S C H O –
E A ly

Important Instructions:

Please read the document thoroughly before you code.
Import the given skeleton code into your Eclipse.
Do not change the Skeleton code or the package structure, method names, variable names, return types, exception clauses, access specifiers etc.
You can create any number of private methods inside the given class.
You can test your code from main() method of the program
Using Spring Core develop the application using **XML Configuration**.

Time: 1 hour

Assessment Coverage:

Classes, Objects and Constructor Injection

Application created should be a demo of how to test petrol engine and diesel engine in a car. Car is totally independent of its engine , performance of the car is totally dependent on the engine fixed or injected . So the application will analyse the engine by injecting petrol or diesel engine in a runtime basis .

S 1 F 1 D 1 :

Import the below attached skeleton code into your eclipse project and implement the required functionalities

EngineAnalysis_CodeSkeleton.zip

T 1 R :

You are required to develop an App following below conditions.

Step 1: Create an abstract class **Engine** with below mentioned public methods :

Variables:

torque of type int , rpm of type int ,fuel of type String

Access Specifier/ Modifier	Method Name	Input Parameters	Output Parameters	Logic
Public abstract	getPerformance	nil	int	This method should be implemented by subclass And calculate the performance

Engine class should be registered as a **bean** as '**abstract= true**' with the spring container via **XML file**.

Step 2: Create class **PetrolEngine** which extends **Engine** and give implementation for **getPerformance** method by using the below formula and return horsepower.

horsepower = (torque * rpm)/5252

PetrolEngine class should be registered as a **bean** with the spring container via **XML file** with **bean id** as **petrolEngine**. The values for the attributes should be **torque=300** , **rpm=4000** and **fuel=petrol**.

Step 3: Create class **DieselEngine** which extends Engine and give implementation for getPerformance method by using the below formula and return horsepower.

horsepower = (torque * rpm)/63025

DieselEngine class should be registered as a **bean** with the spring container via **XML file** with **bean id** as **dieselEngine**. The values for the attributes should be **torque=500** , **rpm=3000** and **fuel=diesel**.

Step 4: Create class **Car** which has following methods and variables.

Variables:

name of type String , engine of type Engine

	Method Name	Input Parameters	Output Parameters	Logic
	getReport	nil	nil	This method should display the name of the car , fuel used and performance

Constructor	Car	String name , Engine engine	NA	This parameterized constructor takes name and Engine object which should be injected using XML file.

Car class should be registered as a **bean** with the spring container via **XML file** with **bean id** as **petrolCar**. The values for the attributes should be constructor injected with **name=Honda** and **engine referred to petrolEngine** bean**.**

Car class should be registered as a **bean** with the spring container via **XML file** with **bean id** as **dieselCar**. The values for the attributes should be constructor injected with **name=Suzuki** and **engine referred to dieselEngine** bean**.**

General Design Constraints:

Ensure that all the Java Coding Standards are followed.
Assume that the method inputs are valid always, hence exceptional blocks are not needed to be included in the development.

Sample Input Output 1:

```
Select option
 1.Petrol Engine
 2.Diesel Engine
1
Honda car with petrol engine gives 228 horsepower
```

Sample Input Output 2:

```
Select option
 1.Petrol Engine
 2.Diesel Engine
2
Suzuki car with diesel engine gives 23 horsepower
```