Principles of Object Oriented Programming - Spring 2018 Assignment 1

Published: 20.3 Due: 3.4

General note: although it is possible to implement this assignment without objects (like any program, according to Church-Turing conjecture) you must implement it in an object-oriented manner, according the principles learned in the lectures and practical sessions. In addition, you must write your programs according to the coding-standards document that is published on the Assignments page of the course website.

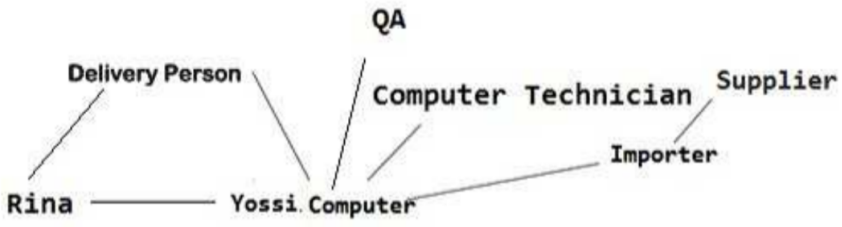
1 General Description

The assignment goal is to practice the following concepts

- Class design
- UML
- The message-passing concept in OOP

In this assignment you need to solve a problem using object-oriented programming, related to the story on the next page. You need to think of the classes that are required for the solution, and how they are related. You should first design the program classes using UML and only then implement the program.

General Description In a computer store, a customer orders a computer according to a specification: *CPU, Motherboard, Peripherals and Screen*. The sore orders the computer from an importer who orders the computer from a supplier. The supplier prepares an unassembled computer according to the specification and sends it back to the importer who sends it to the computer store. The computer store asks a technician to assemble the computer. After getting the assembled computer, it is delivered to a QA person who checks the computer and returns it back to the computer store. Finally, a delivery person provides the computer to the client.
You have to implement a simulation of the computer store presented below:



The links in the above figure represent connections between objects that should exchange messages (i.e., invoke methods) during the simulation. For example, the link between Importer and Supplier means that one of these objects should invoke a method of the other one.
You should perform the following steps (the percentage of each step in the assignment grade appears in brackets):

1. (10 %) Describe the **responsibilities** of **each agent** that appears in the figure.
2. (10 %) For **each pair of agents** that are connected by a link, decide who is the **client** and who is the **server**. (It is possible that an agent is a client on some link, and a server on another link.)
3. (30 %) Decide on the classes that the simulation should contain. Think about the similarities and differences of the various objects you have to implement, and design an appropriate UML class diagram. The hierarchy structure in your class diagram should be reasonable, according to the principles learned in the lectures and sessions. In particular, the description should contain the methods provided by each class.
4. (50 %) Implement the simulation according the design in parts 1 - 3, using the guidelines that appear in the next section.

2 Implementation requierments

In addition to the above classes, you should implement a class representing a **Computer**. The class should contain string representing the names of the computer parts, and two boolean fields representing whether the computer is assembled or not and whether the computer is tested or not.
Your program should produce the following output:

```
Client Rina orders computer from Yossi-Computer:  I7 Processor,Asus,Microsoft,LG.
Yossi-Computer forwards order to Importer.
Importer forwards request to Supplier
Supplier prepares an unassembled computer.
Supplier returns the computer to Importer.
```

```
 Importer returns the computer to Yossi-Computer.
 Yossi-Computer requests a computer assembling to the Computer Technician

 Technician assembles a computer.
 Technician returns the computer to Yossi-Computer.
 Yossi-Computer requests a quality check from QA.
 QA checks the computer.
 QA returns the computer to Yossi-Computer.
 Yossi-Computer forwards the computer to Delivery Person.
 Delivery Person delivers the computer to Rina.
 Client Rina receive:  Computer [I7 Processor, Asus, Microsoft, LG],
[assembled=true, installed=true].
```

Each object is allowed to produce only its output. In other words, a line of output that start with a name X can be produced only by X. (for example, the line "Yossi-Computer requests a computer assembling to the Computer Technician" can be produced only by Yossi-Computer). The object names should not be hard-coded in the message, but obtained using appropriate **getters**. The goal of the last requirement is making the program more general, and more flexible for modifications.

3 Submission instructions

Submit to the CS submission system a single archive file (.zip or .tar.gz) with the following contents:

1. hw1.jar - this file should contain the **source code** and **compiled class files**.

   (Check that you can run the file hw1.jar properly from the command line, as described in the submission instructions on the Assignments page.) Use the coding conventions given on the web site. Use packages for your classes: Do not use the default package.

2. hw1.pdf - a file with answers for tasks 1 - 3 of Section 1.