COMP1161

LAB3

PREAMBLE

This lab aims to reinforce the concept of inheritance. The specific skills to be covered are:

1. Extending a class to create a new class
2. Overriding methods
3. Writing constructors in a subclass

The lab also aims to demonstrate the power of inheritance in managing collections.

THE PROBLEM

The internet of things, or IoT, is a system of interrelated computing devices, mechanical and digital machines, objects, animals or people that are provided with unique identifiers (UIDs) and the ability to transfer data over a network without requiring human-to-human or human-to-computer interaction. IoT concepts have enabled actualization of SmartHome, which allow common devices within a home to incorporate processing and communication capability, and together these devices are should be able to increase the satisfaction of the user. Examples of IoT devices that could be in a home could include a refrigerators, a light bulbs and smart phones. For this lab you will create a set of classes that can model a smart home, using the concepts of inheritance and aggregation. The lab can be found online at https://www.hackerrank.com/comp1161-lab3-25.

IN-LAB MARKING CRITERIA
(PLEASE REVIEW BEFORE STARTING)

Criterion	Mark(s)
Explanation of the implications of using static attributes and methods.	1
Correct implementation of constructors (super class and sub class)	1
Identification of use of inherited attributes and methods	1
Describe the implication of overriding	1
Program readability - ie. good coding style	1
A PENALTY OF 1 MARK WILL BE APPLIED FOR ANY OF THE FOLLOWING: □ Shadowing	

LAB EXERCISES

Your tasks for this lab include

a. Completing classes InternetThing, SmartPhone and LightBulb

b. Editing in the SmartHome class to refer to newly defined classes

c. Editing the Refrigerator class to override the getPowerUse method.

1. Fix the compile time error in the starting code. Your implementation is to contain an private attribute called numThings in the InternetThing class, that stores the total number of InternetThings which have been instantiated.

2. Write code for the InternetThing class to complete:

   a. A constructor that accepts a manufacturer and a serial number as its parameter. This constructor performs the following actions:

      i Store the serial number as a string that is equal to the argument serial.

      ii Store the manufacturer as a string that is equal to the argument manufacturer.

      iii Set the id number to numThings,

      iv Set the IP address to the value "192.168.0."+idnumber

      v Set the powerUse to 1.

      vi Set the password to "admin".

      vii Prints the value "Created "+ the result of the toString() method after assigning all instance data.

      viii Increment the number of InternetThings (make this the last instruction in the constructor).

3. In method addThing of class SmartHome, ,

   - Set the item in array **things** which is at the position rf.getId() to the newly created refrigerator object. (Hint – note how the SmartPhone at the appropriate position of **things** is set).
   - Uncomment the line that sets returnval to rf.getId().

4. Complete the **SmartPhone** class. **SmartPhone** is an **InternetThing** that includes:

   a. A constructor which accepts a manufacturer(string),), a serial number(String) , a model(String) and the number of associated megapixels(int) as its parameters. This constructor will initialize the superclass and store each argument appropriately. It will then set the locked attribute on the SmartPhone to true.. After assigning all instance data, the constructor should print the value "Created "+ the result of the toString() method.

   b. A toString method that returns data in the format "Thing#"+getId()+"::PHONE made by "+***manufacturer***+":Model="+***model***+"@IP:"+getIPAddress();

5. Complete the constructor for class **LightBulb**. **LightBulb** is an **InternetThing**. The constructor that accepts a manufacturer(String), a serial number(String), and a lumenCount(int). It then initializes the superclass and instance attributes. lightOn is to be set to false. The constructor then prints the value "Created "+ the result o`f the toString method.

6. In class **LightBulb**, override the method named getPowerUse() . getPowerUse() should return the lumenCount multiplied by the power use of an InternetThing ONLY IF the light is on. If the light is off, getPowerUse() returns 0.

7. In the **Refrigerator** class, override the method getPowerUse() so that it returns a value that is equal to basePowerUse+capacity*powerRating as an integer. Truncate power use down to the previous integer.

8. In the **SmartHome** class, Observe the operation of the method showAllItems and then complete the method sumAllPower(). sumAllPower() returns the sum of the power use for all items in things. Specifically, you are to complete the loop, include the calculations for a running total to aggregate the power used by each InternetThing into the variable sumPower. At the end of the calculation, the method should output it's result in the format `"TOTAL POWER = "+sumPower+"mW",`where sumPower represents the aggregated power use.

9. Submission (You will not get marks for this submission. The submission is practice for actions you will need to perform when submitting project1 – your marks for the lab are earned from the Hackerrank exercise and from the scores assigned by lab techs)

   a. Save all classes in a separate file that matches the name of the class

   b. Zip the files, and make the name of your zip file Lab3.<yourIdNumber>.zip.

   c. Submit the zip file to the submission link for Lab 3.