2. Add Two Numbers

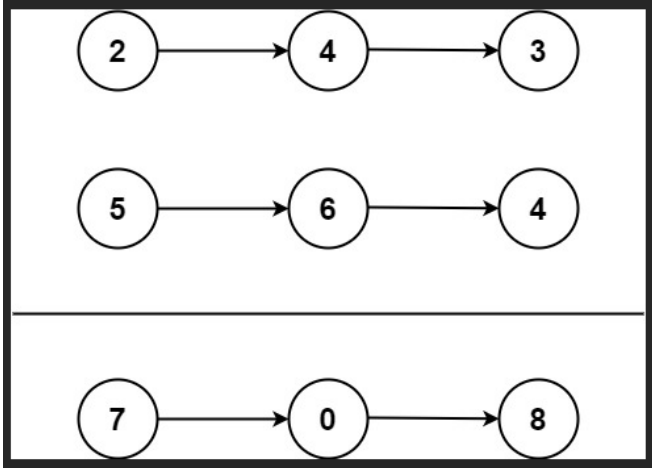
Solved

Medium | Topics | Companies

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:




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

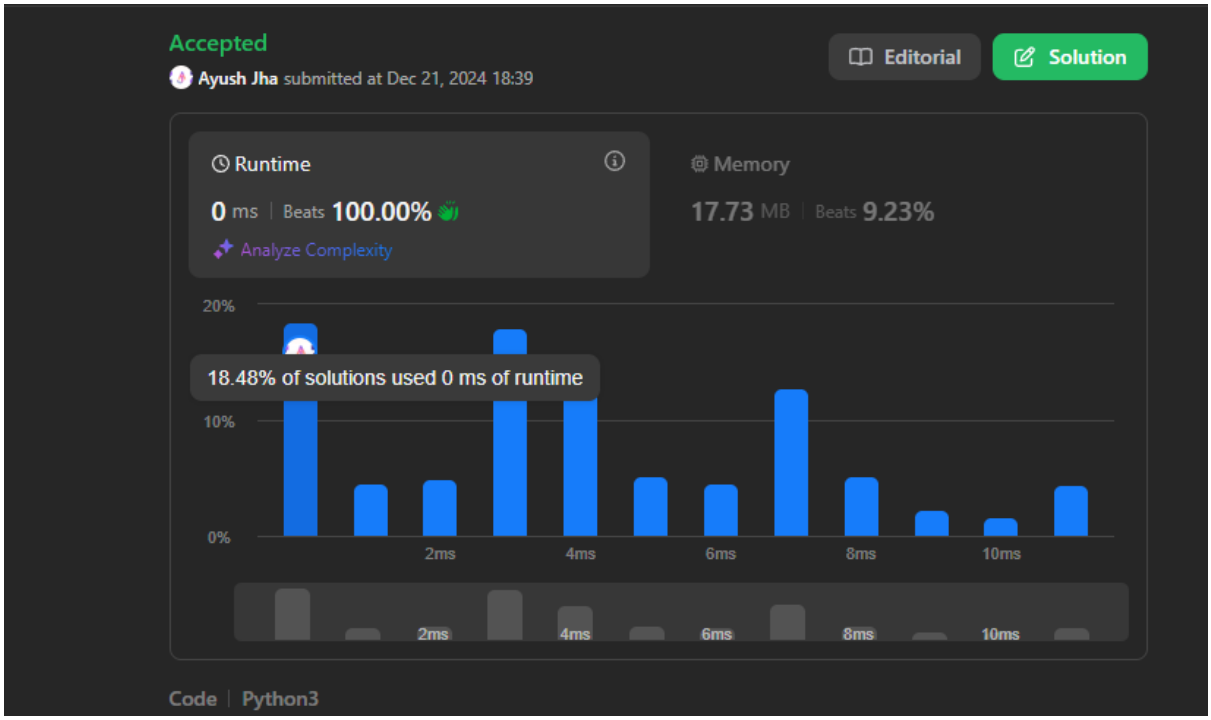
Solution:-

Python3 | Auto

```python
class Solution:
    def twoSum(self, nums: List[int], target: int) -> List[int]:
        numMap = {}
        n = len(nums)

        for i in range(n):
            complement = target - nums[i]
            if complement in numMap:
                return [numMap[complement], i]
            numMap[nums[i]] = i

        return []
```



