[CS 11] Prac 9h – Roman Numerals

Problem Statement

Ramon numerals are a simple numeral system where each letter has a corresponding numerical value, and a number is written as a string of letters whose total value is equal to that number.

Roman is having none of that. He thinks it's too simplistic. So, he invented his own numeral system, which we can call **Roman numerals.** Roman numerals are similar to Ramon Numerals, but more complicated—if a letter appears somewhere to the left of another letter with a strictly larger value, its value is *subtracted* instead of added. For example, if I has value 1 and V has value 5, then IV is 4, while VI is 6.

Note that the letter doesn't have to be to the *immediate* left of the letter with the higher value—for example, if D has value 500, then VID has value 494 (and **not** 504).

We can also look at **generalized Roman numerals**, where someone selects letters along with their numeric values. For example, if Y = 10, A = 3, and P = 5, then "PAPAYA" is -3.

Given a Roman numeral, what is its value?

Submit solution [CS 11] Practice 9

Points: 145 (partial)
Time limit: 4.0s
Memory limit: 1G

Author: kvatienza (Kevin Atienza)
Problem type
Allowed languages ~~NONE~~, py3

Task Details

Your task is to implement a function called `roman_numeral_value`. This function has two parameters:

- the first parameter is a `dict` representing the value of each letter in a generalized Roman numeral system.
- the second parameter is a `str` representing a Roman numeral.

The function must return an `int` denoting the value of the Roman numeral of the given matrix.

Restrictions

(See 9a for more restrictions)

For this problem:

- Recursion is allowed.
- The source code limit is 2000.

Example Calls

Example 1 Function Call

```
roman_numeral_value({
    'I': 1,
    'V': 5,
    'X': 10,
    'L': 50,
    'C': 100,
    'D': 500,
    'M': 1000,
}, 'VIII')
```

Example 1 Return Value

```
8
```

Example 2 Function Call

```
roman_numeral_value({
    'I': 1,
    'V': 5,
    'X': 10,
    'L': 50,
    'C': 100,
    'D': 500,
    'M': 1000,
}, 'IV')
```

Example 2 Return Value

```
4
```

Example 3 Function Call

```
roman_numeral_value({
    'I': 1,
    'V': 5,
    'X': 10,
    'L': 50,
    'C': 100,
    'D': 500,
    'M': 1000,
}, 'VID')
```

Example 3 Return Value

```
494
```

Example 4 Function Call

```
roman_numeral_value({
    'Y': 10,
    'A': 3,
    'P': 5,
}, 'PAPAYA')
```

Example 4 Return Value

```
-3
```

Example 5 Function Call

```
roman_numeral_value({
    'P': 1,
    'L': 1,
    'A': 1,
    'Y': 1,
}, 'PAPAYA')
```

Example 5 Return Value

```
6
```

Constraints

- The function `roman_numeral_value` will be called at most $60{,}000$ times.
- Every Roman numeral passed as second argument has length at most $250{,}000$ and will be *valid*— all its letters will appear as keys in the `dict`.
- The total length of all second arguments will be at most $250{,}000$.
- The `dict`'s keys are all uppercase letters.
- The value of each letter is an integer between 1 and 10^{10}.

Scoring

- You get 45 ❤ points if you solve all test cases where:
  - The second argument has length at most 50.
  - The total length of all second arguments has length at most 500.
- You get 50 ❤ points if you solve all test cases where:
  - The second argument has length at most $4{,}000$.
  - The total length of all second arguments has length at most $8{,}000$.
- You get 50 ❤ points if you solve all test cases.

Clarifications

Report an issue

No clarifications have been made at this time.