I. Modify the Product Viewer application

This exercise guides you through the process of testing and modifying the Product Viewer application.

Review and test the project

1. Open the project named ex3_Product. Then, open the Product, ProductDB, and ProductApp classes and review their code.

2. Run the project and test it with valid product codes like "java", "jsp", and "mysql" to make sure that this application works correctly. Then, test it with an invalid code to see how that works.

Modify the ProductDB class

3. In the ProductDB class, modify the if/else statement so it includes another product.

4. Run the project and test it to make sure the new product code works. This shows that you can modify the code for a class without needing to modify the other classes that use it.

Add a constructor to the Product class

5. In the Product class, add a constructor that defines three parameters and uses them to set the values of the three instance variables.

6. In the ProductDB class, modify the code so it uses the new constructor to set the data in the Product object instead of using the setCode, setDescription, and setPrice methods. To do that, you can assign a new Product object to the Product variable within each if/else clause.

7. Run the project to make sure it still works correctly.

Add a method to the Product class.

8. In the Product class, add a method named getPriceNumberFormat that returns the price with number formatting (not currency formatting). This method should return the number with 2 decimal places but no currency symbol.

9. In the ProductApp class, modify the code so it uses this method.

10. Run the project to make sure it still works correctly.

Modify the ProductDB class so it defines an object

11. In the ProductDB class, modify the getProduct method so it's a regular method instead of a static method.

12. In the ProductApp class, modify the code so it creates a ProductDB object named db. Then, use this object to call the getProduct method of the ProductDB class.

13. Run the project to make sure it still works correctly.

II. Use objects in the Area and Perimeter application

This exercise guides you through the process of converting an Area and Perimeter application from a procedural application to an object-oriented application.

Create and use an object

1. Open the project named ex3_AreaAndPerimeter. Then, review the code for the Main class.

2. Create a class named Rectangle and store it in the informatics.rectangle package.

3. In the Rectangle class, add instance variables for length and width. Then, code the get and set methods for these instance variables. If possible, use your IDE to generate the get and set methods. With NetBeans, you can get started by selecting the RefactorEncapsulate Fields command.

4. Add a zero-argument constructor that initializes the length and width to 0.

5. Add a get method that calculates the area of the rectangle and returns a double value for the result. If you want, you can copy the code that performs this calculation from the Main class.

6. Add a get method that returns the area as a String object with standard numeric formatting and a minimum of three decimal places. To make it easy to refer to the NumberFormat class, you should add an import statement for it.

7. Repeat the previous two steps for the perimeter.

8. Open the Main class. Then, add code that creates a Rectangle object and sets its length and width.

9. Modify the code that displays the calculations so it uses the methods of the Rectangle object to get the area and perimeter of the rectangle.

10. Remove any leftover code from the Main class that's unnecessary including any unnecessary import statements.

11. Run the application and test it with valid data. It should calculate the area and perimeter for a rectangle.

Overload the constructor

12. Open the Rectangle class. Then, overload the constructor by supplying a second constructor that accepts two arguments: length and width. This constructor should set the length and width of the rectangle to the values supplied by these arguments.

13. Open the Main class. Then, modify its code so it uses this constructor instead of the zero-argument constructor.

14. Run the application and test it to make sure it still works correctly.