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-lab3-23.

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 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. Writing 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. 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
      ii Set a unique id number as an integer*(hint- use the nextId)*.
      iii Set the IP address to the value "192.168.0."+idnumber
      iv Set a local representation of a the power used in mW to the integer 1.
      v Set the password to "admin".
      vi Prints the value "Created "+ the result of the toString() method after assigning all instance data.
      vii Increment the number of things on record.
   b. Accessors for all private variables – Ensure the accessors for id, password and powerUse are named getId(), getPassword() and getPowerUse()
   c. A method setPassword that accepts a string argument and sets the instance password to the argument.

   All instance variables, except the password, should be **private**. The password should be **protected**.

2. In class SmartHome, method addThing,
   - Create a refrigerator object by calling the relevant constructor with the following arguments (in order... manufacturer, serialNo, basePower, powerRating, capacity)
   - Set the value in array **things**, at the position that matches the ID of the refrigerator ie (rf.getId()), to the refrigerator object.

3. Write code for **SmartPhone** class which 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 shall store the each argument appropriately and will set the phone's status to locked. After assigning all instance data, the constructor should print the value "Created "+ the result of the toString method.
   b. A method named setPassword that accepts two strings (old password and new password) as its parameters. The method shall change the password to the new password value only if the old password given as a parameter matches the password stored on the phone, and if the phone is unlocked. If the password is successfully changed, the method will print the value "Successfully changed password for "+ the result of the toString method
   c. A method named lock that will lock the phone(), and print the value "Locked "+ the result of the toString method**.**
   d. A method named isLocked that returns the value true is the phone is locked.
   e. A method named unlock that accepts a string (the password) as its parameter and unlocks the phone only if the password is correct. If the phone becomes unlocked the method should print "Unlocked "+ the result of the toString method**.**
   f. A toString method that returns data in the format "Thing#"+getId()+"::PHONE made by "+<manufacturer>+":Model="+<model>+"@IP:"+getIPAddress();

4. Write code for a LightBulb class that is an InternetThing which includes:
    a. A constructor that accepts a manufacturer(String), a serial number(String), and a lumenCount(int) then sets values in the arguments parameter appropriately, and sets a property that reflects whether the light is on to false. After assigning all instance data, the constructor should print the value "Created "+ the result of the toString method.
    b. A method named turnOn() that turns on the light. The method should then print "Turned on "+ the result of the toString method**.**
    c. A method named turnOff() that turns off the light. . The method should then print "Turned off "+ the result of the toString method**.**
    d. In class **SmartHome**, method addThing, at the secton where the argument is checked for "LIGHTBULB", instantiate a LightBulb with a reference called lb, and then uncomment the lines in the below that add lb to things, then return the id.
5. In class LightBulb, override the method named getPowerUse() that evaluates current power used by the lightbulb as the product of it's lumenCount and the power use of it's base class if the light is on. If the light is off, getPowerUse() returns 0.
6. In the Refrigerator class, override the method getPowerUse() so that it returns a value that is equal to basePowerUse+contentSize*rating.
7. In the SmartHome class, Observe the operation of the method showAllItems and then complete the method sumAllPower(). For each item that is assessed while evaluating the total power, the method should print the power used by the item, a tab, and then the result of the toString method. 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.
8. 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.