Inheritance

**1.** Extend the Vehicle class by Car class and Car class by Duster class. Inherit the properties

from Vehicle and add new properties in Car and Duster.

**2.**Show is use of super keyword in Car and Duster.

**3.** Replace some existing properties of Vehicle in Car and Replace the some existing properties

of Car in Duster.

**4.**Show the use of Dynamic Method Dispatch in above inheritance.

**Dynamic method dispatch** is the mechanism by which a call to an overridden **method** is resolved at run time, rather than compile time.

**5.**Restrict the class Duster to extend further.

Final class

**6.**Restrict any of the method of Car to override.

Final method

**7.**Implement the Has-A relationship in between Car class and Engine class.

## Abstract Classes

**1.** Write a program to achieve partial abstraction in Vehicle class.

**2.**Declare the Vehicle class as an abstract and define some abstract methods in Vehicle. Implement the code to override the abstract methods.

**1.** Define the interface which shows the abstract functionality of Car. Implement this interface in Car. Define the final shared variable in interface.

**2.**Show the use of Extended interface, Nested interface, Functional interface and Marker

interface in above example.

## Packages

**1.**Store the Vehicle class into package named user, Car class in package named type, Duster class in package named model.

**2.** Control the visibility of some of the variables within the package only.

**3.**Import appropriate classes and show the execution of above [**inheritance**](http://gem.squadinfotech.in/mod/assign/view.php?id=703). Also show the use of static import.