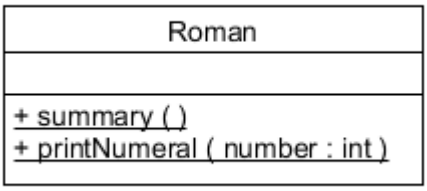
A4 Roman Numerals CSIS-1400

Learning Objectives:

- Use loops to implement repetitive actions
- Use `if-else` statements to provide different paths of execution
- Develop algorithms to produce the desired result
- Generate random numbers within a specified range



Description:

In this assignment you are going to display numbers as roman numerals, as described in this article .

Create a jGrasp project called **A4** that includes two java files: **Roman.java** and **RomanApp.java**
Implement class Roman as specified in the UML class diagram above. Pay special attention to the class name, method names, parameter lists, and return types. They all need to match the specification in the UML class diagram. No fields, public methods, or constructors should be added.
Notice the underlined operations. This indicates that the methods are static.

Roman.java:

Method `summary`:

Print the roman numerals that correspond to 1, 5, 10, 50, 100, and 1000 and the permitted subtractive forms (see expected output)
The summary should be printed in tabular form and the columns should be left aligned.
Use format specifiers to implement the columns and include a larger space to separate the numerals on the left from the subtracted forms on the right. This makes the summary easier to read.
Notice that there is no subtractive form in the first line.

Method `printNumeral`:

If the number passed to the method is between 1 and 39 (inclusive) print the roman numeral corresponding to the parameter value. Otherwise print: *`Valid number range: 1 - 39`*
In order to print numbers as roman numerals, you need to understand how the roman numerals work. Depending on the resource you might find slight variations. For this assignment I want you to use the description provided in this article: http://www.purplemath.com/modules/romannum.htm (*FYI: The article consists of two pages*).
Resist the temptation to hard-code the numerals for all 39 numbers. Instead look for patterns and use control statements (e.g. if, if-else, loops) to implement this method. Look for patterns that avoid code repetition.

Even though it is forbidden to add public methods to the class (it needs to be implemented exactly as specified in the UML class diagram) you are free to add private methods to structure your code.

RomanApp.java

RomanApp includes the `main` method and it is a test client for class Roman.

Hint: static methods should be called on the type NOT on an instance.

In order to test the class Roman do the following:

a) Display the summary

b) Use a loop to call the method printNumeral repeatedly.
Pass the numbers 0 to 40 (inclusive) as argument.
After each method call the cursor should be advanced to the next line.

c) Use a loop to print roman numerals corresponding to three random numbers between 10 and 20 (inclusive).

Your random numbers may differ from the output on the right.

All 3 parts should be separated by a single empty line for clarity (see output)

Turning in:

Zip up your project**.**

Ensure that your zip file includes both java files and that each java file includes a block comment that lists the author, the last date the assignment was modified, and the assignment number.

Turn it in via Canvas.

Expected Output:

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

Valid number range: 1 - 39
I
II
III
IV
V
VI
VII
VIII
IX
X
XI
XII
XIII
XIV
XV
XVI
XVII
XVIII
XIX
XX
XXI
XXII
XXIII
XXIV
XXV
XXVI
XXVII
XXVIII
XXIX
XXX
XXXI
XXXII
XXXIII
XXXIV
XXXV
XXXVI
XXXVII
XXXVIII
XXXIX
Valid number range: 1 - 39

3 random numbers between 10 and 20:
XIX
XV
XX
```