expectations.

This document covers Beautiful Soup version 4.11.0. The examples in this documentation were written for Python 3.8.

You might be looking for the documentation for Beautiful Soup 3. If so, you should know that Beautiful Soup 3 is no longer being developed and that all support for it was dropped on December 31, 2020. If you want to learn about the differences between Beautiful Soup 3 and Beautiful Soup 4, see Porting code to BS4.

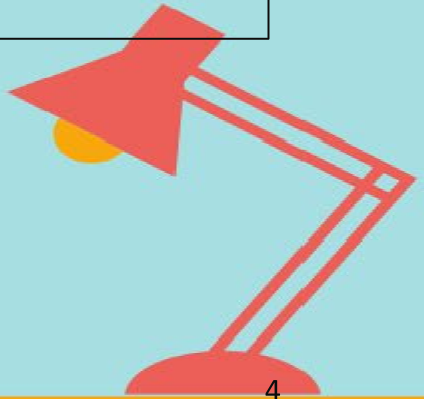
This documentation has been translated into other languages by Beautiful Soup users:

- 这篇文档当然还有中文版.
- このページは日本語で利用できます(外部リンク)
- 이 문서는 한국어 번역도 가능합니다.
- Este documento também está disponível em Português do Brasil.
- Эта документация доступна на русском языке.

https://www.crummy.com/software/BeautifulSoup/bs4/doc/

Regular Expressions and BeautifulSoup

```
from urllib.request import urlopen
from bs4 import BeautifulSoup
import re

html = urlopen('http://www.pythonscraping.com/pages/page3.html')
bs = BeautifulSoup(html, 'html.parser')
images = bs.find_all('img', {'src':re.compile('\.\.\/img\/gifts/img.*\.jpg')})
for image in images:
    print(image['src'])
```



Lambda Expressions

function definition

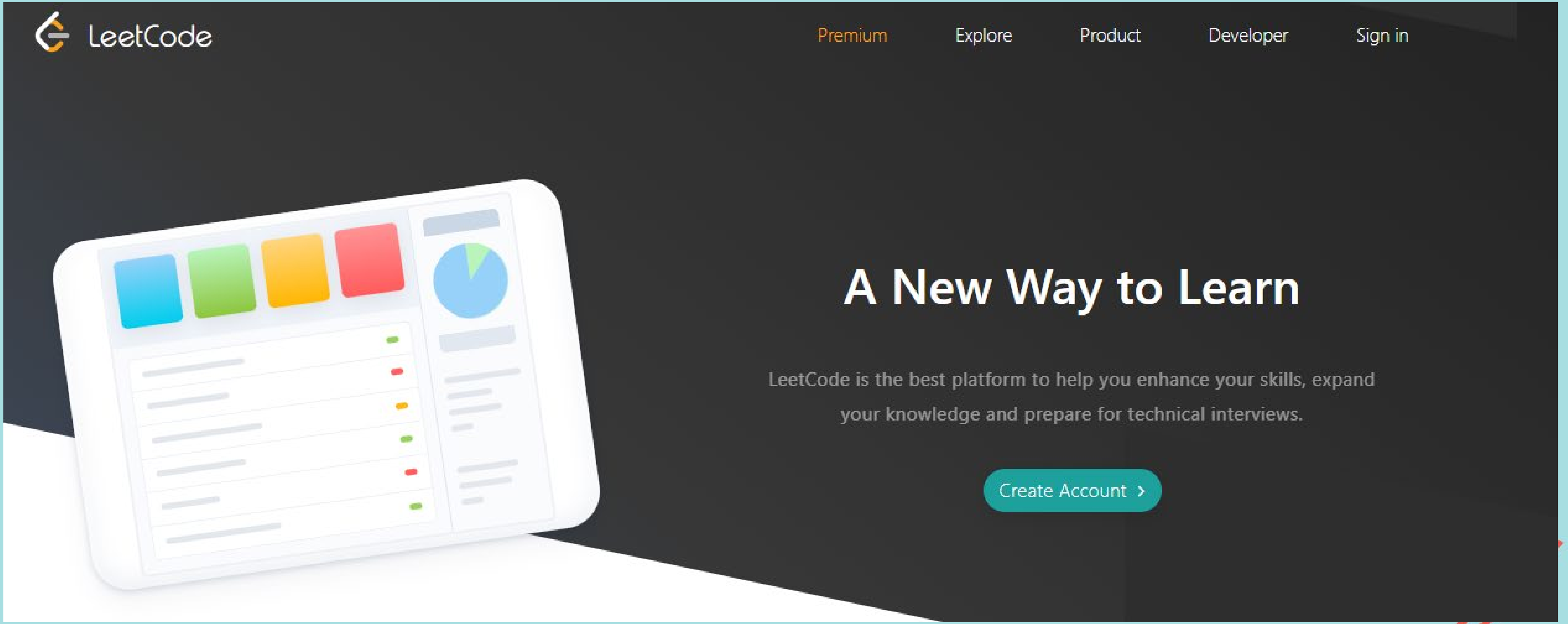
```
def max(m, n):
    return m if m > n else n

print(max(10, 3)) # 顯示 10
```

Lambda function

```
max = lambda m, n: m if m > n else n
print(max(10, 3)) # 顯示 10
```

```
bs.find_all(lambda tag: len(tag.attrs) == 2)

bs.find_all(lambda tag: tag.get_text() == 'Or maybe he\'s only resting?')
```

leetcode



A New Way to Learn

LeetCode is the best platform to help you enhance your skills, expand your knowledge and prepare for technical interviews.