6. Zigzag Conversion

Solved

Medium | Topics | Companies

The string `"PAYPALISHIRING"` is written in a zigzag pattern on a given number of rows like this: (you may want to display this pattern in a fixed font for better legibility)

```
P   A   H   N
A P L S I I G
Y   I   R
```

And then read line by line: `"PAHNAPLSIIGYIR"`

Write the code that will take a string and make this conversion given a number of rows:

```
string convert(string s, int numRows);
```

Example 1:

```
Input: s = "PAYPALISHIRING", numRows = 3
Output: "PAHNAPLSIIGYIR"
```

Solution:-

Python3 | Auto

```python
class Solution:
    def convert(self, s: str, numRows: int) -> str:
        if numRows == 1:
            return s

        row_arr = [""] * numRows
        row_idx = 1
        going_up = True

        for ch in s:
            row_arr[row_idx-1] += ch
            if row_idx == numRows:
                going_up = False
            elif row_idx == 1:
                going_up = True

            if going_up:
                row_idx += 1
            else:
                row_idx -= 1

        return "".join(row_arr)
```

Accepted

Ayush Jha submitted at Dec 21, 2024 18:42

Editorial | Solution

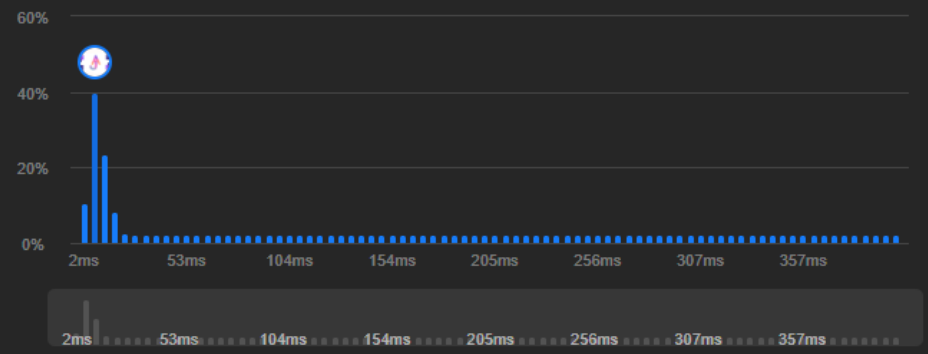
