Basics of Java

This document details the requirement specifications for the above-named project. Reach out to your SME / Trainer for any query.

Technology	Java
Document Type	Basics of Java Practice Exercise – Day 1
Author	MLA
Current Version	1.0
Current Version Date	10-07-2024
Status	Active

Document Control

Version	Change Date	Change Description	Changed By
1.0	10-07-2024	Document Creation	Vanitha G

Problem Statement 1: Classes and Objects in Java

Solve following sub problems,

1.1 Write a program to list all, even numbers less than or equal to the number n. Take the value of n as input from the user.

1.2 Define a class Rectangle with its length and breadth. Follow the below steps,

a. Provide appropriate constructor(s), which gives facility of constructing Rectangle object with default values of length and breadth as 0 or passing value of length and breadth externally to constructor.

b. Provide appropriate accessor & mutator methods to Rectangle class.

c. Provide methods to calculate area & to display all information of Rectangle.

d. Design different class TestRectangle class in a separate source file, which will contain main method. From this main method, create a Rectangle object by taking all necessary information from the user and calculate respective area of rectangle objects and display it.

1.3 Create a class **Book** which describes its **book_title** and **book_price**. Follow the below steps,

a. Use getter and setter methods to get & set the Books description.

b. Implement createBooks and showBooks methods to create n objects of Book in an array.

c. Display the books along with its description as follows,

Book Title	Price
Java Programming	Rs 350.50
Let Us C	Rs.200.00

d. **Note: createBooks & showBooks** should not be member functions of **Book** class.

1.4 Modify the program, which is created in sub problem 1.2 as follows,

a. The class has attributes length and width, each of which defaults to 1.

b. It should have member functions that calculate the perimeter and area of the rectangle.

c. It should have set and get functions for both length and width.

d. The set functions should verify that length and width are each floating-point number larger than 0.0 and less than 20.0.

1.5 Create a **class Date** with **day**, **month**, and **year** attributes for manipulating dates.

Follow the below steps,

a. Provide a constructor that enables an object of this class to be initialized when it is declared (You can select any default values for the day, month & year, e.g., your birth date).

b. Provide the necessary functionality to perform error checking on the initializer values for data members day, month, and year.

c. Provide a member function to add an integer in a date to obtain a new date.

d. Design separate class Employee which will have following information. Refer below table,

Field Name	Data Type
employeeNumber	Int
employeeName	String
joiningDate	Date (User Defined Type)

e. Provide appropriate constructor(s) & methods to this class. Provide main function which willcreate 5 objects of Employee class and display employee information.

Problem Statement 2: Encapsulation and Inheritance in Java OOPs

Design a Java program to manage different types of vehicles (Car, Motorcycle, Truck) with the following features:

Vehicle Class:

- Create a superclass Vehicle with attributes such as manufacturer, model, year.
- Include methods to get and set these attributes (getManufacturer(), getModel(), getYear(), setManufacturer(String), setModel(String), setYear(int)).
- Implement a method displayDetails() to display all attributes of the vehicle.

Subclasses:

Car Class:

- Extend Vehicle and include an additional attribute seatingCapacity.
- Implement methods getSeatingCapacity() and setSeatingCapacity(int) to manipulate this attribute.
- Override the displayDetails() method to include seatingCapacity.
- Include specific operations such as accelerate(), brake().

Motorcycle Class:

- Extend Vehicle and include an additional attribute engineCapacity.
- Implement methods getEngineCapacity() and setEngineCapacity(double) to manipulate this attribute.
- Override the displayDetails() method to include engineCapacity.
- Include specific operations such as startEngine(), stopEngine().

Truck Class:

- Extend Vehicle and include an additional attribute cargoCapacity.
- Implement methods getCargoCapacity() and setCargoCapacity(double) to manipulate this attribute.
- Override the displayDetails() method to include cargoCapacity.
- Include specific operations such as loadCargo(), unloadCargo().

Main Class:

- Create instances of Car, Motorcycle, and Truck.
- Set values for their attributes and demonstrate the usage of methods like displayDetails(), accelerate(), brake(), startEngine(), stopEngine(), loadCargo(), unloadCargo().
- Ensure that each vehicle type behaves correctly based on its specific attributes and operations.

Problem Statement 3: Abstraction in Java OOPs

Design a Java program to calculate areas of different shapes (Circle, Rectangle, Triangle):

- Define an abstract class Shape with an abstract method calculateArea().
- Implement classes for each shape extending Shape and provide necessary attributes (radius and sideLength for circle, rectangle, and triangle respectively).
- Use abstraction to ensure that each shape class implements its own logic to calculate the area based on its specific attributes.

Problem Statement 4: Implement the Static Classes and Methods in Java

VastMindz Company is creating a performance rating system to calculate the performance of each employee. Design the below classes to achieve the same.

Employee class:

Field Name / Method Name	Data Type / Method Signature	Description
Name	String	Name of the Employee
Point	int	It provides points to the employee

Note: Use getter and setter methods to get and set the properties of an employee.

PerformanceRating class:

Field Name / Method Name	Data Type / Method Signature	Description
outstanding	int	Value will be 5
Good	int	Value will be 4
Average	int	Value will be 3
Poor	int	Value will be 2
calculatePerformance	static int calculatePerformance (Employee e)	Calculates performance of each employee based on some business rules

Note: The above variables must be constants.

Range	Rating
Point is between 80 - 100	Outstanding
Point is between 60 - 79	Good
Point is between 50 - 59	Average
Point is between 1 - 49	Poor

Develop the main class PerformanceCalculator. Create some Employee instances and set their attributes.

Calculate their performance rating based on the point they have achieved and print their respectiverating on the console. Count the number of employee instance created.

Output should be displayed as follows:

```
Total Number of Employees :3 and Their Ratings are

Oliver Has Performed with a Rating 5
================================================
Jayden Has Performed with a Rating 2
================================================
Daniel Has Performed with a Rating 4
================================================
```