Roman Numerals

5E Hãy khảo sát cẩn thận chí ít 5 ví dụ

*Cho số La Mã x dưới dạng string gồm các chữ số I: 1, V: 5, X: 10,  
L: 50, C: 100, D: 500, M: 1000. Hãy chuyển đổi x qua dạng thập phân.*

Ví dụ

ToInt('MMXXI') = 2021

Algorithm

Người La Mã dùng bảy chữ số

I: 1, V: 5, X: 10, L: 50, C: 100, D: 500, M: 1000

để ghi các số theo quy tắc phổ biến sau đây.

Quy tắc

Nếu hai chữ số đứng cạnh nhau là AB thì:

* Quy tắc trừ: Nếu A < B thì AB = B − A.

Quy tắc này được áp dụng cho các cặp AB sau đây:

IV = 4, IX = 9

XL = 40, XC = 90

CD = 400 và CM = 900

Ta quy ước gọi quy tắc này là *trừ 4-9*

* Quy tắc cộng: Nếu A ≥ B thì AB = A + B.

Quy tắc này được áp dụng cho các dãy gồm 2 hoặc 3 chữ số tròn chục giống nhau như sau:

II = 2, III = 3,

XX = 20, XXX = 30,

CC = 200, CCC = 300,

MM = 2000, MMM = 3000

Ta quy ước gọi quy tắc này là *cộng 2+3.*

Hai quy tắc trên khá phổ biến. Ngoài ra, một số hệ thống ghi số La Mã có thể dùng thêm các phương án mở rộng khác mà ta không xét trong sách này.

Ta thiết lập tương ứng 1-1 giữa hai dãy giá trị, ival chứa các giá trị nguyên và rval chứa các giá trị nguyên Roman tương ứng như sau:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ival | 1000 | 900 | 500 | 400 | 100 | 90 | 50 | 40 | 10 | 9 | 5 | 4 | 1 |
| rval | M | CM | D | CD | C | XC | L | XL | X | IX | V | IV | I |

Mỗi lần ta xét hai chữ số La Mã kề nhau AB. Nếu AB xuất hiện trong rval tại vị trí i thì ta lấy ival[i] thêm vào tổng. Nếu AB không xuất hiện trong rval thì chắc chắn A phải xuất hiện trong rval tại vị trí i, ta thêm ival[i] vào tổng.

Lưu ý rằng người La Mã không dùng chữ số 0 nên ta tạm dùng ký tự rỗng '' hoặc dấu '.' để hiển thị cho số 0.

Roman Numerals Version 1: Roman To Integer

# Roman Numeral System: Rom to Int

# 27 Nov 2020

MN = 13

ival = [1000, 900, 500, 400, 100, 90, 50, 40, 10, 9, 5, 4, 1]

rval = ["M", "CM", "D", "CD", "C", "XC", "L", "XL", "X", "IX", "V", "IV", "I"]

def ToInt(r):

if r == '' or r == '.':

return 0

r += '.' # (1)

val = i = 0

while i < len(r):

if r[i] == '.':

break

if r[i:i + 2] in rval: # (2)

j, i = rval.index(r[i:i + 2]), i+2 # (3)

else:

j, i = rval.index(r[i]), i+1

val += ival[j]

return val

def Go(): # (4)

if input(' ? ') == '.'

exit(0)

# A P P L I C A T I O N S

INT = [39, 246, 789, 2421, 160, 207, 1009, 1066, 1776, 1918, 1954, 2014, 2021]

ROM = ['XXXIX', 'CCXLVI', 'DCCLXXXIX', 'MMCDXXI', 'CLX', 'CCVII', 'MIX', 'MLXVI', 'MDCCLXXVI', 'MCMXVIII', 'MCMLIV', 'MMXIV', 'MMXXI']

for i in range(len(INT)):

y = ToInt(ROM[i])

print(ROM[i], '->', y, ' =? ', INT[i])

if INT[i] != y:

print('Error ')

Go()

print('CORRECT.')