Runtime

7 ms | Beats 89.30%

Analyze Complexity

Memory

17.70 MB | Beats 15.56%

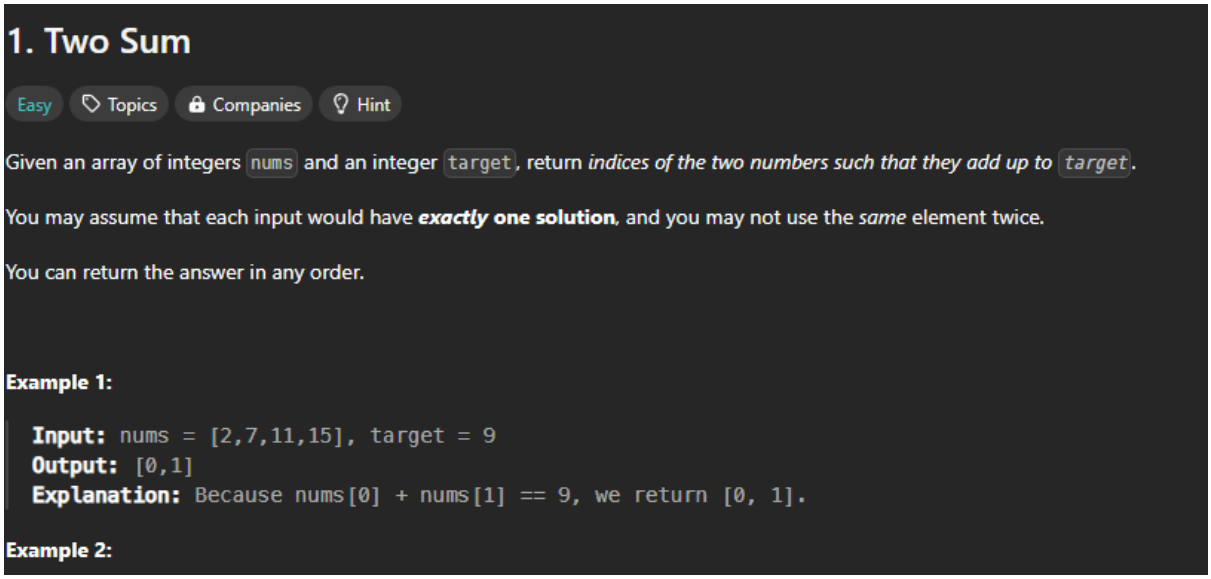


Code | Python3

```python
class Solution:
    def convert(self, s: str, numRows: int) -> str:
        if numRows == 1:
            return s

        row_arr = [""] * numRows
        row_idx = 1
        going_up = True
```

View more



1. Two Sum

Easy | Topics | Companies | Hint

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].
```

Example 2:

Solution:-

Code

Python3 | Auto

```python
class Solution:
    def twoSum(self, nums: List[int], target: int) -> List[int]:
        numMap = {}
        n = len(nums)

        for i in range(n):
            complement = target - nums[i]
            if complement in numMap:
                return [numMap[complement], i]
            numMap[nums[i]] = i

        return []
```

Accepted

Ayush Jha submitted at Dec 21, 2024 18:45

Editorial | Solution

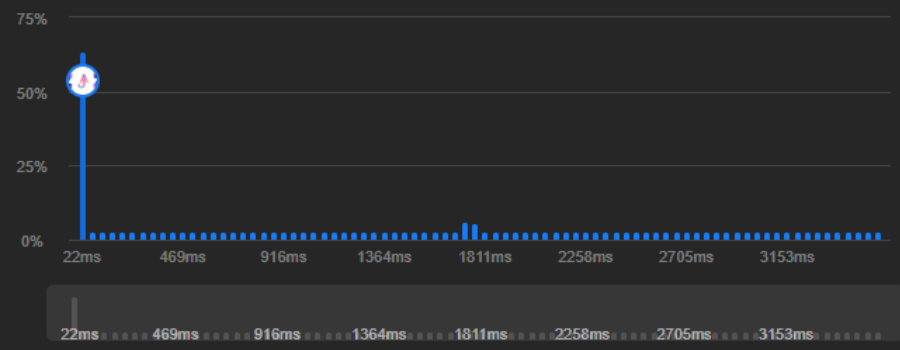
Runtime

0 ms | Beats 100.00%

Analyze Complexity

Memory

18.96 MB | Beats 5.35%



Code | Python3

```python
class Solution:
    def twoSum(self, nums: List[int], target: int) -> List[int]:
        numMap = {}
        n = len(nums)

