Assignment 1

Classes, objects, and interactions

Submit a single ZIP file called **assignment1.zip** which must contain an IntelliJ project with your Java files. **You should create the zip by going to File → Export to Zip File… in the IntelliJ menu within your project**. This assignment has 25 marks - see the marking rubric included on the assignment page.

Problem 1 (Electronics Store – Version 1)

Your goal for this problem is to create a simple implementation of an electronics store. You must implement the following five classes with the defined functionality:

Desktop Class: Must store the CPU speed (in Ghz as a double), amount of RAM (in GB as an integer), amount of storage (in GB as an integer), and whether or not the storage is an SSD or HDD (a Boolean, with true meaning SSD and false meaning HDD). The class must have a four-argument constructor that accepts an input parameter for each of these values in the same order they are listed above (CPU, RAM, storage, SSD). The class must have a toString method that returns a string indicating the object is a desktop PC along with the object's properties. Two examples of what the toString method could return are included below:

```
Desktop PC with 3.5ghz CPU, 8GB RAM, 500GB HDD drive.
Desktop PC with 3.0ghz CPU, 16GB RAM, 250GB SSD drive.
```

Laptop Class: Must store the CPU speed (in Ghz as a double), amount of RAM (in GB as an integer), amount of storage (in GB as an integer), whether or not the storage is an SSD or HDD (a Boolean, with true meaning SSD and false meaning HDD), and the screen size (in inches as an integer). The class must have a five-argument constructor that accepts an input parameter for each of these values in the same order they are listed above (CPU, RAM, storage, SSD, screen size). The class must have a toString method that returns a string indicating the object is a laptop PC along with the object's properties. Two examples of what the toString method could return are included below:

```
15" Laptop PC with 3.1ghz CPU, 32GB RAM, 500GB SSD drive.
13" Laptop PC with 2.5ghz CPU, 8GB RAM, 250GB HDD drive.
```

Fridge Class: Must store the size of the fridge (in cubic feet, as a double), whether it includes a freezer or not (a Boolean that is true if it has a freezer), and the color (as a String). The class must have a toString method that returns a string indicating the object is a fridge along with the object's properties. Two examples of what the toString method could return are included below:

```
15.6 cubic foot Fridge with Freezer (Gray)
10.5 cubic foot Fridge (White)
```

ElectronicStore Class: Must have an instance variable called **name** to store the name of the store. Must have a one-argument constructor that accepts a string to specify the store name. The constructor for this class must also create three instances of *each* of the previous three classes (9 items in total, using the constructors defined in those classes) and store them within the ElectronicStore instance being created. For storage purposes in the ElectronicStore class, you can use one or more arrays, lists, or hash maps to store the various products (your design choice). You can hard-code the argument values for the product constructors (i.e., CPU speed, RAM, color, etc.) or use a random number generator to randomly assign them. This class should also have a void method called **printStock()** that will iterate over all of the store's stock and print them to the console in a readable format. Additionally, this class should have a **searchStock(String)** method. The searchStock method should accept a string argument and return true if the toString() of any product in the store's stock contains the given string argument (otherwise, the method should return false). The searchStock method should also be case insensitive. That is, searches for "desk" and "hdd" should both return true if the store contains a "Desktop PC with 3.5ghz CPU, 8GB RAM, 500GB HDD drive".

ElectronicStoreTester Class: This class should have a single main method, which will first instantiate a single ElectronicStore and call the printStock() method. The test class should then repeatedly prompt the user to enter an item to search for. If the user enters the word "quit", this process should stop, and the program should end. Otherwise, the searchStock method of the ElectronicStore instance should be used to search for the given search term and the result should be printed out. You should use the searchStock method's return value to determine what message should be displayed. The output from this tester class should be similar to below (user input text is highlighted):

```
The store stock includes:
Desktop PC with 3.5ghz CPU, 8GB RAM, 500GB HDD drive.
Desktop PC with 3.0ghz CPU, 16GB RAM, 250GB SSD drive.
Desktop PC with 4.3ghz CPU, 32GB RAM, 500GB SSD drive.
15" Laptop PC with 3.1ghz CPU, 32GB RAM, 500GB SSD drive.
13" Laptop PC with 2.5ghz CPU, 8GB RAM, 250GB HDD drive.
15" Laptop PC with 3.0ghz CPU, 16GB RAM, 250GB SSD drive.
16.5 cu. ft. Fridge with Freezer (Black)
12.0 cu. ft. Fridge (White)
23.0 cu. ft. Fridge with Freezer (Stainless Steel)

Enter a term to search for: desk
A matching item is contained in the store's stock.
Enter a term to search for: LaPtOP
A matching item is contained in the store's stock.
Enter a term to search for: Toast
No items in the store's stock match that term.
Enter a term to search for: television
```

```
No items in the store's stock match that term.
Enter a term to search for: GB
A matching item is contained in the store's stock.
Enter a term to search for: quit
```