COMP249/2 – Section D – Fall 2015

ASSIGNMENT #1
Due: October 5th

Purpose: The purpose of this assignment is to allow you practice inheritance, as well as other subjects in relation to it, such as default and other constructors, access rights, method overriding, etc. The assignment would also allow you to practice the creation and use of packages.

General Guidelines When Writing Programs:

- Include the following comments at the top of your source codes

```
// -----------------------------------------------------------
// Assignment (include number)
// Question: (include question number)
// Written by: (include you name and student id)
// -----------------------------------------------------------
```

- In a comment, give a general explanation of what your program does. As the programming questions get more complex, the explanations will get lengthier.
- Include comments in your program describing the main steps in your program.
- Display a welcome message which includes your name(s).
- Display clear prompts for users when you are expecting the user to enter data from the keyboard.
- All output should be displayed with clear messages and in an easy to read format.
- End your program with a closing message so that the user knows that the program has terminated.

Part 1

The description of various publications is given as follow. A **PaperPublication** class concerns of the following: *price* (double type) and *number of pages* (int type).

A **Book** is a **PaperPublication** that additionally concerns with the following: an *ISBN* (long type), an *issue year* (int type), a *title* (String type), and *author(s) name* (String type).

A **ChildrenBook** is a **Book** that additionally concerns with the following: *minimum age* (int type), which indicates the minimum age that this book is expected for.

An **EducationalBook** is a **Book** that additionally concerns with the following: *edition number* (int type) and *speciality field* (String type), such as pharmaceutical, engineering, commerce, etc.

A **Journal** is a **PaperPublication** that additionally concerns with the following: *issue number* (int type) and *speciality field* (String type), such as pharmaceutical, engineering, commerce, etc.

A **Magazine** is a **PaperPublication** that additionally concerns with the following: *paper quality* (enumeration type that can be: *High*, *Normal*, or *Low*), and *issuing frequency* (enumeration type that can be: *weekly*, *Monthly*, or *Yearly*).

For that part, you are required to:

1. Draw a UML representation for the abovementioned class hierarchy. Your representation must also be accurate in terms of UML representation of the different entities and the relation between them. You must use a software to draw your UML diagrams (no hand-writing/drawing is allowed);
2. Write the implementation of the above mentioned classes using inheritance and according to the specifications and requirements given below;
3. You must have 4 different Java packages for the classes. The first package will include the **PaperPublication** class. The second package will include the **Book**, **ChildrenBook** and **EducationalBook** classes. The third package will include the **Journal** class and the last package will include the **Magazine** class.
4. For each of the classes, you must have at least two constructors, a default constructor, and a parametrized constructor, which will accept enough parameters to initialize ALL the attributes of the created object from this class. For instance the parametrized constructor of the **ChildrenBook** class must accept 7 parameters to initialize the *price*, the *number of pages*, the *ISBN*, the *issuing year*, the *title*, the *author name*, and the *minimum age*.
5. An object creation using the default constructor must trigger the default constructor of its ancestor classes, while creation using parametrized constructors must trigger the parametrized constructors of the ancestors.
6. For each of the classes, you must include at least the following methods: accessors, mutators, ***toString()*** and ***equals()*** methods (notice that you are always overriding the last two methods). The ***toString()*** method must display clear description and information of the object (i.e *"This Children Book has 57 pages, and costs 28$. It is suitable for age 4 and up ....."*). The ***equals()*** method returns true if all attributes of the two compared objects have the same values; otherwise it returns false.
7. For all classes, with the exception of the **Book** class, you are required to use either private or packages access rights. For the **Book** class, you are required to use protected access rights.
8. When accessing attributes from a base class, you must take full advantage of the permitted rights. For instance, if you can directly access an attribute by name from a base class, then you must do so instead of using a public method from that base class to access the attribute.

9. Write a driver program (where the main() method is) that would utilize all of your classes. The driver class can be in a separate package or in any of the already existing four packages. In the main() method you must:
    a. Create various objects from the 6 classes, and display all their information using the ***toString()*** method;
    b. Test the equality of some to the created objects using the ***equals()*** method;
    c. Create an array of 10 **PaperPublication** objects and fill that array with various objects from the 6 classes (each class must have at least one entry in that array);
    d. Trace that array to find the object that has the cheapest price. Display all information of that object along with its location (index) in the array.

Part 2

For that part, you will need to modify your implementation from Part1 as follows:

1. All classes must have the most restrictive (secure/protective) access rights to their attributes. Adjust your implementation from Part1 accordingly;
2. Modify the ***equals()*** method of the classes so that the method would first check if the passed object (to compare to) is both not null and that it is of the same type of the calling object. The method would clearly return false if any of these conditions is true; otherwise the comparison of the attributes are conducted to see if the two objects are actually equal.
3. In the driver program, you must add sufficient code to test your additions/changes.

Submitting Assignment #1

- Zip together the source codes for part 1 and part 2. (Please use ARCHIVE tool).
- Naming convention for zip file: Create one zip file, containing all source files for your assignment using the following naming convention:
  The zip file should be called *a#_studentID*, where # is the number of the assignment *studentID* is your student ID number. For example, for the first assignment, student 123456 would submit a zip file named *a1_123456.zip*
- Submit your zip file at: https://fis.encs.concordia.ca/eas/ as **Programming Assignment** and submission **#1**. Assignments submitted to the wrong directory would be discarded and no replacement submission will be allowed.
- Submit only **ONE version** of an assignment. If more than one version is submitted the first one will be graded and all others will be disregarded.

Evaluation Criteria or Assignment #1 (5 points)

Part 1 (4 points)	
UML representation of the class hierarchy	0.5 pt
Proper use of packages	0.5 pt
Correct implementation of the classes	0.5 pt
Constructors	0.5 pt
Accessor/mutator methods	1 pt
toString() & equals()	0.5 pt
Driver program & general correctness of code	0.5 pt
Part 2 (1 point)	
Access rights	0.5 pt
equals()	0.5 pt