https://leetcode.com/

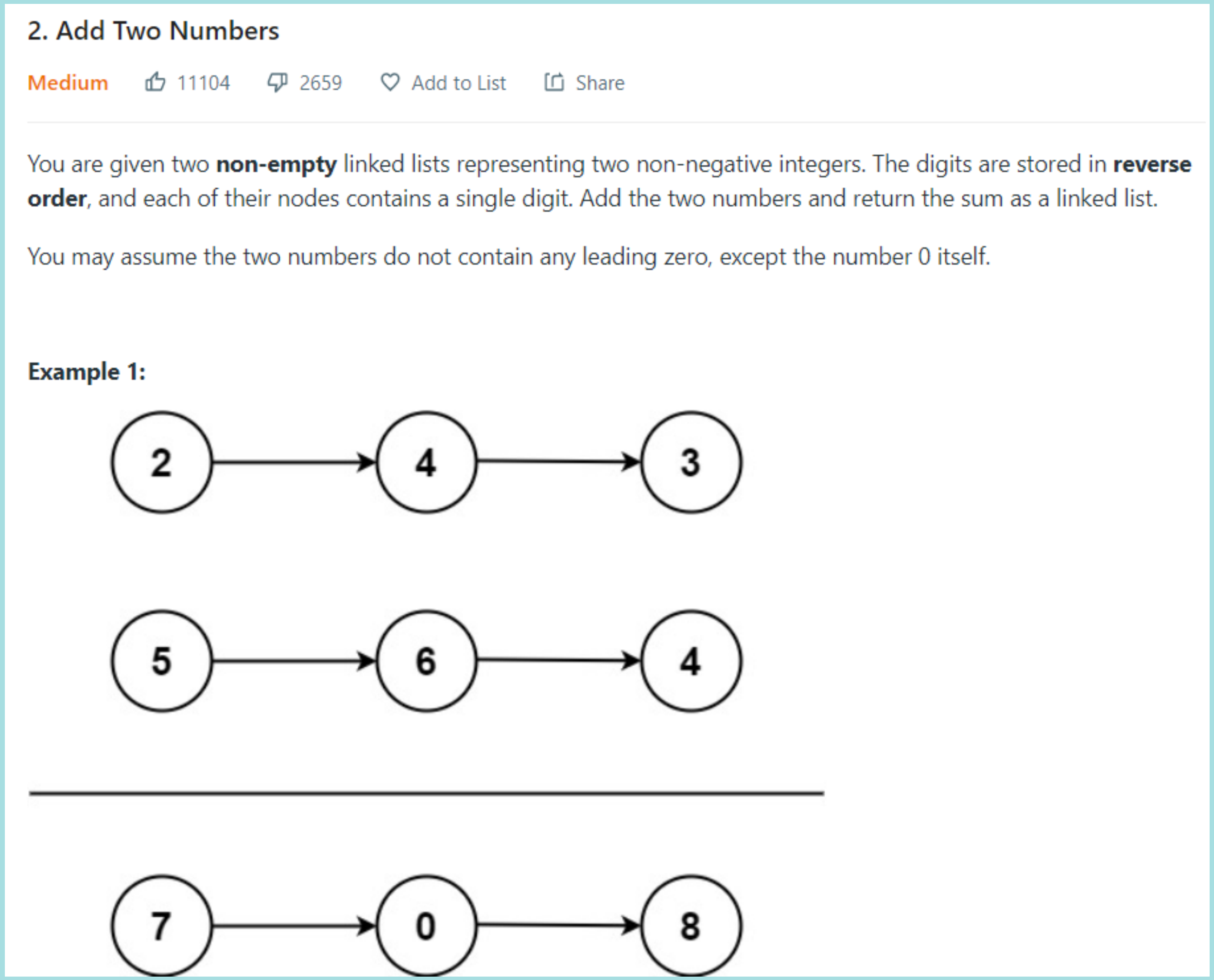
Leetcode problem: **Two Sum**

- 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.
- https://leetcode-cn.com/problems/two-sum

Leetcode problem: **Three Sum**

- Given an array nums of n integers, are there elements a, b, c in nums such that a + b + c = 0?
- Find all unique triplets in the array which gives the sum of zero.
- Notice that the solution set must not contain duplicate triplets.
- You can return the answer in any order.
- https://leetcode.com/problems/3sum/

Leetcode problem: Add Two Numbers



2. Add Two Numbers

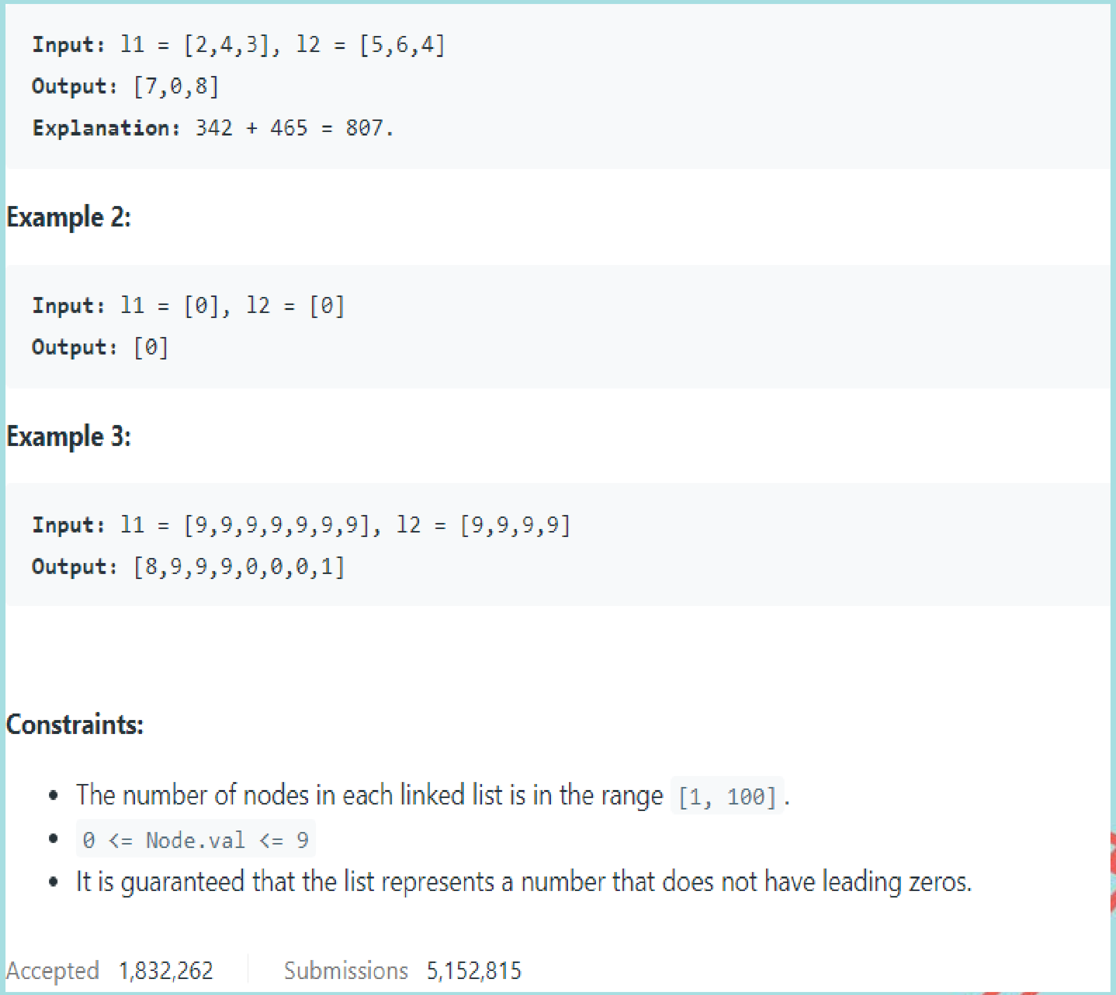
Medium 11104 2659 Add to List Share

You are given two **non-empty** linked lists representing two non-negative integers. The digits are stored in **reverse order**, and each of their nodes contains a single digit. Add the two numbers and return the sum as a linked list.

You may assume the two numbers do not contain any leading zero, except the number 0 itself.

Example 1:

$$\begin{tikzcd} 2 \arrow[r] & 4 \arrow[r] & 3 \\ 5 \arrow[r] & 6 \arrow[r] & 4 \\ 7 \arrow[r] & 0 \arrow[r] & 8 \end{tikzcd}$$



```
Input: l1 = [2,4,3], l2 = [5,6,4]
Output: [7,0,8]
Explanation: 342 + 465 = 807.
```

Example 2:

```
Input: l1 = [0], l2 = [0]
Output: [0]
```

Example 3:

```
Input: l1 = [9,9,9,9,9,9,9], l2 = [9,9,9,9]
Output: [8,9,9,9,0,0,0,1]
```

Constraints:

- The number of nodes in each linked list is in the range `[1, 100]`.
- `0 <= Node.val <= 9`
- It is guaranteed that the list represents a number that does not have leading zeros.

