Atlassian Sydney - Graduate 2018 Develo... 01h : 08m : 18s to test end

☆ Romanizer

Complete the function *romanizer*, that has one parameter, an array, *numbers*, of *n* integers. The function should return an array of n strings, such that i^{th} element in the array is the roman representation of i^{th} element of numbers.

You can use the roman representation of following integers:

Integer	*Roman Representation*
1	I
4	IV
5	V
9	IX
10	X
40	XL
50	L
90	XC
100	C
400	CD
500	D
900	CM
1000	M

Input Format

The first line of the input is an integer *n*, total number of elements in the array, numbers. Each of the next *n* lines contains a single integer, denoting the elements of the array.

Constraints

Output Format

The function should return an array of n strings, such that i^{th} element in the array is the roman representation of i^{th} element of numbers.

Sample Input 1

```
5
1
2
3
4
5
```

Sample Output 1

```
I
II
III
IV
V
```

Sample Input 2

```
5
75
80
99
100
50
```

Sample Output 2

```
LXXV
LXXX
XCIX
C
L
```

YOUR ANSWER

Atlassian

Sydney - Graduate 2018 Develo...

01h : 08m : 18s
to test end

Original code | Python 2

```
#!/bin/python

import sys
import os


# Complete the function below.


def  romanizer(numbers):
    dic = [None] * 1001
    dic[1] = 'I'
    dic[4] = 'IV'
    dic[5] = 'V'
    dic[9] = 'IX'
    dic[10] = 'X'
    dic[40] = 'XL'
    dic[50] = 'L'
    dic[90] = 'XC'
    dic[100] = 'C'
    dic[400] = 'CD'
    dic[500] = 'D'
    dic[900] = 'CM'
    dic[1000] = 'M'
    return [getRoman(dic, num) for num in numbers]

def getRoman(dic, num):
    if dic[num] is not None:
        return dic[num]

    for i in xrange(num,0,-1):
        if dic[i] is not None:
            return dic[i] + getRoman(dic, num - i)


f = open(os.environ['OUTPUT_PATH'], 'w')↔

_numbers_cnt = 0
_numbers_cnt = int(raw_input())
_numbers_i=0
_numbers = []
while _numbers_i < _numbers_cnt:
    _numbers_item = int(raw_input());
    _numbers.append(_numbers_item)
```

Atlassian Sydney - Graduate 2018 Develo... 01h : 08m : 18s to test end

```
res = romanizer(_numbers);
for res_cur in res:
    f.write( str(res_cur) + "\n" )

f.close()

```

Line: 29 Col: 15

Test against custom input

Run Code | Submit code & Continue

(You can submit any number of times)

Download sample test cases *The input/output files have Unix line endings. Do not use Notepad to edit them on windows.*