CSCI 24000 – Fall 2017
Assignment #5 – A Healthy Dose of Inheritance
Due: 11/20/2017

This fifth assignment will allow you to explore the important OO concept of Inheritance in a Java implementation. Through this assignment, you will be able to explore this very important pillar (and potentially all four pillars…) of the OO paradigm and it will allow you to ramp up to the exciting, fun, yet very challenging sixth assignment.

For this assignment, you are tasked with the responsibility of building the framework for the Healthy Pharmacy that we discussed in lab/recitation. The Healthy Pharmacy is looking to develop a software application to model their employee hierarchy. Currently the pharmacy has the following types of job titles for their employees: Staff Pharmacist, Pharmacy Manager, Staff Technician, and Senior Technician. Healthy Pharmacy has elected to proceed with an hourly payment schedule for all employees – obviously the rates for each job title, degrees, and skills will vary (e.g., a Pharmacy Manager will make more per hour than a Technician). Every employee will have the following attributes:

- Employee ID (Integer)
- First Name (String)
- Last Name (String)
- Hourly Rate (Double)

A Pharmacy Manager will have the following additional attribute(s):

- Leadership (Boolean)

A Staff Pharmacist will have the following additional attribute(s):

- Licensed (Boolean)

A Senior Technician will have the following additional attribute(s):

- Service Award (Boolean)

A Staff Technician will have the following additional attribute(s):

- Degree (Boolean)

This program will be written in **Java** and must compile and run on **Tesla** (tesla.cs.iupui.edu).

Your program will be menu driven in which you will provide the following prompts to the user:

```
1. Load Employees (From File)
2. Exit Program
```

Once the user has loaded the employees the following options should be provided:

```
1. Print Employee Information
2. Enter Hours Worked
3. Calculate Paychecks
4. Exit Program
```

The text file containing employee data will be provided to you. The filename will be **employees.txt** – this file can be found (and downloaded) on Canvas. The file will contain **four** (4) employees – one in each role. These roles and their associated Role ID are shown below.

The following is the hourly rate for each respective job title – this is a predefined and set amount:

- Pharmacy Manager (ROLE ID=1): $50/hour
- Staff Pharmacist (ROLE ID=2): $40/hour
- Staff Technician (ROLE ID=3): $20/hour
- Senior Technician (ROLE ID=4): $25/hour

Development Process:

All development must take place on the **master branch** in a **private** GitHub repository. You must add Me (rrybarcz) and all four (4) TA's as collaborators. It is required that you commit and push often! You also need to provide useful comments on your commits. We will be checking to make sure that you are actively pushing changes to your repository – failure to do so will result in a loss of points. You are also required to include the Honor Pledge and Digital Signature in each one of your source files (any file that you have written "code" in – you need to include the Honor Pledge and Digital Signature).

You are required to implement the following items as part of this assignment:

- Your program must contain at-least **six** (6) classes.
  - Driver
  - Employee
  - Pharmacy Manager
  - Staff Pharmacist
  - Staff Technician
  - Senior Technician
- You must use the concept of **Inheritance** to model the relationships between the classes – this should be present in your source code as well as your UML Class Diagram.
- You are **required** to handle all exceptions (I/O).

```
1. Load Employees (From File)
2. Exit Program
Enter Your Selection: 1

File Load Failed!
java.io.FileNotFoundException: employees.txt (The system cannot find the file specified)

Program Exiting...
```

- You are **required** to use Arrays to store each of the respective employees when you load them from the text file – in the basic submission each Array will hold only one employee.
- The hours worked (Option #2) is the number of hours worked for each employee – here we assume all employees work the same amount of hours each week.
- If the user attempts to print the paychecks (Option #3) prior to entering the amount of hours worked an error message should be displayed. See below:

```
1. Load Employees (From File)
2. Exit Program
Enter Your Selection: 1

1. Print Employee Information
2. Enter Hours Worked
3. Calculate Paychecks
4. Exit Program
Enter Your Selection: 3

Please enter the hours worked (Option #2) before trying to calculate the paycheck amounts!
```

- You are **required** to submit an algorithm – your algorithm should outline the step-by-step process you have taken to solve this problem.
- You are also **required** to provide a UML Class Diagram with this project – you can achieve this by using a commercial tool (e.g., Microsoft Visio), but you are not required to use a commercial tool to complete this component.
    - This diagram should outline the classes (**name**, **attributes**, and **methods**) and relationships between the various classes. This diagram should match the program (code) that you submit.
    - You are **required** to specify the visibility of each attribute and method in your Class Diagram (e.g., public/private/protected).

Below is the output of what your program should display when executed:

```
1. Load Employees (From File)
2. Exit Program
Enter Your Selection: 1

File Successfully Loaded!

1. Print Employee Information
2. Enter Hours Worked
3. Calculate Paychecks
4. Exit Program
Enter Your Selection: 1

ID: 90000123     Name: John Jones          Rate: 50         Licensed: true           Has Leadership: true
ID: 90012342     Name: Crystal Konrad      Rate: 40         Licensed: true
ID: 90001234     Name: Brittany Willis     Rate: 25         Has Degree: true         Has Service Award: true
ID: 90006783     Name: Tim Duncan          Rate: 20         Has Degree: true

1. Print Employee Information
2. Enter Hours Worked
3. Calculate Paychecks
4. Exit Program
Enter Your Selection: 2

Please enter the hours worked: 10

1. Print Employee Information
2. Enter Hours Worked
3. Calculate Paychecks
4. Exit Program
Enter Your Selection: 3

ID: 90000123     Check Amount: $500.0
ID: 90012342     Check Amount: $400.0
ID: 90001234     Check Amount: $250.0
ID: 90006783     Check Amount: $200.0

1. Print Employee Information
2. Enter Hours Worked
3. Calculate Paychecks
4. Exit Program
Enter Your Selection: 4

Goodbye!
```

Submission:

All assignments must be submitted on IU GitHub (github.iu.edu) in your master branch by the due date. The name of your IU GitHub repository must be as follows: **csci24000_fall2017_A5**. You should be submitting the source files (e.g., ***.java**), a Makefile (e.g., **makefile**), a UML class diagram for your program, as well as a file (e.g. **algorithm.txt**) outlining your algorithm for accomplishing this assignment. Please be sure to have us added as collaborators to your assignment repository prior to your submission. If you submit a late, it is your duty to notify me (Dr. Rybarczyk) of the late submission – we will not be checking for late submissions without proper notification. Failure to follow any of these rules will result in a **zero** (0) on your assignment submission.

Blackbelt Challenge(s):

For this project the Blackbelt extensions are as follows:

- Allow the text file to include more than just four (4) entries and provide a mechanism for the program to handle this dynamically – (+5).
- Allow the hours worked to vary from employee to employee in the program – (+5). You will need to determine how this is handled at the menu.