Accepted 1,832,262 | Submissions 5,152,815

https://leetcode.com/problems/add-two-numbers/

Leetcode problem: **Reverse Integer**

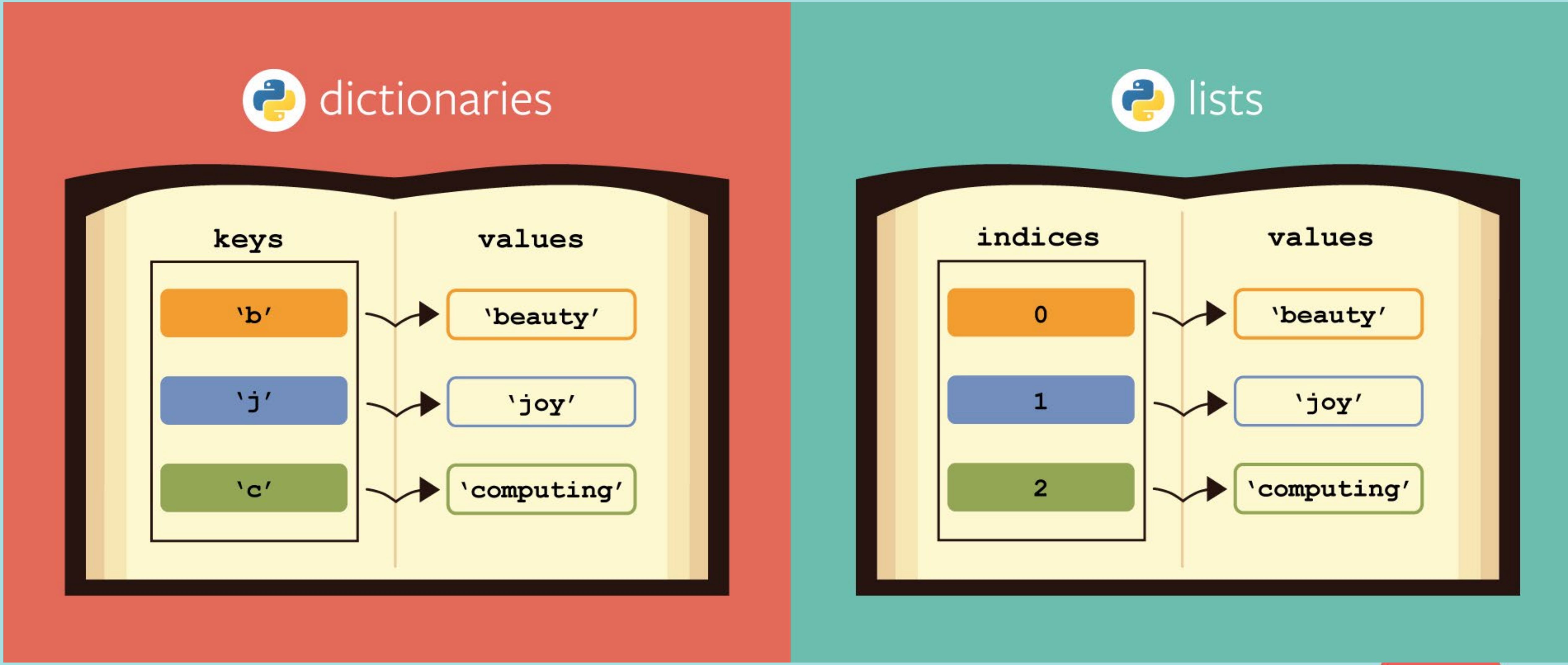
- Given a 32-bit signed integer, reverse digits of an integer.
- **Note:** Assume we are dealing with an environment that could only store integers within the 32-bit signed integer range: $[-2^{31}, 2^{31} - 1]$. For the purpose of this problem, assume that your function returns 0 when the reversed integer overflows.
- （LeetCode）
- https://leetcode.com/problems/reverse-integer/

3. MORE PYTHON PROGRAMMING

(A) Python Syntax

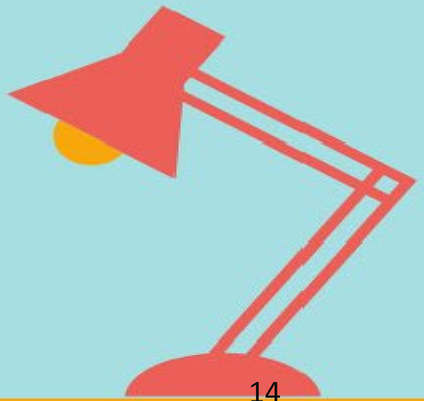
- Topic 1: The use of data structure Dict
- Topic 2: Loop: items(), enumerate(), zip(),
- Topic 3: Condition: Boolean and comparison operations

Topic 1: List and Dictionary



Topic 1: List and Dictionary

- List:
  - Lists are used to store multiple items in a single variable.
  - Lists are created using square brackets
  - nums = [1, 3, 5, 11, 16, 27, 28]
  - nums.append(39)
  - len(nums)
  - n = nums[2]
  - num2= nums[3:6]
- Dictionary:
  - Dictionaries are used to store data values in key:value pairs.
  - dmap = {1:2, 3:11, 5:7, 11:6, 16:5, 27:7, 28:8}
  - dmap[2]=10
  - To determine whether the string exists in dictionary A, you can use “in” 。
  - len(dmap)



Topic 2: items(), keys(), values()

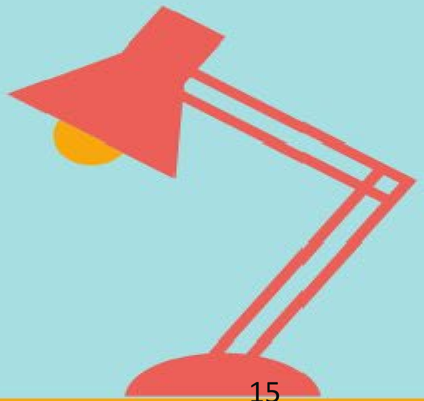
- dmap = {1:2, 3:11, 5:7, 11:6, 16:5, 27:7, 28:8}

- for key, values in dmap.items():
-       print (key, values)

- for key in dmap.keys():
-     print(key)

- for value in dmap.values():
-     print(value)

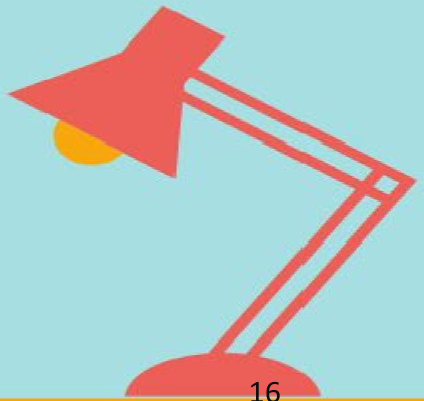
- for i, pair in enumerate(dmap.items()):
-     print(i, pair)



Topic 2: enumerate()

- nums = [1, 3, 5, 11, 16, 27, 28]
- for i, value in enumerate(nums):
-     print (i, value)

- dmap = {1:2, 3:11, 5:7, 11:6, 16:5, 27:7, 28:8}
- for i, pair in enumerate(d.items()):
-     print(i, pair)



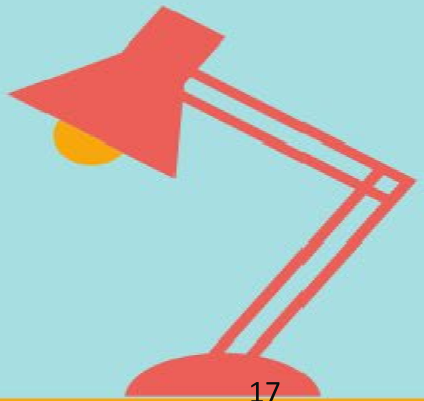
Topic 3: Concatenation of Boolean operators and comparison operations

```
a=5 ; b=6; c=10
if a < b < c:
  print("ascending")
elif a > b > c:
  print("descending")
```

比較運算子 Comparison operators
> >= < <= == !=

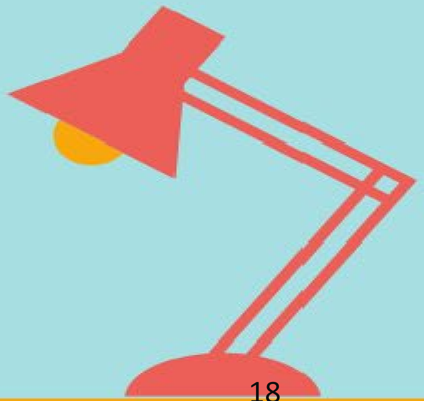
```
a, b, c=5, 6, 10
if a < b and b < c:
  print("ascending")
elif a > b and b > c:
  print("descending")
```

布林運算子Boolean operators
and / or



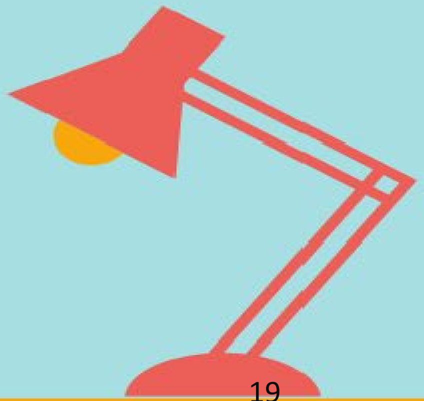
twoSum (1)

```python
class Solution:
    def twoSum(self, nums: List[int], target: int) -> List[int]:
        for i in range(len(nums)):
            for j in range(len(nums)):
                if i == j:
                    continue
                if nums[i] + nums[j] == target:
                    return [i, j]
        return []
```



