# WORKSHEET #13

## Assumes: Ch1, Ch2, Ch3, Ch5, Ch6, Ch7, Ch8, Ch9, Ch10, Ch11

1. Draw UML diagram to show class hierarchy that represents various types of sales transactions in a store (cash, credit, check, etc.) Show what characteristics would be represented in the various classes of the hierarchy. Write an algorithm that shows how polymorphism plays a role in the payment process.
2. Using Object Oriented methodology, design and implement a program that creates an exception class called InvalidDocumentCodeException to be thrown when improper *designation* for a document is encountered during processing. Suppose in a particular business all documents are given a two-character *designation* starting with U, C, or P, standing for unclassified, confidential, or proprietary respectfully. If a document designation is encountered that does not fit that description, the exception is thrown. Design a Document class with the following attributes: *author*, *title*, *number of pages*, *creation date*, and *designation* as defined above. Create a text file documents.txt containing documents’ informstion – one line per document. The data is read from the file and saved in an ArrayList of Document objects that is the only instance variable of the DocumentLibrary class that you need to design and implement as well. The InvalidDocumentCodeException must be handled, so only valid Document objects are saved in the ArrayList. Your client program should print all documents, should print only documents with the designation specified by the user’s input, should print the oldest document, should print all documents with more than 50 pages – all should be accomplished by calling appropriate methods from the DocumentLibrary class.

Start by drawing the UML diagram for the classes; next create all the classes with the skeletons of each method. Finally implement the classes incrementally.

1. Using Object Oriented methodology, design a class encapsulating Roman numerals, say called Roman. An object of class Roman has one instance variable that stores a Roman numeral as a String. It has methods that can convert a Roman numeral into decimal; to convert decimal number to Roman numeral; print the number as a Roman numeral or decimal number as requested by the user. Do not forget to also implement the toString method for this class that returns the string containing the number in Roman format.

The decimal values of the Roman numerals are:

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

For simplicity, we assume that only the letter I can appear in front of another letter and that it appears only in front of the letters V and X. For example, 4 is represented as IV, 9 is represented as IX, 39 is represented as XXXIX, and 49 is represented as XXXXIX. Also, 40 is represented as XXXX, 190 is represented as CLXXXX, and so on.

Create ExtendedRoman class which is a subclass of Roman class. The class has methods add, subtract, multiply and divide so that arithmetic operations can be performed on Roman numerals.

To add (subtract, multiply, or divide) Roman numerals, add (subtract, multiply, or divide, respectively) their decimal representations and then convert the result to the Roman numeral format. For subtraction, if the first number is smaller than the second number, throw an exception, *“Because the first number is smaller than the second, the numbers cannot be subtracted”.* Similarly, for division, the numerator must be larger than the denominator.

Create also class InvalidRomanNumeralException to be thrown if the given numeral is not a valid Roman numeral. This exception must be properly handled by your program.

Your client class should be designed to thoroughly test the implemented functionality. Consider having a text file with input data that is either a Roman numeral or a decimal number, the data should be read in and saved in two text files: one containing all numbers converted to Roman numerals and the other containing the same numbers as decimal numbers. The client should also prompt the user for the input to perform arithmetic operations on Roman numerals.

1. ☺