CSC1016S Assignment 2: Basic Java Classes

Assignment Instructions

This assignment involves building, testing and debug Java programs, that create and manipulate composition of user-defined types of objects comprising public instance variables. In most cases, the class definitions have been provided. Furthermore, where functions are used, the function definition will be given, or the signature of the function will be provided for you to complete.

The goal is to test classes and use the dot notation to access and use their instance variables and methods.

Furthermore**,** in future assignments, your solutions will be evaluated for correctness and for the following qualities:

- The use of object types, object creation, and the reading and writing of object fields.
- Documentation
    - Use of comments at the top of your code to identify program purpose, author and date.
    - Use of comments within your code to explain each non-obvious functional unit of code.
- General style/readability
    - The use of meaningful names for variables and functions.
- Algorithmic qualities
    - Efficiency, simplicity

These criteria will be manually assessed by tutors and commented upon. The number of marks to be deducted for non-compliance with the above will be communicated in each assignment in future.

Exercise 1: Shopping Cart [50 Marks]

This exercise involves the use of the `Item` and `ShoppingCart` classes of an object that maybe described as follows:

`Class Item`

An Item object represents a product being purchased and it is composed of product name, unit price and quantity.

Constructors

```
public Item(String productName, int quantity, double unitPrice)
        // Create an Item object from a given item purchased. An item can be either clothing or
        // groceries. An item has attributes product name, quantity and unit price.
```

Methods

```
public String getProductName()
        // returns the name of the item purchased
public double getUnitPrice()
        // returns the unit price of an item purchased
public int getQuantity()
        // returns the number of items purchased.
```

`Class ShoppingCart`

A ShoppingCart object represents the total amount to be paid at checkout. This includes the actual amount due, applicable value added tax and discount (if any).

Constructors

```
public ShoppingCart()
```

// Create a ShoppingCart object from an item purchased with the following attributes;

// total cost of items, applicable discount, applicable value added tax and the actual

// amount to be paid at checkout.

Methods

```
public void addItems(Item item)
```

// add items to the shopping cart

```
public void deleteItems(Item itemToDelete)
```

// remove any item(s) not needed from the shopping cart

```
public void queryCart()
```

// display all items in the shopping cart

```
public double getTotalAmount()
```

// calculate the total amount of all items in the shopping cart

```
public void getDiscount(String coupon)
```

// calculate an applicable discount on the items purchased based on the

//discount coupon you have.

```
public double getPayableAmount()
```

// amount paid at checkout when shopping has been completed

```
public void printInvoice()
```

// print the invoice (receipt) of all items purchased

Your Task: on the Vula page for the assignment, you will find the `Item` and `ShoppingCart` classes. Download these and use them to write a class called "`TestShoppingCart`" that has the main method to allow the user to capture details of item(s) purchased and print out an invoice for the customer.

Sample I/O:

```
How many items would you like to add to your Shopping Cart?:
0
Your Shopping Cart is empty.
```

Sample I/O:

```
How many items would you like to add to your Shopping Cart?:
2
Enter the Product Name:
Milk
Enter the Quantity:
16
Enter the Unit Cost:
16.99
Enter the Product Name:
Eggs
Enter the Quantity:
90
Enter the Unit Cost:
1.29
The Shopping Cart has the following items:
Milk: 16
Eggs: 90
--- Shopping Cart with all items ---
Milk 16 16.99 271.84
Eggs 90 1.29 116.10
  Total    :387.94
  Discount :77.59
  Tax      :43.45
  Total    :353.80
```

To capture details of more than one item, your program will require the use of a `for loop`. The Java syntax for a `for loop` is not very different from the Python one. For instance:

```
int i, quantity = 5; // note that 'i' can alternatively be declared and initialised
                     // inside the condition of the for loop (see line below)
for (i = 0; i < quantity; i++)  // or for(int i = 0; i < quantity; i++)
{
      System.out.println(i);
}
```

The code snippet above will print

```
0
1
2
3
4
```

Again, as with the `while loop`, the condition is enclosed in brackets. The body of the `for loop` (the code to be executed) is enclosed within curly brackets.

Exercise 2 [50 marks]

In this question, we will create and manipulate composition of user-defined types of objects comprising public instance variables ONLY.

NOTE: *The concept of instance variables will be covered in next week's lectures.*

Write a class called "`Seller`" that may be used to model shops with the following attributes:

Attribute name	**Sample Stored value**	**Description**
ID	PNP01	A unique key of the seller
Name	Pic n Pay	Name of the seller
Location	Rondebosch	Location of the seller
Product	Meat	The product to sell
unit_price	R49.99	Price per unit
number_of_units	1000	Number of available units

The class will only contain instance variables (we'll move on to constructors and methods in later assignments), one per attribute. Each variable should bear exactly the same name as the attribute it models. You must choose a suitable type for each variable based on the sample values. Hint: Money and Currency from assignment 1?

Based on the following sample I/O, write a simple program called "`TestSeller.java`" that asks the user to input seller details (as specified in the table), creates a Seller object and inserts the details in the appropriate fields, then retrieves and prints out those details.

Sample I/O

```
Please enter the details of the seller.
ID: UCT01
Name: UCT
Location: Rondebosch
Product: UCT T-shirts
Price: R300.00
Units: 120
The seller has been successfully created:
ID of the seller: UCT01
Name of the seller: UCT
Location of the seller: Rondebosch
The product to sell: UCT T-shirts
Product unit price: R300.00
The number of available units: 120
```

Submission

Submit the `TestShoppingCart.java`, `Seller.java` and `TestSeller.java` files to the automatic marker. The zipped folder should have the following naming convention:

`yourstudentnumber.zip`