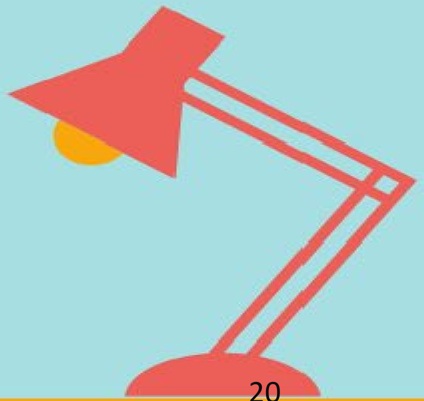
twoSum (2)

```
class Solution:
    def twoSum(self, nums: List[int], target: int) -> List[int]:
        for i in range(len(nums)-1):
            for j in range(i+1, len(nums)):
                if nums[i] + nums[j] == target:
                    return [i, j]
        return []
```



twoSum (3)

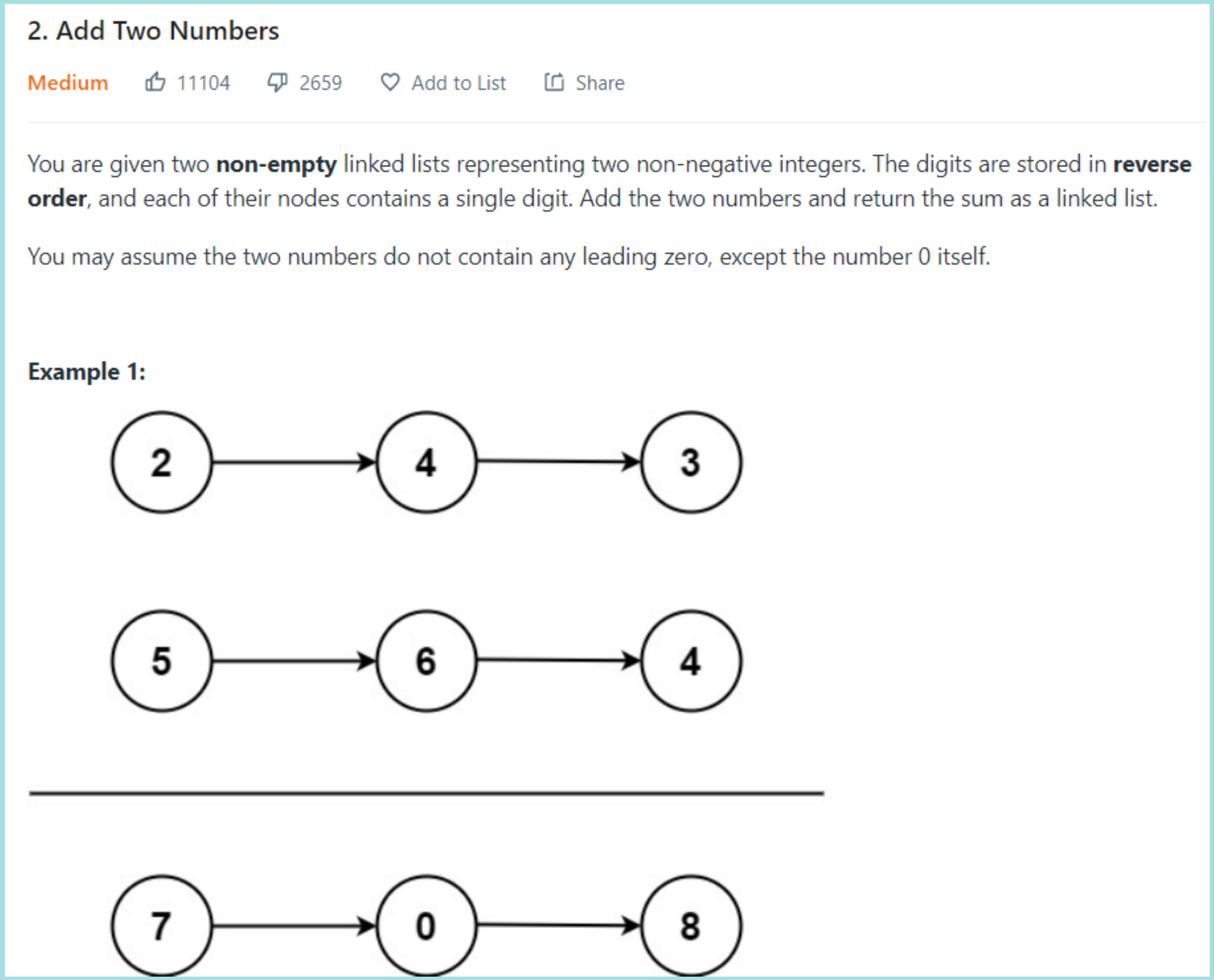
```
class Solution:
    def twoSum(self, nums: List[int], target: int) -> List[int]:
        for i, num1 in enumerate(nums):
            for j, num2 in enumerate(nums):
                if i == j:
                    continue
                if num1 + num2 == target:
                    return [i, j]
```



twoSum (4)

```
class Solution:
    def twoSum(self, nums: List[int], target: int) -> List[int]:
        complement ={}
        for i, num in enumerate(nums):    # a dictionary
            complement[num]=i
        for i, num in enumerate(nums):
            num2 = target-num
            if num2 in complement:
                j = complement[num2]
                if j != i:
                    return [i, j]
```

Leetcode problem: Add Two Numbers



2. Add Two Numbers

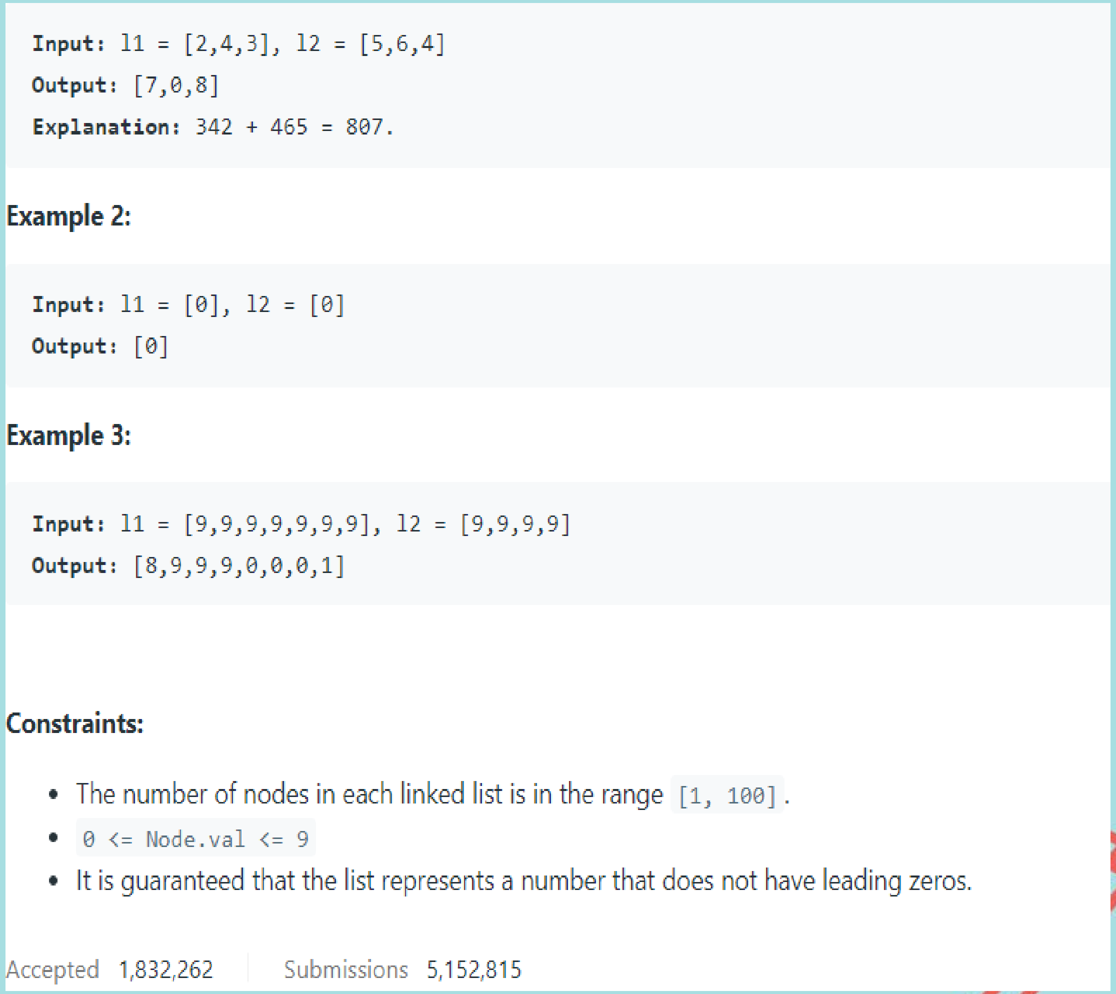
Medium 11104 2659 Add to List Share

You are given two **non-empty** linked lists representing two non-negative integers. The digits are stored in **reverse order**, and each of their nodes contains a single digit. Add the two numbers and return the sum as a linked list.