Giải thích

Dòng lệnh

r += '.' # (1)

thêm dấu chấm (.) vào cuối string biểu diễn số Rom r. Vì ta sẽ duyệt hai ký tự AB liền nhau trong r nên việc thêm dấu chấm làm phần tử gọi là *lính canh* vào cuối dãy sẽ không phát sinh lỗi *tràn chỉ số.*

out of index range

Dòng lệnh

if r[i:i + 2] in rval: # (2)

kiểm tra xem string gồm hai chữ số La Mã r[i] và r[i+1] có xuất hiện trong danh sách rval hay không ?

Nếu cặp hai ký tự AB = r[i:i+2] có trong rval thì ta tìm vị trí đầu tiên j là nơi r[i:i+2] xuất hiện trong rval và tăng i thêm 2 đơn vị để duyệt cặp ký tự tiếp theo.

Ngược lại, nếu hai ký tự AB = r[i:i+2] không xuất hiện trong rval thì chắc chắn ký tự đầu tiên A = r[i] phải xuất hiện trong rval. Ta tìm vị trí đầu tiên j là nơi r[i] xuất hiện trong rval và tăng i thêm 1 đơn vị để duyệt cặp ký tự tiếp theo.

Dòng lệnh

j, i = rval.index(r[i:i + 2]), i+2 # (3)

là phép gán song song đã thảo luận trong bài Burrows-Wheeler Transform.

Các dòng lệnh bắt đầu từ (4)

def Go(): # (4)

if input(' ? ') == '.'

exit(0)

mô tả hàm Go() như sau:

* Nhận một ký tự với dấu mời '?'
* Kiểm tra, nếu ký tự này là dấu chấm (.) thì dừng việc thực hiện chương trình.

Như vậy hàm Go làm nhiệm vụ *tạm dừng chương trình* để người dùng quan sát kết quả trung gian.

Chương trình test với hai danh sách được lập sẵn:

Danh sách INT chứa các số nguyên dương;

Danh sách ROM chứa các số La Mã tương ứng.

Kết quả

XXXIX -> 39 =? 39

CCXLVI -> 246 =? 246

DCCLXXXIX -> 789 =? 789

MMCDXXI -> 2421 =? 2421

CLX -> 160 =? 160

CCVII -> 207 =? 207

MIX -> 1009 =? 1009

MLXVI -> 1066 =? 1066

MDCCLXXVI -> 1776 =? 1776

MCMXVIII -> 1918 =? 1918

MCMLIV -> 1954 =? 1954

MMXIV -> 2014 =? 2014

MMXXI -> 2021 =? 2021

CORRECT.

Đệ quy (Recursive)

Bạn cũng nên thử phương án đệ quy trong Python. Trong hàm đệ quy ToInt đã dùng các ký pháp sau cho string biểu diễn số La Mã r:

r[0] phần tử đầu tiên của string r

r[1:] hậu tố của r sau khi bỏ qua phần tử đầu tiên r[0].

r[:2] tiền tố gồm hai phần tử đầu của r là r[0] và r[1].

r[2:] hậu tố của r sau khi bỏ qua hai phần tử đầu tiên r[0] và r[1].

# Roman Numeral System: Rom to Int

# 27 Nov 2020

MN = 13

ival = [1000, 900, 500, 400, 100, 90, 50, 40, 10, 9, 5, 4, 1]

rval = ["M", "CM", "D", "CD", "C", "XC", "L", "XL", "X", "IX", "V", "IV", "I"]

def ToInt(r):

if len(r) == 0:

return 0

if len(r) == 1:

return ival[rval.index(r)]

# len(r) >= 2

if r[:2] in rval:

return ival[rval.index(r[:2])] + ToInt(r[2:])

return ival[rval.index(r[0])] + ToInt(r[1:])

def Go():

if input(' ? ') == '.'

exit(0)

