0013 - Roman to Integer

大意說明

題幹給了羅馬數字的轉換規則:

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

還有這一段

- `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.

思考過程

- 讓程式循序讀字串裡面的字元，然後根據當下字元，以及前一個字元代表的數字去做處理
  - 如果前一個字元沒有遇到 I、X、C，那就照常加入沒差
  - 如果前一個字元有遇到上述字元，因為前面的字元已經加入到 total 當中，那只需要把剩下沒加入的數字加入進去，就代表上面的特殊轉換規則了

所以自己的程式碼就長成這樣:

```
class Solution(object):
    def romanToInt(self, s):
        """
        :type s: str
        :rtype: int
        """
        total = 0
```

```
formar_char = ''

for char in s:
    if char == 'I':
        total += 1
    elif char == 'V':
        if formar_char == 'I':
            total += 3 # 不是    4 喔, 因為前面的            I 已經加過了          XD, 底下其他也同理
        else:
            total += 5
    elif char == 'X':
        if formar_char == 'I':
            total += 8
        else:
            total += 10
    elif char == 'L':
        if formar_char == 'X':
            total += 30
        else:
            total += 50
    elif char == 'C':
        if formar_char == 'X':
            total += 80
        else:
            total += 100
    elif char == 'D':
        if formar_char == 'C':
            total += 300
        else:
            total += 500
    elif char == 'M':
        if formar_char == 'C':
            total += 800
        else:
            total += 1000
    formar_char = char
return total
```

更漂亮的解法

一堆 if-else 搞得我好像在寫 yandev...XD

建立一組無序 Map，然後用 Map 下去跑會比較快，然後可以不用 iterator 的方式寫，用傳統 for 寫法可以少存一個變數:

```
class Solution(object):
    def romanToInt(self, s):
        """
        :type s: str
        :rtype: int
        """
        total = 0

        roman_map = {
            'I': 1,
            'V': 5,
            'X': 10,
            'L': 50,
            'C': 100,
            'D': 500,
            'M': 1000
        }

        for i in range(0, len(s)):
            if i != 0 and roman_map[s[i - 1]] < roman_map[s[i]]:
                # 要減兩次的原因是，根據羅馬數字規則，若與後面的大數合併來看的話，前面已經加入的結果要扣除
                total = total + roman_map[s[i]] - (roman_map[s[i - 1]] * 2)
            else:
                total += roman_map[s[i]]

        return total
```

我還看到更白爛的，用 str.replace() 把特殊羅馬數字規則直接替換 ...我喜歡（但我應該不會先想到這個）

以下則是 Java 版本的程式碼:

```
class Solution {
    public int romanToInt(String s) {
        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;
        for(int count = 0; count < s.length(); count++){
            if(count != 0 && romanMap.get(s.charAt(count)) > romanMap.get(s.charAt(count - 1))){
                total -= romanMap.get(s.charAt(count - 1)) * 2;
            }

            total += romanMap.get(s.charAt(count));
        }

        return total;
    }
}
```

Revision #3
Created 2 July 2024 05:42:48 by Nesquate
Updated 8 July 2024 03:17:01 by Nesquate