You may assume the two numbers do not contain any leading zero, except the number 0 itself.

Example 1:

$$\begin{tikzcd} 2 \arrow[r] & 4 \arrow[r] & 3 \\ 5 \arrow[r] & 6 \arrow[r] & 4 \\ 7 \arrow[r] & 0 \arrow[r] & 8 \end{tikzcd}$$



```
Input: l1 = [2,4,3], l2 = [5,6,4]
Output: [7,0,8]
Explanation: 342 + 465 = 807.
```

Example 2:

```
Input: l1 = [0], l2 = [0]
Output: [0]
```

Example 3:

```
Input: l1 = [9,9,9,9,9,9,9], l2 = [9,9,9,9]
Output: [8,9,9,9,0,0,0,1]
```

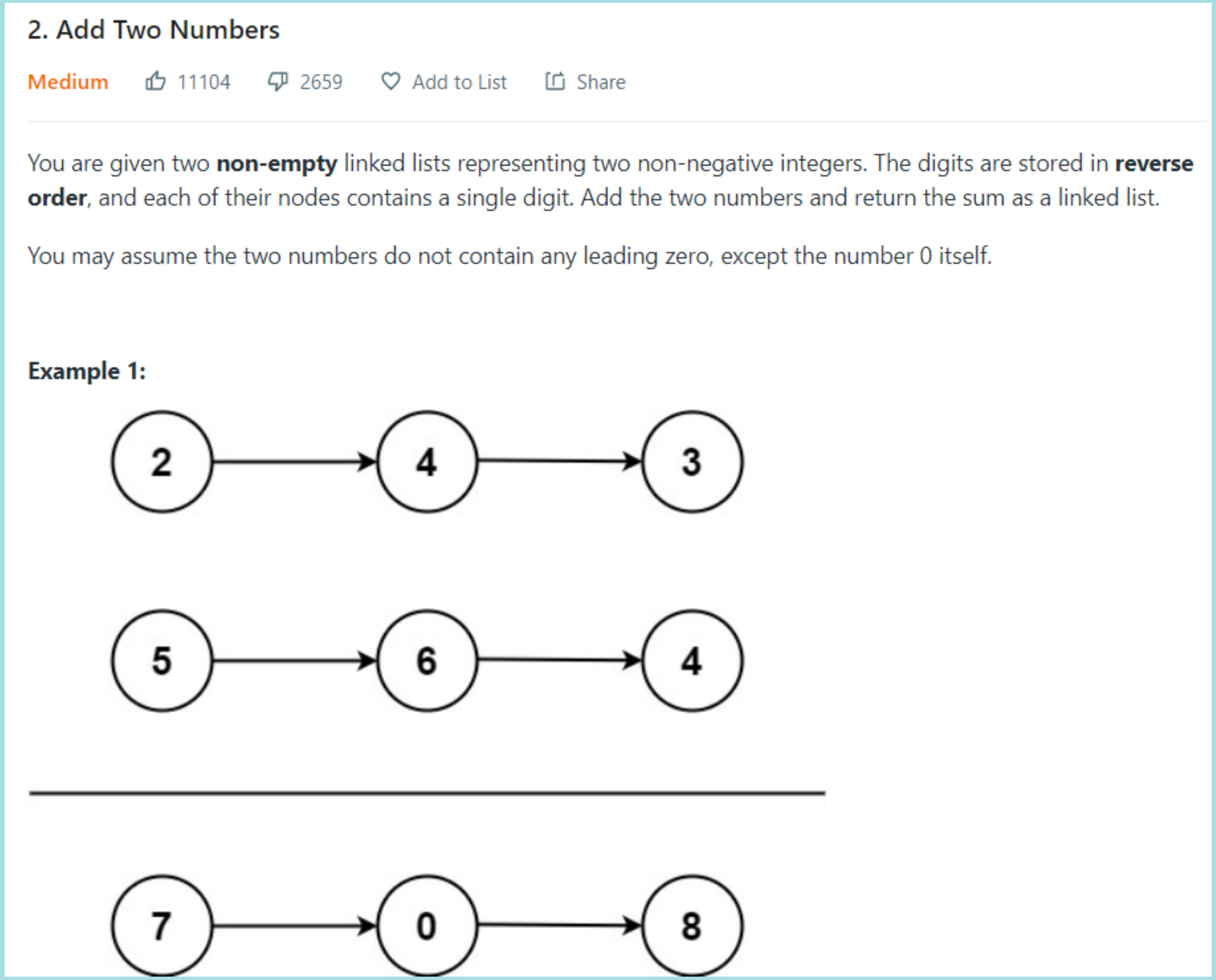
Constraints:

- The number of nodes in each linked list is in the range `[1, 100]`.
- `0 <= Node.val <= 9`
- It is guaranteed that the list represents a number that does not have leading zeros.

