Software Development (CS2514) Assignment 1

Inheritance (Specifications subject to change; Due: March 14th; Marks: 50)

General Comments

Please, carefully read the submission guidelines before you submit the assignment.

This is an exercise about implementing *maintainable* classes. You should always assume the specifications may change (slightly) and you should make sure your implementation can implement these changes with the minimum amount of effort.

Before you start implementing your classes, please make sure you understand the API. If you do, this'll make your life much much easier.

Learning Objectives

For this assignment you will learn about inheritance in `Java`. You will learn how to design a class hierarchy for `Garments` and `Fabrics`. You should start by studying the dependencies between the different `Garment`/`Fabric` classes, by working out which are the more general `Garment`/`Fabric` types, and by determining which `Garment`/`Fabric` types are more specific than other `Garment`/`Fabric` types. As part of this exercise you may have to introduce intermediate `Garment`/`Fabric` classes and other classes that aren't mentioned in the specification. You will then translate your class hierarchy to a `Java` **inheritance**-based hierarchy, with the more general classes at the top of the hierarchy and the more specific classes at the bottom of the hierarchy. Please note that you are supposed to implement an **inheritance**-based hierarchy: you may not use interfaces or enumerated classes.

Your aim should be to maximise code re-use in the hierarchy, so *don't use copy-and-paste* to "implement" shared behaviour.

Main Details

In this assignment you will implement a `Garment` class hierarchy that has the class `Garment` sitting at the top of the hierarchy.

Each garment also has a name, which is a proper English name of the garment. There are three kinds of `Garments`: *trousers, shirts,* and *coats.* Each garment has `getName( )` behaviour, which returns the name of the garment. Garments also have `printPurpose( )` behaviour. This behaviour prints the purpose of the garment. The following are the purposes of the garments.

Trousers Cover the legs; The only trousers at the moment are *jeans.*

Shirts Cover the body. The only shirt at the moment is a *t-shirt.*

Coats Provide extra protection against the elements. The only coats at the moment are: *winter coat, rain coat,* and *jacket.*

Garment	**# of Units**	**Fabric**
Jeans	2.0	cotton
T-Shirt	1.5	cotton
Winter coat	2.5	polyester
Rain coat	2.5	acrylic
Jacket	2.0	tweed

Table 1: Garments and the number of units of fabric which they're made of

Natural Fabrics	**Price per Unit**
Cotton	2.0
Tweed	3.0
Synthetic Fabrics	**Price per Unit**
polyester	5.0
acrylic	6.0

Table 2: Fabrics and their (made up) price per unit of fabric

Each garment has a price which depends on the fabric which it is made of and the number of units of fabric which are needed to produce one item of the garment. A garment is made from a single *fabric,* which may be *natural* or *synthetic.*

Natural fabrics For natural fabrics (*tweed, cotton, ...*) the price of the garment is $c \times u$, where c is the cost of the fabric per unit and u is the number of units of fabric which are needed to produce one item of the garment.

Synthetic fabric For synthetic fabrics, such as *polyester* and *acrylic,* the price of the garment is $(c + e) \times u$, where c is the cost of the fabric per unit, e is the 2.0 Euro *environment tax* per unit of fabric, and u is the number of units of fabric which are needed to produce one item of the garment.

Table 1 lists each garment, the fabric which it's made of, and the number of units of farbic which are needed to produce one item of garment. Table 2 lists the kind of each fabric (natural or synthetic) and the unit price of the fabric.

Natural fabrics also have a *source,* which provides additional information about the origin of the material. For example the source of tweed and cotton, which are the only natural fabrics, are wool for tweed and Gossypium for cotton. You get the source of a natural fabric with its `getSource( )` method. Synthetic fabrics don't have a source.

Each garment can print an itemised bill of its name and price with a method called `void printItemisedBill( )`.

Additional Implementation Details

Please remember that your classes should respect encapsulation, so all class and instance attributes should be `private`. Using `private` attributes is part of the challenge. If you use attributes that are not `private` you can lose up to 50% of the marks.

Each class in your implementation should be in its own `.java` source file. Multiple classes per source file are not allowed. You may lose up to 50% of the marks if you submit files consisting of more than one class.

You should provide proper `JavaDoc` comments for all public instance methods.

Example

The file `Main-Assignment1.java` which is posted on the CS2514 assignment section *Canvas* provides an example of a `Main`. The file `OUTPUT-Assignment1` shows the output.

Your `Main` class should create instances of all garments listed in this assignment.

Submission Details

- Remember that each class should be in its own file. You will lose marks if you violate this rule.
- Remember that all classes, attributes, and public methods should be commented using a proper `JavaDoc` comment.
- The `JavaDoc` class comment should explain the purpose of the class; *not the purpose of the assignment.*
- Please provide your name and student ID as part of your class `JavaDoc`. You can use the `@author` tag for this:

```java
@author Java Joe (ID 12345678)
```

- Use the CS2514 `Canvas` section to upload your classes as a single *.tgz* archive called *Lab-1.tgz* before 23.55pm, March 14th, 2020. To create the *.tgz* archive, do the following:
  - ⋆ Create a directory `Lab-1` in your working directory.
  - ⋆ Copy `Main.java` and your other user-defined Java files into the directory. Do not copy any other files into the directory.
  - ⋆ Run the command '`tar cvfz Lab-1.tgz Lab-1`' from your working directory. The option '`v`' makes `tar` very chatty: it should tell you exactly what is going into the `.tgz` archive. Make sure you check the `tar` output before submitting your archive.
  - ⋆ Note that file names in `Unix` are case sensitive and should not contain spaces.
- Note that the format of your submission should be `.tgz`: do *not* submit `zip` files, do *not* submit `tar` files, do *not* submit `bzip` files, and do *not* submit `rar` files. If you do, it may not be possible to unzip your assignment.