13. Roman to Integer

Solved

Easy | Topics | Companies | Hint

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 as `IV`. Because the one is before the five we subtract it making four. The same principle applies to the number nine, which is written as `IX`. There are six instances where subtraction is used:

- `I` can be placed before `V` (5) and `X` (10) to make 4 and 9.
- `X` can be placed before `L` (50) and `C` (100) to make 40 and 90.

- `C` can be placed before `D` (500) and `M` (1000) to make 400 and 900.

Given a roman numeral, convert it to an integer.

Example 1:

```
Input: s = "III"
Output: 3
Explanation: III = 3.
```

Example 2:

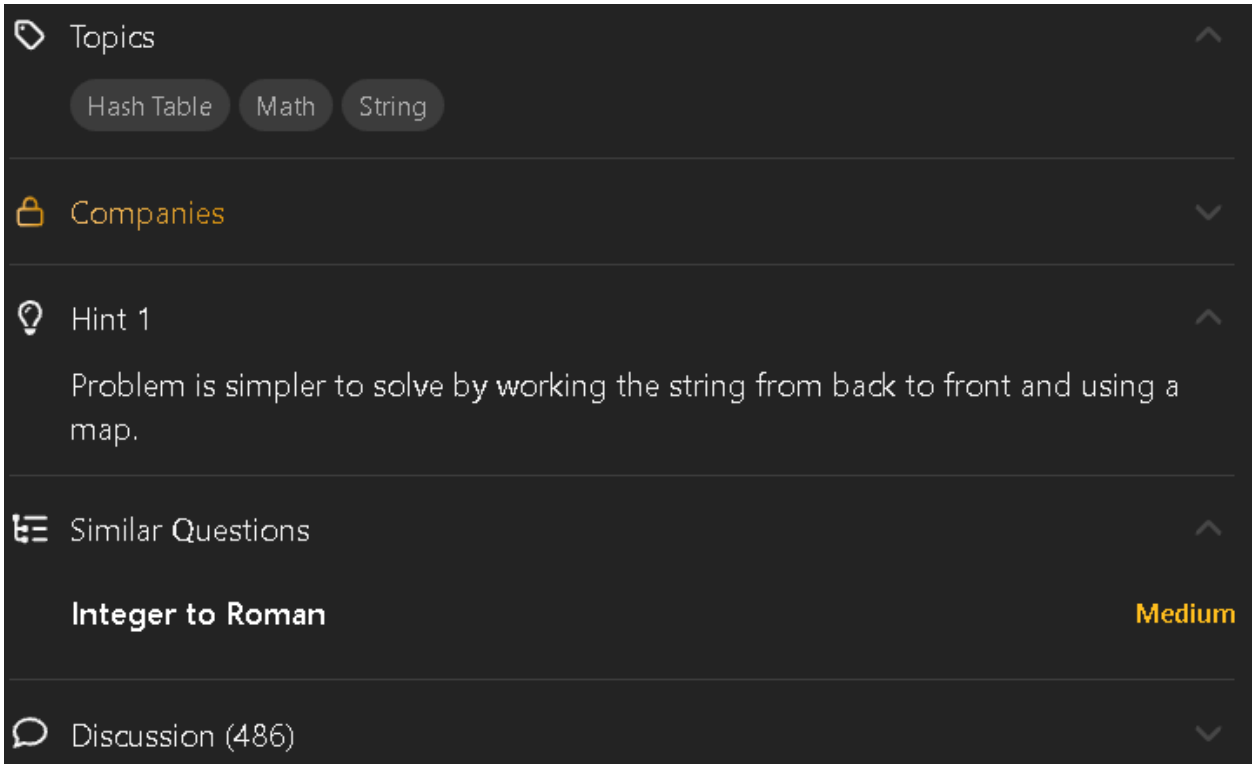
```
Input: s = "LVIII"
Output: 58
Explanation: L = 50, V= 5, III = 3.
```

Example 3:

```
Input: s = "MCMXCIV"
Output: 1994
Explanation: M = 1000, CM = 900, XC = 90 and IV = 4.
```

Constraints:

- `1 <= s.length <= 15`
- `s` contains only the characters `('I', 'V', 'X', 'L', 'C', 'D', 'M')`.
- It is **guaranteed** that `s` is a valid roman numeral in the range `[1, 3999]`.



Python:

```
class Solution:
    def romanToInt(self, s: str) -> int:
        symbol = {"I": 1, "V": 5, "X": 10, "L": 50, "C": 100, "D": 500, "M": 1000}
        inpu = s
        value, index = 0, 0
        condition = True
        while condition:
            var1 = symbol[inpu[index]]
            var2 = symbol[inpu[index + 1]] if index < len(inpu) - 1 else 0
            if var1 >= var2:
                value += var1
                index += 1
            if var1 < var2:
                value += var2 - var1
                index += 2
            if index >= len(inpu):
                condition = False
        return value
```

Java:

```
class Solution {
    public int romanToInt(String s) {
        // Step 1: Create a HashMap for Roman symbols and their values
```

```
        HashMap<Character, Integer> romanMap = new HashMap<>();
        romanMap.put('I', 1);
        romanMap.put('V', 5);
        romanMap.put('X', 10);
        romanMap.put('L', 50);
        romanMap.put('C', 100);
        romanMap.put('D', 500);
        romanMap.put('M', 1000);

        int total = 0;
        int prevValue = 0;

        // Step 2: Traverse the string from right to left
        for (int i = s.length() - 1; i >= 0; i--) {
            char currentChar = s.charAt(i);
            int currentValue = romanMap.get(currentChar);

            // Step 3: If current value is less than the previous value, subtract it
            if (currentValue < prevValue) {
                total -= currentValue;
            } else {
                total += currentValue;
            }

            // Step 4: Update previous value
            prevValue = currentValue;
        }

        return total;
    }
}
```

Javascript:

```
/**
 * @param {string} s
 * @return {number}
 */
var romanToInt = function(s) {
    // Roman numeral values
    const romanMap = {
        'I': 1,
        'V': 5,
        'X': 10,
        'L': 50,
```

```
        'C': 100,
        'D': 500,
        'M': 1000
    };

    let total = 0;

    // Loop through the string
    for (let i = 0; i < s.length; i++) {
        const currentVal = romanMap[s[i]];
        const nextVal = romanMap[s[i + 1]];

        // If the next value is greater, subtract current from total
        if (nextVal > currentVal) {
            total += (nextVal - currentVal);
            i++; // Skip the next character as it's already used
        } else {
            total += currentVal;
        }
    }

    return total;
};
```