Accepted 1,832,262 Submissions 5,152,815

https://leetcode.com/problems/add-two-numbers/

Add Two Numbers



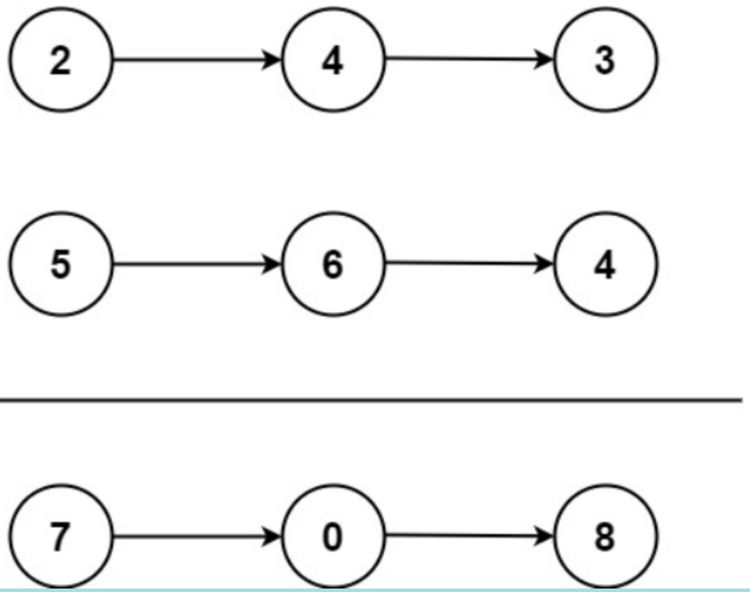
2. Add Two Numbers

Medium 11104 2659 Add to List Share

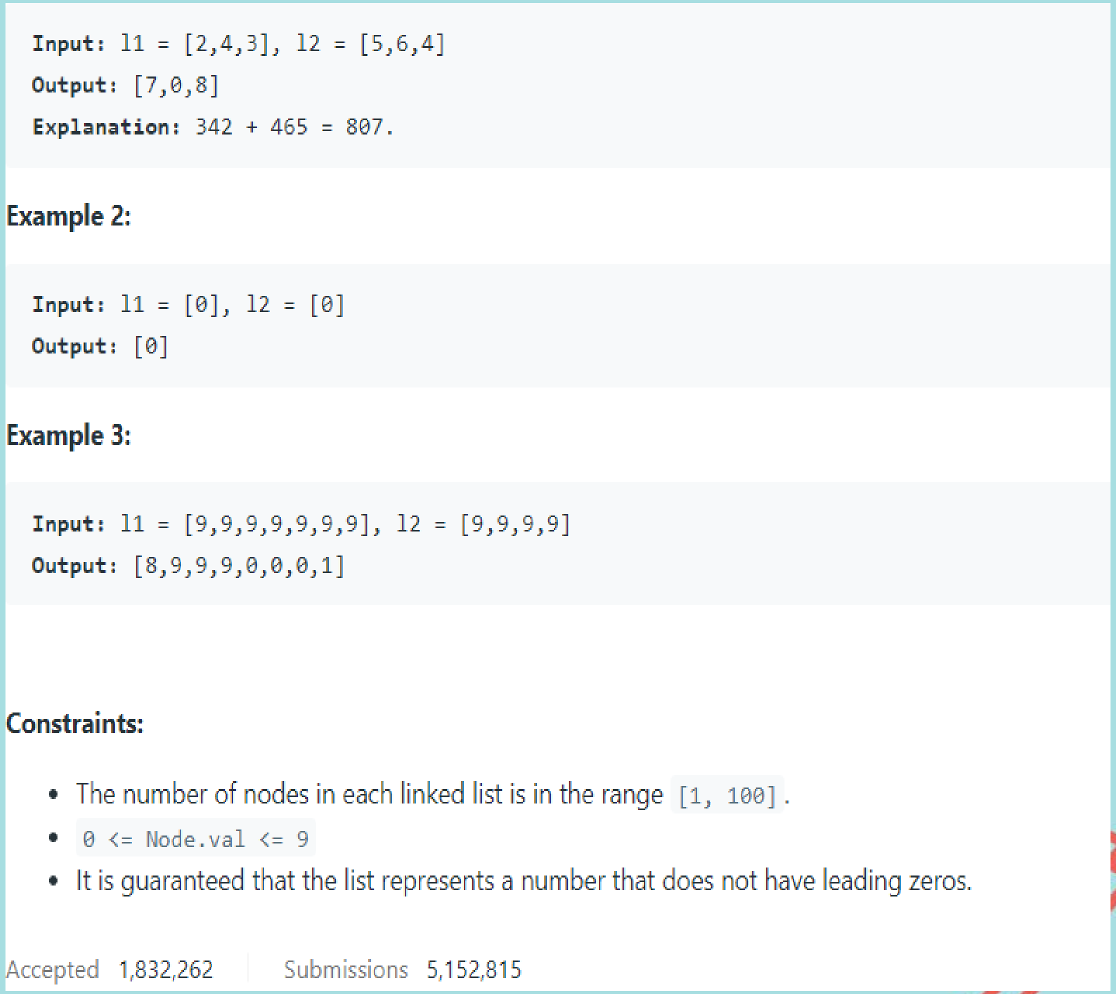
You are given two **non-empty** linked lists representing two non-negative integers. The digits are stored in **reverse order**, and each of their nodes contains a single digit. Add the two numbers and return the sum as a linked list.

You may assume the two numbers do not contain any leading zero, except the number 0 itself.

Example 1:



$$\begin{tikzcd}
2 \arrow[r] & 4 \arrow[r] & 3 \\
5 \arrow[r] & 6 \arrow[r] & 4 \\
7 \arrow[r] & 0 \arrow[r] & 8
\end{tikzcd}$$



```
Input: l1 = [2,4,3], l2 = [5,6,4]
Output: [7,0,8]
Explanation: 342 + 465 = 807.
```

Example 2:

```
Input: l1 = [0], l2 = [0]
Output: [0]
```

Example 3:

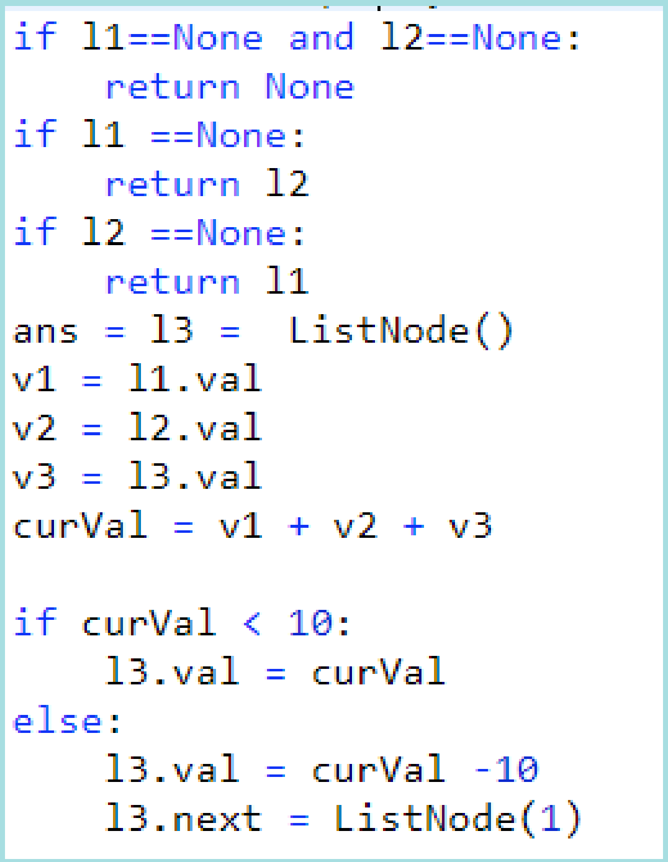
```
Input: l1 = [9,9,9,9,9,9,9], l2 = [9,9,9,9]
Output: [8,9,9,9,0,0,0,1]
```

Constraints:

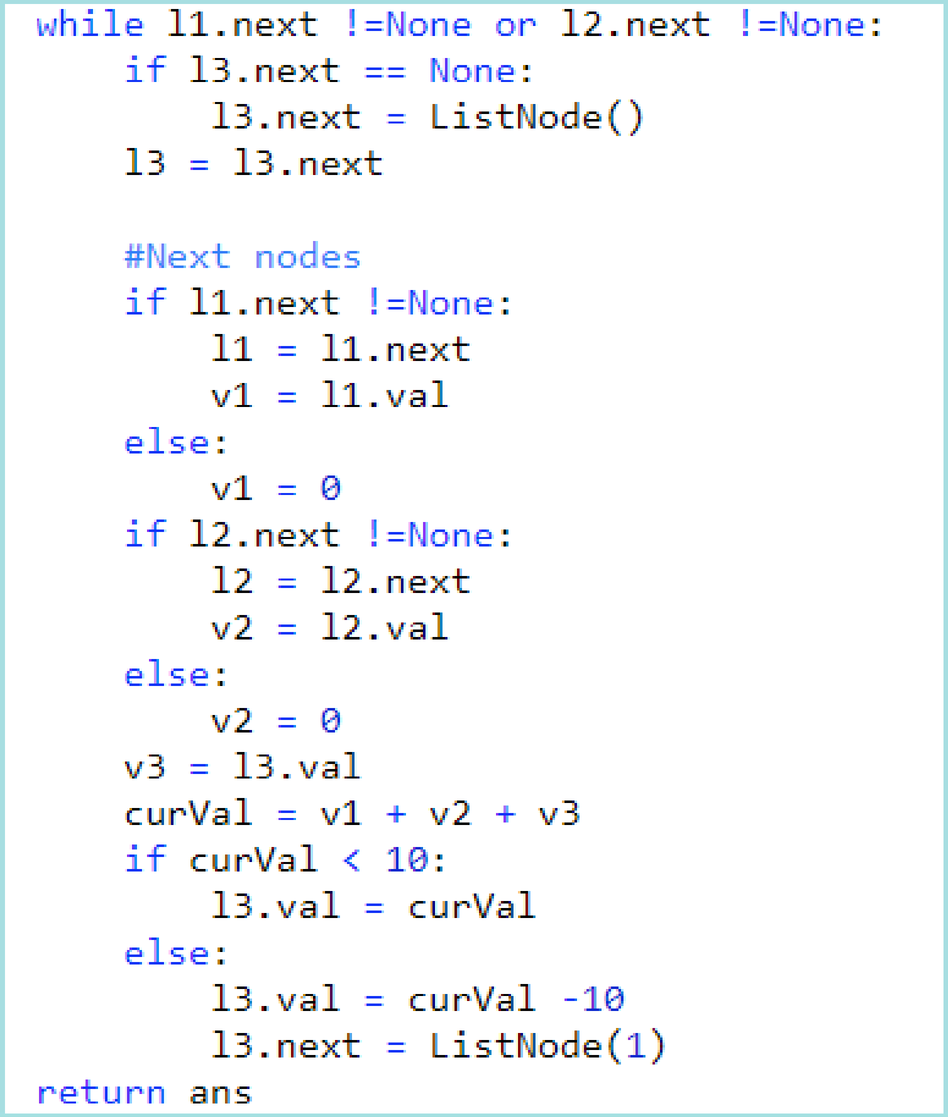
- The number of nodes in each linked list is in the range `[1, 100]`.
- `0 <= Node.val <= 9`
- It is guaranteed that the list represents a number that does not have leading zeros.