        for i in range(n):
            complement = target - nums[i]
            if complement in numMap:
```

View more

12. Integer to Roman

Solved

Medium | Topics | Companies

Seven different symbols represent Roman numerals with the following values:

Symbol	Value
I	1
V	5
X	10
L	50
C	100
D	500
M	1000

Roman numerals are formed by appending the conversions of decimal place values from highest to lowest. Converting a decimal place value into a Roman numeral has the following rules:

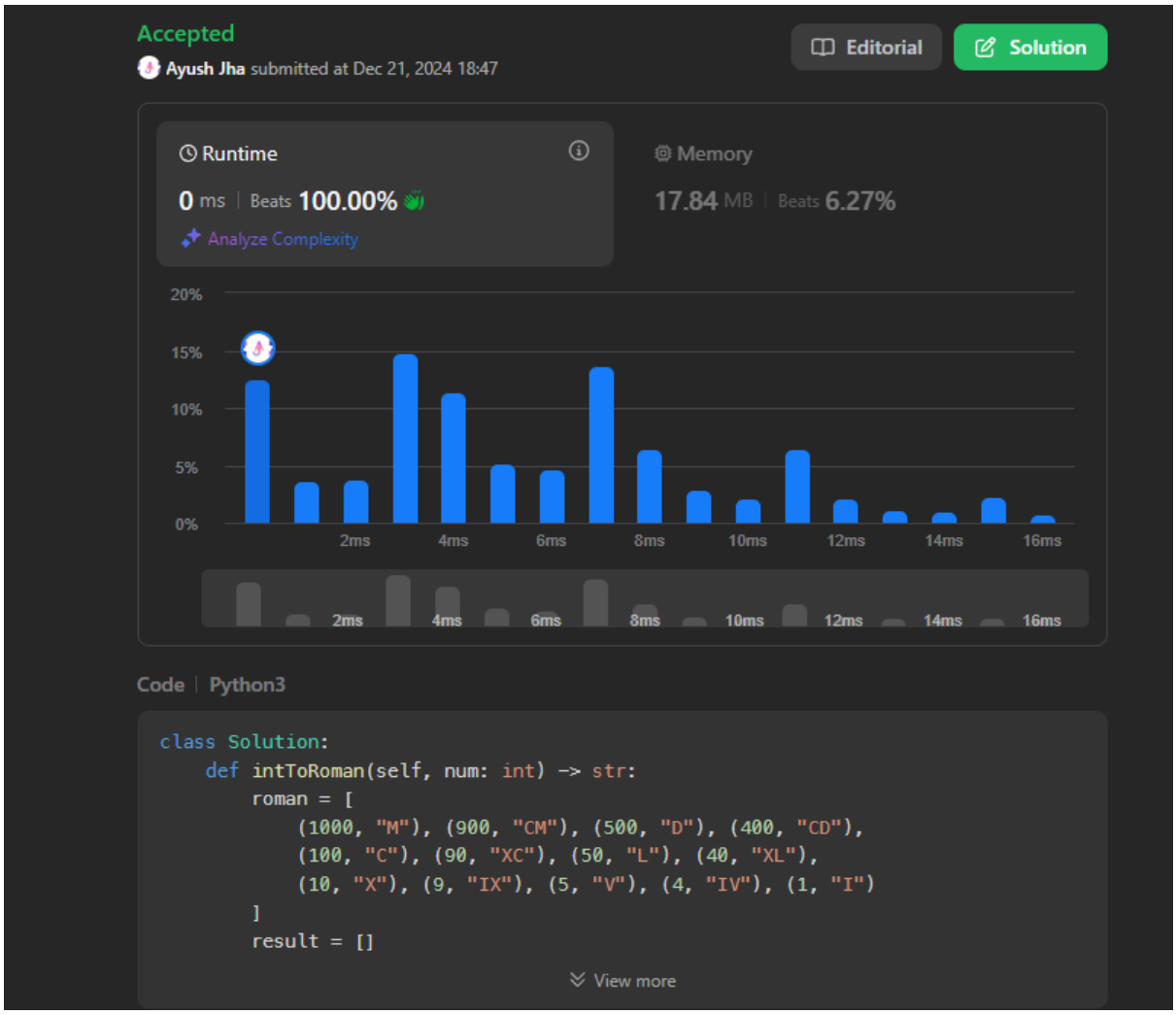
- If the value does not start with 4 or 9, select the symbol of the maximal value that can be subtracted from the input, append that symbol to the result, subtract its value, and convert the remainder to a Roman numeral.
- If the value starts with 4 or 9 use the **subtractive form** representing one symbol subtracted from the following symbol, for example, 4 is 1 (`I`) less than 5 (`V`): `IV` and 9 is 1 (`I`) less than 10 (`X`): `IX`. Only the following subtractive forms are used: 4 (`IV`), 9 (`IX`), 40 (`XL`), 90 (`XC`), 400 (`CD`) and 900 (`CM`).
- Only powers of 10 (`I`, `X`, `C`, `M`) can be appended consecutively at most 3 times to represent multiples of 10. You cannot append 5 (`V`), 50 (`L`), or 500 (`D`) multiple times. If you need to append a symbol 4 times use the **subtractive form**.

Given an integer, convert it to a Roman numeral.

Solution:-

Python3 Auto

