SE 3313 Software Design and Architecture

Abstract Factory Pattern

Lab 6

1 Context

The abstract factory pattern provides a way to encapsulate a group of individual factories that have a common theme without specifying their concrete classes. In normal usage, the client software creates a concrete implementation of the abstract factory and then uses the generic interface of the factory to create the concrete objects that are part of the theme. The client does not know (or care) which concrete objects it gets from each of these internal factories, since it uses only the generic interfaces of their products. This pattern separates the details of implementing a set of objects from their general usage and relies on object composition, as object creation is implemented in methods exposed in the factory interface.

2 Problem Description

Scenario: Imagine you are working on a system for manufacturing vehicles. There are two types of vehicles: Cars and Motorcycles. Each type has two variations: Electric and Gasoline.

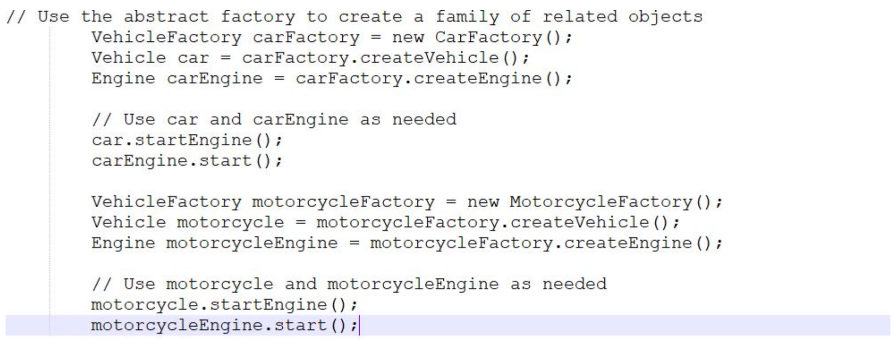
Task: Implement the abstract factory pattern to create families of related vehicle objects.

Requirements:
Define an interface Vehicle with methods like startEngine() and stopEngine().
Create concrete classes Car and Motorcycle that implement the Vehicle interface.
Define another interface Engine with methods like start() and stop().
Create concrete classes ElectricEngine and GasolineEngine that implement the Engine interface.
Create an abstract factory interface VehicleFactory with methods like createVehicle() and createEngine().
Implement two concrete factories: CarFactory and MotorcycleFactory that implement the VehicleFactory interface. Each factory should create the corresponding type of vehicle and engine.
Create a client class VehicleClient that uses the abstract factory to create a family of related objects (e.g., a Gasoline-powered car or an Electric motorcycle).

3 Measure of Success

You are expected to implement necessary classes for this example using Abstract Factory Pattern. You should implement menu and provide proper handling of user input and menu options.

4 Example MainClass(Client)



```
// Use the abstract factory to create a family of related objects
        VehicleFactory carFactory = new CarFactory();
        Vehicle car = carFactory.createVehicle();
        Engine carEngine = carFactory.createEngine();

        // Use car and carEngine as needed
        car.startEngine();
        carEngine.start();

        VehicleFactory motorcycleFactory = new MotorcycleFactory();
        Vehicle motorcycle = motorcycleFactory.createVehicle();
        Engine motorcycleEngine = motorcycleFactory.createEngine();

        // Use motorcycle and motorcycleEngine as needed
        motorcycle.startEngine();
        motorcycleEngine.start();
```

Figure 1: