CSE 1001: Introduction to Computer Programming

Programming Assignment-IX

(Classes and Objects)

1. Design a class **Student** with instance variables **name, roll, mark** and instance methods **setData(), display()**. Write a Java program to create three objects of Student class to input details of three different students and display the details. Enclose **main()** method inside another class **StudentDetails**. (Use the setter method setData() to input details.)

2. Design a class named **Rectangle** to represent a rectangle. The class contains:

   - Two **double** data fields named **width** and **height** that specify the width and height of the rectangle. The default values are **1** for both **width** and **height**.
   - A no-argument constructor that creates a default rectangle.
   - A constructor that creates a rectangle with the specified **width** and **height**.
   - A method named **getArea()** that returns the area of this rectangle.
   - A method named **getPerimeter()** that returns the perimeter.

Write a java program that creates two **Rectangle** objects—one with width **4** and height **40** and the other with width **3.5** and height **35.9**. Display the width, height, area, and perimeter of each rectangle in this order.

3. Design a class named **QuadraticEquation** for a quadratic equation $ax^2 + bx + c = 0$. The class contains:

   - Private data fields **a**, **b**, and **c** that represent three coefficients.
   - A constructor for the arguments for **a**, **b**, and **c**.
   - Three getter methods for **a**, **b**, and **c**.
   - A method named **getDiscriminant()** that returns the discriminant, which is $b^2 - 4ac$.
   - The methods named **getRoot1()** and **getRoot2()** for returning two roots of the equation

$$r_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a} \qquad r_2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

These methods are useful only if the discriminant is nonnegative. Let these methods return **0** if the discriminant is negative.

Write a java program that prompts the user to enter values for *a*, *b*, and *c* and displays the result based on the discriminant. If the discriminant is positive, display the two roots. If the discriminant is 0, display the one root. Otherwise, display "The equation has no roots."

4. Design a class named **Account** that contains:

- A private **int** data field named **id** for the account (default **0**).
- A private **double** data field named **balance** for the account (default **0**).
- A private **double** data field named **annualInterestRate** that stores the current interest rate (default **0**). Assume all accounts have the same interest rate.
- A private data field named **dateCreated** that stores the date when the account was created.
- A no-argument constructor that creates a default account.
- A constructor that creates an account with the specified id and initial balance.
- The accessor and mutator methods for **id**, **balance**, and **annualInterestRate**.
- The accessor method for **dateCreated**.
- A method named **getMonthlyInterestRate()** that returns the monthly interest rate.
- A method named **getMonthlyInterest()** that returns the monthly interest.
- A method named **withdraw** that withdraws a specified amount from the account.
- A method named **deposit** that deposits a specified amount to the account.

(*Hint*: The method **getMonthlyInterest()** is to return monthly interest, not the interest rate. Monthly interest is **balance * monthlyInterestRate**. **monthlyInterestRate** is **annualInterestRate / 12**. Note that **annualInterestRate** is a percentage, e.g., like 4.5%. You need to divide it by 100.)

Write a java program that creates an **Account** object with an account ID of 1122, a balance of Rs. 20,000, and an annual interest rate of 4.5%. Use the **withdraw** method to withdraw Rs. 2,500, use the **deposit** method to deposit Rs. 3,000, and print the balance, the monthly interest, and the date when this account was created.

5. Design a class named **Fan** to represent a fan. The class contains:

- Three constants named **SLOW**, **MEDIUM**, and **FAST** with the values **1**, **2**, and **3** to denote the fan speed.
- A private **int** data field named **speed** that specifies the speed of the fan (the default is **SLOW**).
- A private **boolean** data field named **on** that specifies whether the fan is on (the default is **false**).
- A private **double** data field named **radius** that specifies the radius of the fan (the default is **5**).
- A string data field named **color** that specifies the color of the fan (the default is **blue**).
- The accessor and mutator methods for all four data fields.
- A no-argument constructor that creates a default fan.
- A method named **toString()** that returns a string description for the fan. If the fan is on, the method returns the fan speed, color, and radius in one combined string. If the fan is not on, the method returns the fan color and radius along with the string "fan is off" in one combined string.

Write a java program that creates two **Fan** objects. Assign maximum speed, radius **10**, color **yellow**, and turn it on to the first object. Assign medium speed, radius **5**, color **blue**, and turn it off to the second object. Display the objects by invoking their **toString** method.

6. Design a class named **StopWatch**. The class contains:

- Private data fields **startTime** and **endTime** with getter methods.
- A no-argument constructor that initializes **startTime** with the current time.
- A method named **start()** that resets the **startTime** to the current time.
- A method named **stop()** that sets the **endTime** to the current time.
- A method named **getElapsedTime()** that returns the elapsed time for the stopwatch in milliseconds

Write a java program that measures the execution time of sorting 100 numbers using **bubble sort**.

****