Accepted 1,832,262 | Submissions 5,152,815

add-two-numbers-Version 1

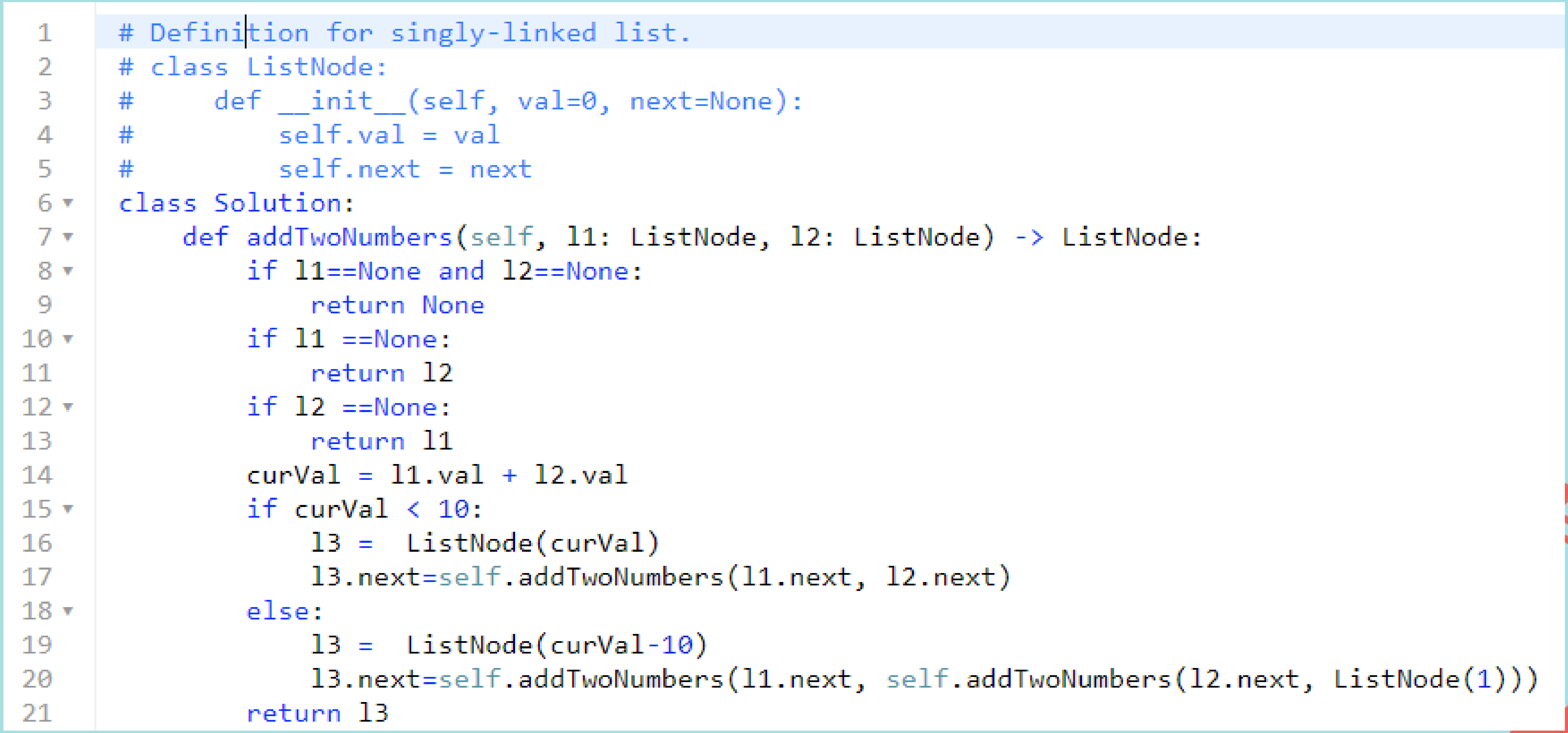


```
if l1==None and l2==None:
    return None
if l1 ==None:
    return l2
if l2 ==None:
    return l1
ans = l3 =  ListNode()
v1 = l1.val
v2 = l2.val
v3 = l3.val
curVal = v1 + v2 + v3

if curVal < 10:
    l3.val = curVal
else:
    l3.val = curVal -10
    l3.next = ListNode(1)
```



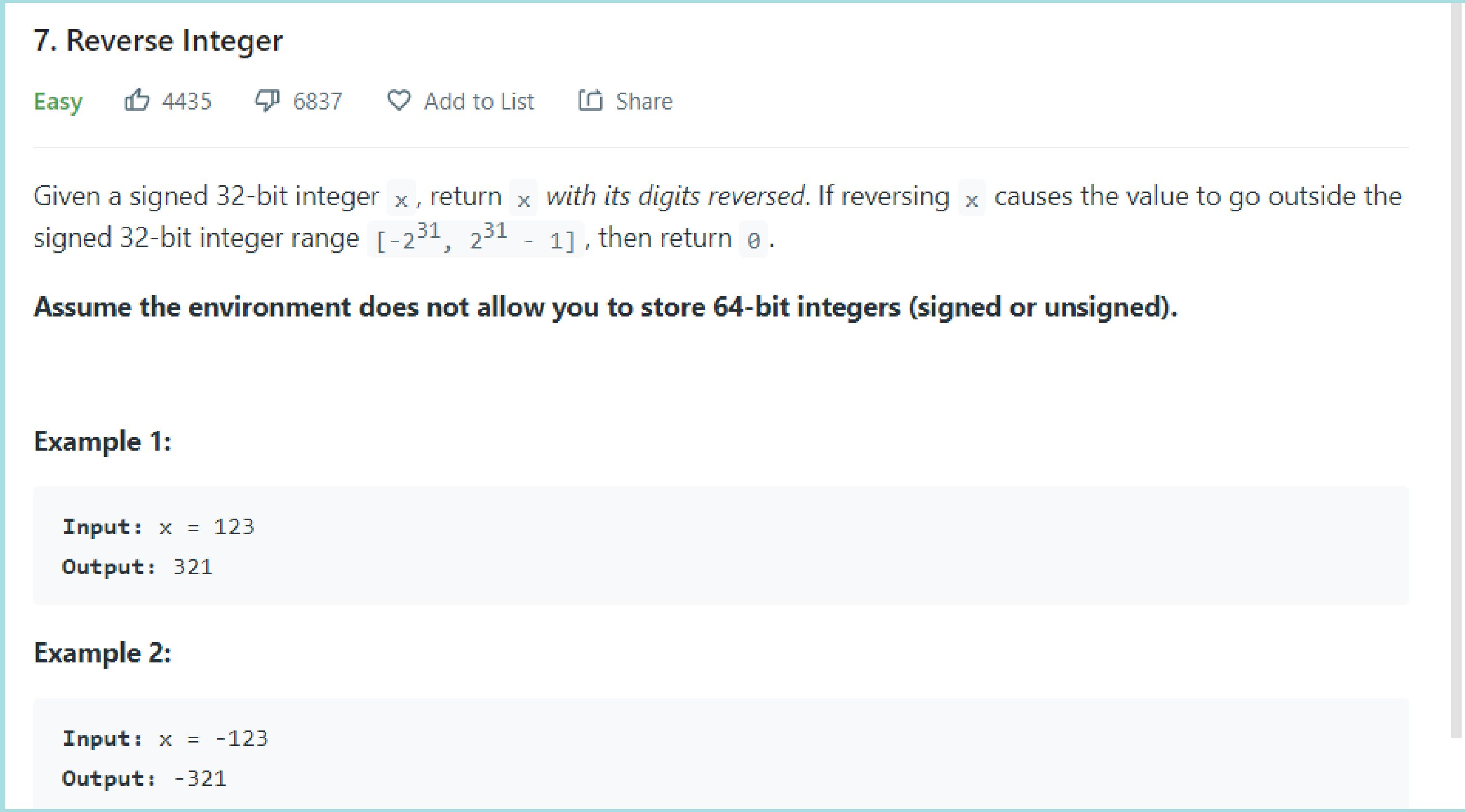
```
while l1.next !=None or l2.next !=None:
    if l3.next == None:
        l3.next = ListNode()
    l3 = l3.next

    #Next nodes
    if l1.next !=None:
        l1 = l1.next
        v1 = l1.val
    else:
        v1 = 0
    if l2.next !=None:
        l2 = l2.next
        v2 = l2.val
    else:
        v2 = 0
    v3 = l3.val
    curVal = v1 + v2 + v3
    if curVal < 10:
        l3.val = curVal
    else:
        l3.val = curVal -10
        l3.next = ListNode(1)
return ans
```

