Lab Cycle Problems

1. Create a class called Account to represent a bank account held by a customer (account holder). An account is characterized by the private fields – an account number (type String), the account balance (type double) and the name of the account holder (type String). The Account class should have a constructor that initializes the instance fields. Provide a *set* and a *get* method for each instance field. In addition, provide the deposit and withdrawal method and a method named computeInterest that calculates the interest earned by an account if the interest rate is 6% per annum. Test the class using the main method showing the invocation of deposit, withdrawal and computeInterest methods.

2. Create a class called Cab to represent a cab on hire. A cab should include four pieces of information as instance variables - a cab registration number (type String), name of the driver (type String), the rate per kilometer (type double) and the total distance it covered on hire on a day (type double). The Cab class should have a constructor that initializes the four instance variables. Provide a *set* and a *get* method for each instance variable. In addition, provide a method named computeIncome that calculates the daily income based on the total distance that it travelled on hire and returns the daily income as a double value. The registration number and driver name must be available and the rate per kilometer and total distance cannot be negative. Write a test application named Income that demonstrates creation of an array of objects of size three of the Cab class and display the daily income from each cab along with its registration number and driver's name.

3. Create a class called Invoice that a hardware store might use to represent an invoice for an item sold at the store. An Invoice should include four pieces of information as instance variables—a part number (type String), a part description (type String), a quantity of the item being purchased (type int) and a price per item (double). Your class should have a constructor that initializes the four instance variables. Provide a *set* and a *get* method for each instance variable. In addition, provide a method named getInvoiceAmount that calculates the invoice amount (i.e., multiplies the quantity by the price per item), then returns the amount as a double value. If the quantity is not positive, it should be set to 0. If the price per item is not positive, it should be set to 0.0. Write a test app named InvoiceTest that demonstrates class Invoice's capabilities.

4. Create a class called Employee that includes three instance variables—a first name (type String), a last name (type String) and a monthly salary (double). Provide a constructor that initializes the three instance variables. Provide a *set* and a *get* method for each instance variable. If the monthly salary is not positive, do not set its value. Write a test app named EmployeeTest that demonstrates class Employee's capabilities. Create two Employee objects and display each object's *yearly* salary. Then give each Employee a 10% raise and display each Employee's yearly salary again.

5. Create a class called Date that includes three instance variables—a month (type int), a day (type int) and a year (type int). Provide a constructor that initializes the three instance variables and assumes that the values provided are correct. Provide a *set* and a *get* method for each instance variable. Provide a method displayDate that displays the month, day and year separated by forward slashes (/). Write a test app named DateTest that demonstrates class Date's capabilities.

6. A parking garage charges a $2.00 minimum fee to park for up to three hours. The garage charges an additional $0.50 per hour for each hour *or part thereof* in excess of three hours. The maximum charge for any given 24-hour period is $10.00. Assume that no car parks for longer than 24 hours at a time. Write an application that calculates and displays the parking charges for each customer who parked in the garage yesterday. You should enter the hours parked for each customer. The program should display the charge for the current customer and should calculate and display the running total of yesterday's receipts. It should use the method calculateCharges to determine the charge for each customer.