```python
class Solution:
    def intToRoman(self, num: int) -> str:
        roman = [
            (1000, "M"), (900, "CM"), (500, "D"), (400, "CD"),
            (100, "C"), (90, "XC"), (50, "L"), (40, "XL"),
            (10, "X"), (9, "IX"), (5, "V"), (4, "IV"), (1, "I")
        ]
        result = []
        for value, symbol in roman:
            while num >= value:
                result.append(symbol)
                num -= value
        return "".join(result)
```



43. Multiply Strings

Medium | Topics | Companies

Given two non-negative integers `num1` and `num2` represented as strings, return the product of `num1` and `num2`, also represented as a string.

Note: You must not use any built-in BigInteger library or convert the inputs to integer directly.

Example 1:

> **Input:** num1 = "2", num2 = "3"
> **Output:** "6"

Example 2:

> **Input:** num1 = "123", num2 = "456"
> **Output:** "56088"

Solution:-

Code

Python3 Auto

```python
class Solution:
    def multiply(self, num1: str, num2: str) -> str:

        n1,n2 = 0,0
        for i in num1:
            n1 = n1*10 + (ord(i)-48)
        for i in num2:
            n2 = n2*10 + (ord(i)-48)
        return f"{n1*n2}"
```

Accepted

Ayush Jha submitted at Dec 21, 2024 18:50

Editorial | Solution

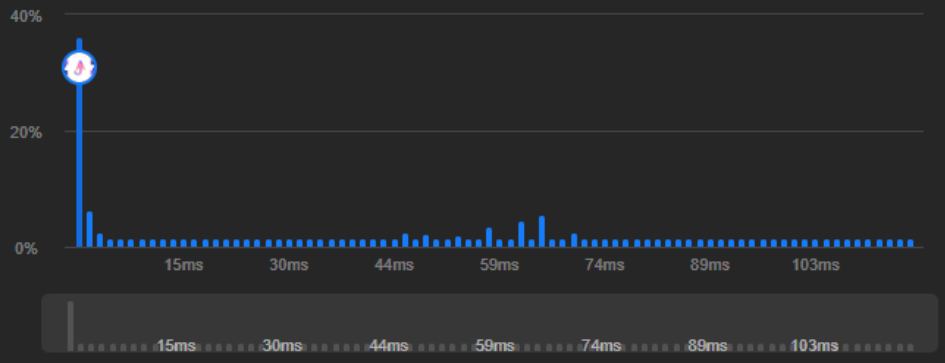
Runtime

0 ms | Beats 100.00%

Analyze Complexity

Memory

17.74 MB | Beats 8.79%



Code | Python3

```python
class Solution:
    def multiply(self, num1: str, num2: str) -> str:

        n1,n2 = 0,0
        for i in num1:
            n1 = n1*10 + (ord(i)-48)
        for i in num2:
            n2 = n2*10 + (ord(i)-48)
        return f"{n1*n2}"
```