add-two-numbers-Version 2



```python
# Definition for singly-linked list.
# class ListNode:
#     def __init__(self, val=0, next=None):
#         self.val = val
#         self.next = next
class Solution:
    def addTwoNumbers(self, l1: ListNode, l2: ListNode) -> ListNode:
        if l1==None and l2==None:
            return None
        if l1 ==None:
            return l2
        if l2 ==None:
            return l1
        curVal = l1.val + l2.val
        if curVal < 10:
            l3 =  ListNode(curVal)
            l3.next=self.addTwoNumbers(l1.next, l2.next)
        else:
            l3 =  ListNode(curVal-10)
            l3.next=self.addTwoNumbers(l1.next, self.addTwoNumbers(l2.next, ListNode(1)))
        return l3
```

Reverse Integer



7. Reverse Integer

Easy 4435 6837 Add to List Share

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 `[-2^31, 2^31 - 1]`, then return `0`.

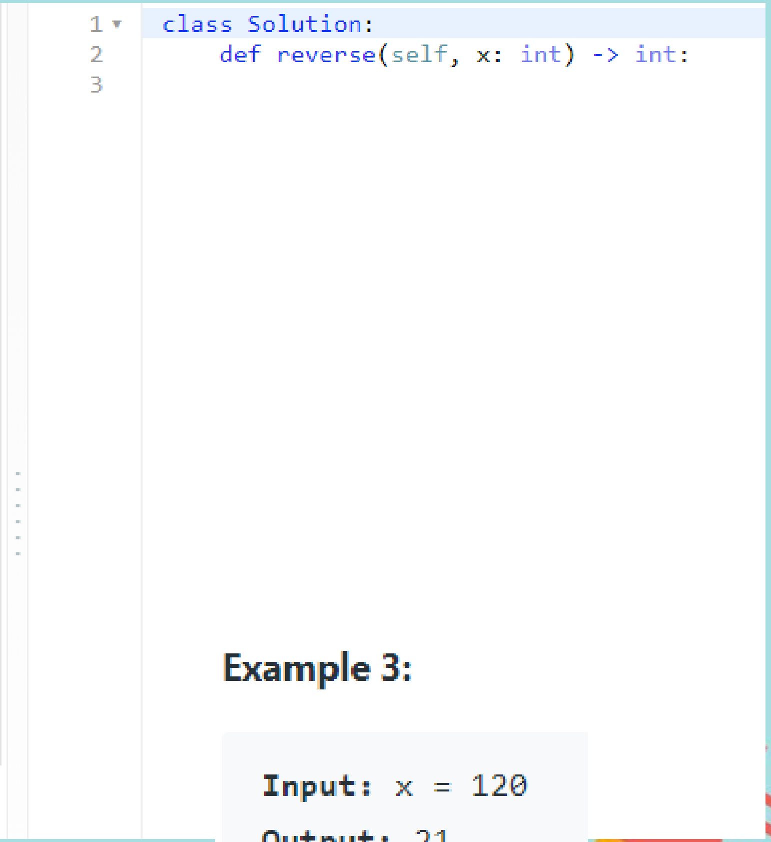
Assume the environment does not allow you to store 64-bit integers (signed or unsigned).

Example 1:

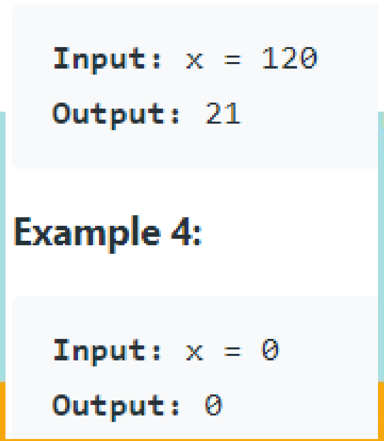
```
Input: x = 123
Output: 321
```

Example 2:

```
Input: x = -123
Output: -321
```



```python
class Solution:
    def reverse(self, x: int) -> int:
```



Example 3:

```
Input: x = 120
Output: 21
```

Example 4:

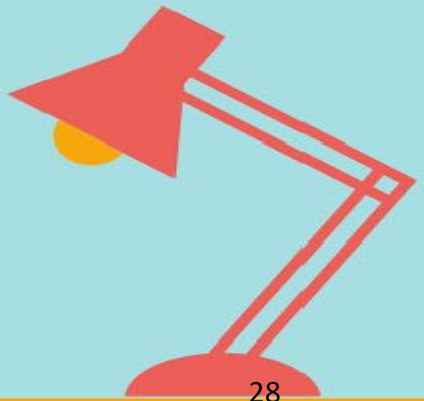
```
Input: x = 0
Output: 0
```

Reverse Integer-Version 1

```
class Solution:
    def addTwoNumbers(self, l1: Optional[ListNode], l2: Optional[ListNode]) -> Optional[ListNode]:
        dummyHead = ListNode(0)
        curr = dummyHead
        carry = 0
        while l1 != None or l2 != None or carry != 0:
            l1Val = l1.val if l1 else 0
            l2Val = l2.val if l2 else 0
            columnSum = l1Val + l2Val + carry
            carry = columnSum // 10
            newNode = ListNode(columnSum % 10)
            curr.next = newNode
            curr = newNode
            l1 = l1.next if l1 else None
            l2 = l2.next if l2 else None
        return dummyHead.next
```

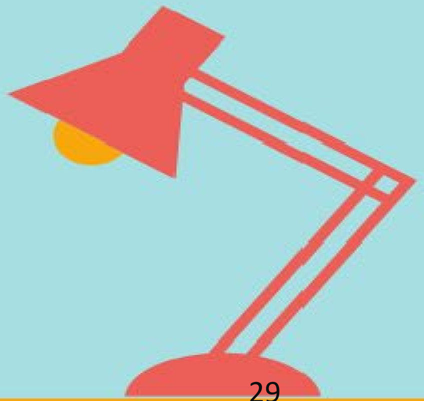
Activity-1 (S-S)

- Open the Google Jamboard for the class
- Check the grammar you have used in Python Cheatsheet.



Activity-2 (S-S)

- Open the Google Jamboard for the class
- Discuss what you know (K), want to know (W), and have learned (L) about this topic in the KWL chart.



Thanks!

Q&A