Practice Problem Set – Class Diagrams

1. Develop class diagram for a class called `Circle` that contains:
   - Two private attributes: `radius` (of type float) and `color` (of type string).
   - Two public methods: `getRadius()` and `getArea()`, which return float type radius and area respectively.

2. Develop class diagram for a class `BankAccount` supporting the following methods:
   - `withdraw()`: takes an amount as input and withdraws it from the balance
   - `deposit()`: takes an amount as input and adds it to the balance
   - `balance()`: returns the balance on the account

   Make all the methods public and decide appropriate parameter and return types wherever needed. Also create appropriate private attributes.

3. Develop a class diagram for a class `Worker` that supports two private attributes `hoursWorked` and `wageRate`, and the following methods:
   - `setHoursWorked()`: a public method that takes the number of hours worked as input and sets `hoursWorked`
   - `changeRate()`: a public method that takes the new pay rate as input and changes the `wageRate` to the new hourly rate
   - `pay()`: an abstract public method

4. Add classes `HourlyWorker` and `SalariedWorker` as subclasses to class `Worker` defined in problem 4, each showing the inherited method `pay()` to compute the weekly pay for the worker. You may add any other attributes and methods to the subclasses. Use appropriate types for all attributes, method parameters and return values wherever needed.

5. Develop a class named `Vehicle` having three public attributes `noOfWheels`, `color` and `modelNo`, and related public getter/setter methods. Use appropriate types for all attributes, method parameters and return values wherever needed.

6. From the class `Vehicle` developed in problem 5, derive the following three subclasses:
   - Subclass `Car` having a public attribute airbag
   - Subclass `Bike` having a public attribute helmet
   - Subclass `Plane` having a public attribute wings

   Make getter/setter methods for each and decide appropriate types for all attributes, method parameters and return values wherever needed.

7. Modify the class diagram developed in problem 6, making the base class abstract by adding a protected method `findMileage()` to it, which returns a value for mileage of the vehicle. Consequently show this method in all subclasses. Also add public method `getMileage()` to the base class or the subclasses wherever you find appropriate. Decide appropriate return types for both these methods.

8. Develop a class `Engine` having private attributes `engineNo` and `dateOfManufacture`, along with appropriate public getter/setter methods. Associate this class to the class `Vehicle` created in problem 7. Decide on the relationship status, whether it is aggregation or composition. Decide appropriate types for all attributes, method parameters and return values for this new class.

9. Develop a class called `Shape` having a private attribute color and two public methods namely `getColor()` and `setColor()`. Now derive two child classes from class Shapes as follows:
   - Subclass `2DShape` having a private attribute `noOfSides` and public methods `findArea()` and `findPerimeter()` along with `getter()/setter()` methods.
   - Subclass `3DShape` having private attributes `noOfVertices`, `noOfFaces` and `noOfEdges` and public methods `findSurfaceArea()` and `findVolume()` along with `getter()/setter()` methods.

   Use appropriate types for all attributes, method parameters and return values.
   For the above class `Shape`, answer the following questions:
   i. What is the number of attributes and methods which an object of class `Shape` can access?
   ii. What is the number of attributes and methods which an object of class `2DShape` can access?
   iii. What is the number of attributes and methods which an object of class `3DShape` can access?
   iv. What is the number of attributes and methods which are accessible through the interface of this class hierarchy?
   v. Do you think the selection of methods in the subclasses is appropriate? Justify your answer.

10. Repeat problem 9 making the methods `findArea()` and `findPerimeter()` abstract in class `2DShape`, inheriting from it three sub-classes as follows:
    - Subclass `Square` having a private attribute `lengthOfSide`
    - Subclass `Circle` having a private attribute `radius`
    - Subclass `triangle` having private attributes `base` and `height`

    Make getter/setter methods for each and decide appropriate types for all attributes, method parameters and return values wherever needed.

11. Develop a class `Mammal` and derive at least five subclasses from it (e.g. Dolphins, Humans, Cats, etc). Add a third level to the hierarchy by extending at least two of the already defined subclasses (e.g. a class clothingStyle can be added to humans, or categories of cats could be added), showing appropriate relationships. For each class think of at least two attributes and one method (other than getter/setter methods). Decide appropriate access specifiers for each attribute and method.