COP-4259 Advanced Java Programming

Programming Assignment 1:
Polymorphism

Florida Atlantic University

1 Problem Specification

For this programming assignment, you will be completing the implementation of a java program for the checkout system of a grocery store which sells rice by pound, eggs by the dozen, baguette, and flavored baguette (baguette flavored with chocolate, vanilla, garlic, caramel, etc). This program is made of multiple classes distributed in two packages named 'util' and 'driver'.

To complete this program, you will implement an inheritance hierarchy of classes derived from the given util.Item abstract class:

- The util.Rice, util.Egg, and util.Baguette classes will be derived from the util.Item class.
- The util.FlavoredBaguette class will be derived from the util.Baguette class.

You will also write a driver.Checkout class which maintains a java.util.ArrayList⟨Item⟩ object of util.Item objects.

1.1 util.Item Class

The util.Item class inside util package is an abstract class from which specific types of Item can be derived. It contains only one data member, a name. It also defines two methods: getName and getCost. The getter of name in the util.Item class is final and getCost is an abstract method w/o implementation. Therefore, getCost method **needs to be overridden** in every subclass of class Item. Moreover, Tax amounts should be rounded to the nearest cent. For example, the tax on a food item with the cost of 199 cents with a tax rate of 2.0% should be 4 cents.

The Item.java file is given along with the assignment specification. Do *not* change the Item.java file! Your code must work with this class as it is provided.

1.2 util.GroceryStore Class

The util.GroceryStore class is given in the file util/GroceryStore.java. It contains some constant instant fields such as the tax rate as well the name of the store, the maximum length

of the string representing an item name and the maximum number of characters allowed to display the costs of items on the receipt. Your code should use these constants to generate the receipt appropriately. The util.GroceryStore class also contains the cents2dollarsAndCents method which takes an integer number of cents and returns it as a String formatted in dollars and cents. For example, 105 cents would be returned as "1.05".

Do *not* change the util/GroceryStore.java file. Your code must work with this class as it is provided.

1.3 Other util Classes

All other classes in util package will be subclasses of Item class. Please see the provided driver.TestCheckout class, in driver/TestCheckout.java file, to find out what input parameters the constructor of each of these classes will need to have. Each subclass should be implemented in a file with the correct name; e.g. util/Rice.java file will contain class util.Rice.

- The util.Rice class should extend the util.Item class. A Rice item has a weight and a price per pound which are used to determine its cost. For example, 2.30 lbs.of rice @ .89 /lb. = 205 cents. The cost should be rounded to the nearest cent.
- The util.Egg class should be derived from the util.Item class. An Egg item has a number and a price per dozen which are used to determine its cost. For example, 4 eggs @ 399 cents /dz. = 133 cents. The cost should be rounded to the nearest cent.
- The util.Baguette class should be derived from the util.Item class. A Baguette item simply has a cost.
- The util.FlavoredBaguette class should be derived from the util.Baguette class. The cost of a FlavoredBaguette is the cost of the Baguette plus the cost of its flavor.

1.4 driver.Checkout Class

The Checkout class provides methods to enter grocery items into the cash register, clear the cash register, get the number of items, get the total cost of the items (before tax), get the total tax for the items, and get a String representing a receipt for the grocery items (you may override the toString method of Object class). The Checkout class needs to store an ArrayList of Item objects. The total tax should be rounded to the nearest cent.

1.5 Testing

A simple test-driver, driver.TestCheckout.java, along with its expected output, is available to help you test your class implementations. You can add additional tests to the driver package to test your code extensively.

2 Submissions

You need to submit a *.zip* file compressing the following items:

- complete packages util and driver
- readme.txt explaining what each method of each class does in one of two sentences.