Roman Calculator Problem

Create a package named roman_calculator for this program.

You are to design and implement a Roman numeral calculator Class.
The subtractive Roman numeral notation commonly in use today was used only rarely during the time of the Roman Republic and Empire.
So we will ignore this aspect of Roman numerals.

For ease of calculation, the Romans most frequently used a purely additive notation in which a number was simply the sum of its
digits (4 equals IIII in this notation, not IV) .... this is what you should do as well.

Each number starts with the digit of highest value and ends with the digit of smallest value. This is the notation you will use in this program.

Your class inputs two Roman numbers and an arithmetic operator and prints out the result of the operation, also as a Roman number.
The values of the Roman digits are as follows:

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

Thus, the number MDCCCCLXXXXVI represents 1996, because 1996 really consists of:

1000 + 500 + 100 + 100 + 100 + 100 + 50 + 10 + 10 + 10 + 10 + 5 + 1.

M + D + C + C + C + C + L + X + X + X + X + V + I

Your code should accept lower case letters for Roman numerals.
MDCLXVI is the same as mdclxvi

The arithmetic operators that your program should recognize in the input are +, -, *, and /. These should perform the operations of integer addition, subtraction, multiplication, and division.

One way of approaching this problem is to convert the Roman numbers into integers, perform the required operation, and then convert the result back into a Roman number for printing.

The following might be a sample run of the class (user inputs are in bold italics):

Operator: + - * / q for quit: *+*
Enter operand1: *MCCXXVI*
Enter operand2: *LXVIIII*
Answer =MCCLXXXXV

Operator: + - * / q for quit

etc.

The program should check for errors in the input, such as illegal digits or arithmetic operators, and display an error message when these errors are found.

```
package roman_calculator;

// ****************************************************************
// *******    STARTING CODE  ***************************************
// ****************************************************************
import java.util.*;

public class RomanCalculator {
    // scan can now be used anywhere within this class
    Scanner scan = new Scanner(System.in);

    // This routine either returns false if the use wants to quit,
    //    or it does one Roman Calculator calculation
    boolean doCalculation()
    {
        char operator ;

        // Call getOperator to get either - + * / or q.
        // If q is returned, we return a false.

        // ************** FILL IN CODE

        // If the operator is + - * or / then call
        // getOperand(1) for the first operand and
        // call getOperand(2) for the second operand

        // ************** FILL IN CODE

        // call doArithmetic and print out the result using
        // convert_to_Roman to generate Roman Numeral output.


        return true;
    }

    // This routine prompts the user with
    // Operator: +  -  *  /  q for quit
    // If none of these are entered, this routine complains and
    //   prompts the user again.  Otherwise the operator is returned.
    char getOperator()
    {

        // ************** FILL IN CODE
    }

    // This routine prompts the user for either operand1 or operand2
    // depending on the value of which.  This routine uppercases the
    // input and calls convert_from_Roman to create an integer.
    // If the input is invalid ( negative return from convert_from_Roman)
    //    then complain and prompt the user again.
    int getOperand(int which)
    {
```

```
        // ************** FILL IN CODE
    }


    // Routine to convert an integer to a Roman Numeral String.
    // When you do this routine, you might find it handy to
    // create a utility routine that looks like:
    //  String addRomanDigit(String starting, int num, char digit)
    String convert_to_Roman(int value)
    {

        // ************** FILL IN CODE
    }


    // Convert Roman Numeral String to an integer.  If the
    // Roman Numeral String is invalid, return -1.
    int convert_from_Roman(String value)
    {

        // ************** FILL IN CODE
    }

    // Perform the arithmetic indicated by the operator (+ - * /)
    // and return answer
    int doArithmetic(int operand1, int operand2, char operator)
    {

        // ************** FILL IN CODE
    }

    public static void main(String[] args) {
        RomanCalculator rc = new RomanCalculator();
        while (rc.doCalculation())
        {
            System.out.println();  // blank line
        }
        System.out.println("Finished Roman Computations");
    }

}
```

Your Roman Calculator should be tested with the following inputs (note your program will fill in the answers):

```
Operator: +  -  *  /  q for quit: +
Enter operand1: M
Enter operand2: mCxV
Answer =_________

Operator: +  -  *  /  q for quit: -
Enter operand1: XX
```

```
Enter operand2: v
Answer =_________

Operator: +  -  *  /  q for quit: *
Enter operand1: x
Enter operand2: lx
Answer =_________

Operator: +  -  *  /  q for quit: /
Enter operand1: dc
Enter operand2: x
Answer =_________

Operator: +  -  *  /  q for quit: +
Enter operand1: mmc
Enter operand2: lxvii
Answer =_________

Operator: +  -  *  /  q for quit: -
Enter operand1: m
Enter operand2: cxi
Answer =_________

Operator: +  -  *  /  q for quit: ^
Your operand is bad ... try again:
Operator: +  -  *  /  q for quit: +
Enter operand1: axi
AXI has a bad character at position: 0
Enter operand1: xi
Enter operand2: xi8i
XI8I has a bad character at position: 2
Enter operand2: xii
Answer =_________

Operator: +  -  *  /  q for quit: q
Finished Roman Computations
```