# A P P L I C A T I O N S

INT = [39, 246, 789, 2421, 160, 207, 1009, 1066, 1776, 1918, 1954, 2014, 2021]

ROM = ['XXXIX', 'CCXLVI', 'DCCLXXXIX', 'MMCDXXI', 'CLX', 'CCVII', 'MIX', 'MLXVI', 'MDCCLXXVI', 'MCMXVIII', 'MCMLIV', 'MMXIV', 'MMXXI']

for i in range(len(INT)):

y = ToInt(ROM[i])

print(ROM[i], '->', y, ' =? ', INT[i])

if INT[i] != y:

print('Error ')

Go()

print('CORRECT.')

Ta chuyển qua cài đặt hàm ToRome(n) chuyển đổi số thập phân n sang dạng Roman.

Hàm ToRome

ToRome(374) sẽ hoạt động theo kỹ thuật rót nước như sau:

Ví dụ

Giả sử bạn có một bình đựng n = 374 lít nước. Bạn cần rót hết nước từ bình vào 13 cốc với các dung tích cho trong mảng ival như sau:

cốc số 0: dung tích 1000 lít, nhãn M

cốc số 1: dung tích 900 lít nhãn CM

cốc số 2: dung tích 500 lít nhãn D

...

cốc số 12: dung tích 1 lít nhãn I

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A white pitcher with a white flower design  Description automatically generated | index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| ival | 1000 | 900 | 500 | 400 | 100 | 90 | 50 | 40 | 10 | 9 | 5 | 4 | 1 |
| rval | M | CM | D | CD | C | XC | L | XL | X | IX | V | IV | I |

Mỗi khi rót đầy một cốc bạn cần lấy một nhãn tương ứng từ mảng rval để gán cho cốc đó.

Bạn chỉ cần duyệt lần lượt từ cốc số 0 đến cốc cuối cùng, mỗi lần bạn kiểm tra lượng nước n có trong bình. Nếu n lớn hơn dung tích cốc thì bạn rót đầy cốc và gán nhãn cho cốc đó.

Với n = 374, bạn thu được

ba cốc nhãn C,

một cốc nhãn L,

hai cốc nhãn X và

một cốc nhãn IV

Bạn có

374 = 'CCC|L|XX|IV'

Vậy bạn có

def ToRome(n): # Int -> Rom

if n == 0:

return '.'

r = ''

for i in range(13):

while n >= ival[i]:

r, n = r + rval[i], n - ival[i]

return r

Bạn có thể thay phép trừ liên tiếp bằng hàm divmod.

def ToRome(n): # Int -> Rom

if n == 0:

return '.'

r = ''

for i in range(MN):

m, n = divmod(n, ival[i]) # m=n//ival[i], n=n%ival[i]

r += rval[i] \* m

return r

Roman Numerals Version 2: Roman To Integer and vise versa

# Roman Numeral System: ToRome and ToInt

MN = 13

ival = [1000, 900, 500, 400, 100, 90, 50, 40, 10, 9, 5, 4, 1]

rval = ["M", "CM", "D", "CD", "C", "XC", "L", "XL", "X", "IX", "V", "IV", "I"]

def ToRome(n): # Int -> Rom

if n == 0:

return '.'

r = ''

for i in range(MN):

m, n = divmod(n, ival[i])

r += rval[i] \* m

return r

def ToInt(r):

if len(r) == 0:

return 0

if len(r) == 1:

return ival[rval.index(r)]

# len(r) >= 2

if r[:2] in rval:

return ival[rval.index(r[:2])] + ToInt(r[2:])

return ival[rval.index(r[0])] + ToInt(r[1:])

def Go():

if input(' ? ') == '.':

exit(0)

# A P P L I C A T I O N S

for n in range(1, 4000):

x = ToRome(n)

y = ToInt(x)

print(n, '->', x, '->', y)

if n != y:

print('Error ')

Go()

print('CORRECT.')