Problem Set 2

(10 points) Cohort Exercise 2:

Given the pizza store example, i.e. PizzaStore.java, refactor the code using Factory Design Pattern.

(10 points) Cohort Exercise 3:

Assume that you are programming a simulator with multiple interacting robots. There are three types of robots: aggressive, defensive and normal. Apply strategy design pattern to complete the code skeleton: RobotGame.java.

(10 points) Cohort Exercise 4:

- Draw the class diagram for the Singleton pattern.
- What is the difference between the Singleton pattern and a class with all methods static?

(10 points) Cohort Exercise 5:

Complete DecoratorDemo.java by adding the ham topping.

(10 points) Cohort Exercise 6:

Given the ugly VisitorOriginal.java, study the Visitor Pattern by yourself, and apply the pattern to improve the code.

(20 points) Homework Question 1:
Assume that we would like to extend the observer pattern example presented in the class to support thousands of different stocks. The problem is that the current design is perhaps inefficient since a large amount of data needs to be sent (imagine that the data is sent over a network as often is the case in practice). Worse yet, it is unlikely that every user is interested in every stock and thus a large amount of data sent over is not used. Implement, based on the sample Java classes given in class, a potential remedy to this problem.

(10 points) Homework Question 2:

Write a Java program for a tic-tac-toe game. Two users take turns to make a move. The game is over after a winner is decided.

(10 points) Homework Question 3:

Write a Java program for a sequential election vote casting application as follows: There are two candidates A and B contesting an election. There are five electorates and each electorate can cast their vote only once and for only one of the two candidates (A or B). The five electorates cast their votes, i.e.

a character 'A' or 'B', one after another. The winner is the candidate who gets the maximum number of votes.

(10 points) Homework Question 4:

Write a Java program that takes a list of filenames (you can hard-code the file names in your program) and prints out the number of lines in each file.