```
package com.itbulls.learnit.javacore.methods.hw;

import java.util.Scanner;

public class ConvertDecimalToRoman {

	public static void main(String[] args) {
		Scanner sc = new Scanner(System.in);
		mainLoop: while (true) {
			System.out.print("Please, select mode. If you want to convert Roman "
					+ "numbers to decimal - type 'R2D' and press enter."
					+ System.lineSeparator()
					+ "If you want to convert decimal numbers to Roman - type
'D2R' and press enter: ");

			String mode = sc.next();
			if (mode.equalsIgnoreCase("R2D")) {
				while (true) {
					System.out.print("Please, enter Roman number you want to
convert: ");
					String romanNumber = sc.next();
					if (isRomanNumberValid(romanNumber)) {

	System.out.println(roman2Decimal(romanNumber));
						break mainLoop;
					} else {
						System.out.println("You entered invalid Roman
number. "
								+ "Please, try one more time.");
						continue;
					}
				}
```

```
                } else if (mode.equalsIgnoreCase("D2R")) {
                        while (true) {
                                System.out.print("Please, enter decimal number "
                                                + "you want to convert: ");
                                int decimalNumber = sc.nextInt();
                                if (isDecimalNumberValid(decimalNumber)) {

        System.out.println(decimal2Roman(decimalNumber));
                                        break mainLoop;
                                } else {
                                        System.out.println("Please, enter positive integer
from 1 to 100.");
                                        continue;
                                }

                        }

                }

                System.out.println("Please, enter 'R2D' or 'D2R.");
            }
        }



        /**
         * Converts decimal numbers to Roman.
         *
         * Takes int value as a parameter. Works only with numbers from 1 to 100.
         *
         * @param number to convert to Roman.
         * @return string of Roman number.
```

```
 */
public static String decimal2Roman(int number) {
	String[] a = new String[] { "I", "IV", "V", "IX", "X", "XL", "L", "XC", "C" };
	int[] b = new int[] { 1, 4, 5, 9, 10, 40, 50, 90, 100 };
	StringBuilder s = new StringBuilder();
	for (int i = a.length - 1; i >= 0; i--) {
		while (number >= b[i]) {
			s.append(a[i]);
			number -= b[i];
		}
	}
	return s.toString();
}

/**
 * Converts Roman numbers to decimal.
 *
 * Takes string value with Roman number as a parameter.
 *
 * At first method validates if input string could be Roman number. After method
 * uses algorithm to convert Roman numeral to decimal.
 *
 * @param romanNumber
 * @return decimal representation of Roman number
 */

public static int roman2Decimal(String romanNumber) {
	String romanNumeral = romanNumber.toUpperCase();

	int decimal = 0;
	int lastNumber = 0;
```

```
for (int x = romanNumeral.length() - 1; x >= 0; x--) {
        char convertToDecimal = romanNumeral.charAt(x);

        switch (convertToDecimal) {
        case 'C':
                decimal = processDecimal(100, lastNumber, decimal);
                lastNumber = 100;
                break;

        case 'L':
                decimal = processDecimal(50, lastNumber, decimal);
                lastNumber = 50;
                break;

        case 'X':
                decimal = processDecimal(10, lastNumber, decimal);
                lastNumber = 10;
                break;

        case 'V':
                decimal = processDecimal(5, lastNumber, decimal);
                lastNumber = 5;
                break;

        case 'I':
                decimal = processDecimal(1, lastNumber, decimal);
                lastNumber = 1;
                break;

        }
```

```
        }
        return decimal;

}

/**
 * Utility method which is the part of algorithm that converts Roman numbers to
 * decimal.
 *
 * @param decimal
 * @param lastNumber
 * @param lastDecimal
 * @return int value
 */
private static int processDecimal(int decimal, int lastNumber, int lastDecimal) {
        if (lastNumber > decimal) {
                return lastDecimal - decimal;
        } else {
                return lastDecimal + decimal;
        }
}

/**
 * Validation for Roman numbers.
 *
 * Use regular expression which is checking if string really could be Roman
 * number.
 *
 * @param romanNumber
 * @return true if String is Roman number
```

```
	 */
	public static boolean isRomanNumberValid(String romanNumber) {
		return romanNumber

	.matches("^(?i)M{0,3}(D?C{0,3}|C[DM])(L?X{0,3}|X[LC])(V?I{0,3}|I[VX])$");
	}

	public static boolean isDecimalNumberValid(int decimalNumber) {
		return decimalNumber > 0 && decimalNumber <= 100;
	}
}
```