Object Oriented Programming (JAVA)



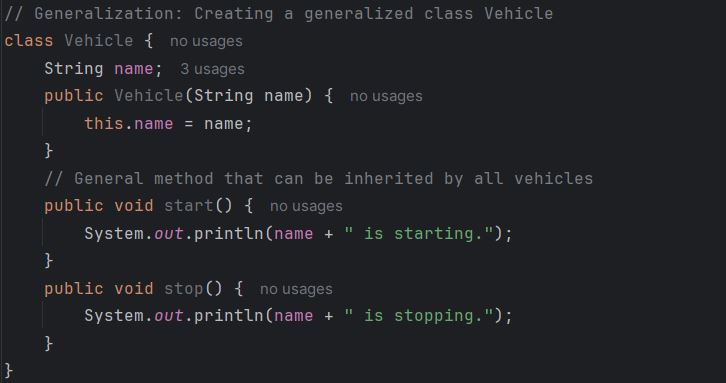
## Semester: Fall 2024

**Software Engineering**

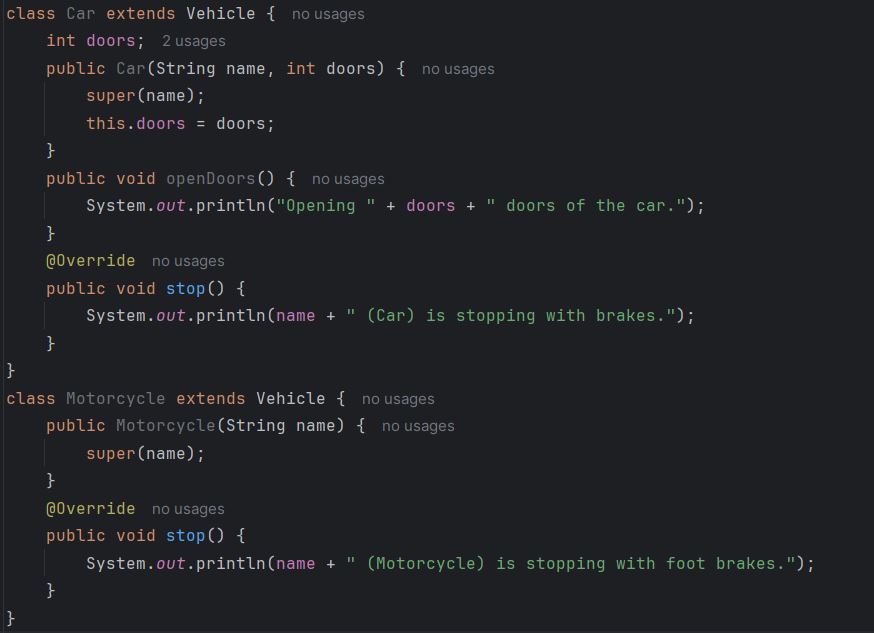
**Faculty of Information Technology UCP Lahore, Pakistan**

|  |  |
| --- | --- |
| **Week 13** | |
| **Topic** | **Generalization, Specialization, and Realization** |
| **Objective** | * Introduction to General class vs. specific class * Differentiating between Widening or up-casting vs. narrowing or down-casting |

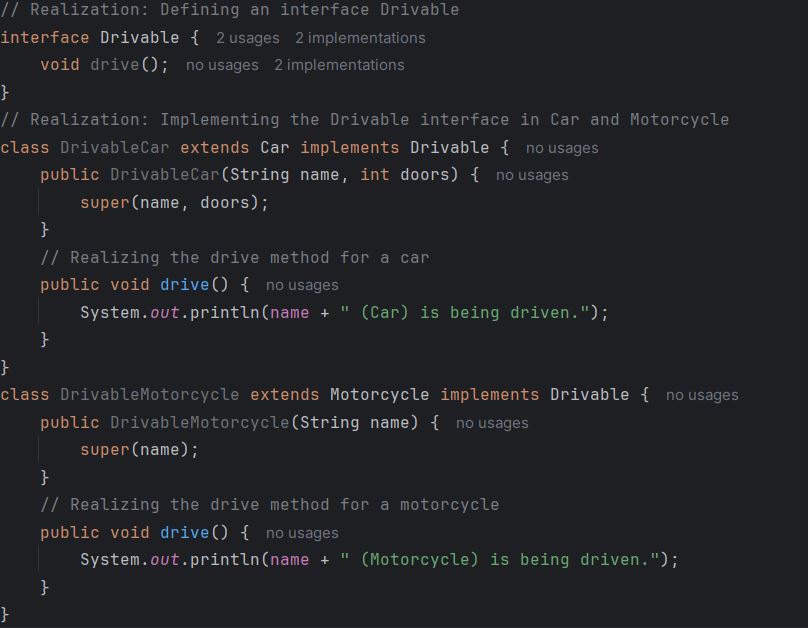
**Define the generalized class**



**Specialize the Class through Inheritance**

****

**Define an Interface and Implement Realization**



# Basic Practice Tasks:

# **Question 1: Car**

#### **1. General Class:** Car

Attributes:

* brand: String (e.g., Toyota, Ford)
* model: String (e.g., Camry, Mustang)
* year: int (e.g., 2023)
* fuelType: String (e.g., Petrol, Diesel, Electric)

Functions:

* displayDetails(): Prints car details.

#### **2.** Specialization class**:** Sedan **(extends** Car**)**

Additional Attributes:

* bootSpace: double (in liters, e.g., 500.0)
* luxuryLevel: String (e.g., High, Medium, Low)

Additional Functions:

* enableCruiseControl(): Enables cruise control.
* disableCruiseControl(): Disables cruise control.

#### **3.** Specialization**:** SUV **(extends** Car**)**

Additional Attributes:

* fourWheelDrive: boolean (e.g., true or false)
* groundClearance: double (in cm, e.g., 20.5)

Additional Functions:

* activateFourWheelDrive(): Activates 4WD.
* deactivateFourWheelDrive(): Deactivates 4WD.

#### **4.** Realization **:Interface:**

Methods:

* startEngine(): Starts the car engine.
* stopEngine(): Stops the car engine.

Here are the attributes and functions for each class and interface in the three questions:

### **Question 1: Car**

#### **1. General Class:** Car

Attributes:

* brand: String (e.g., Toyota, Ford)
* model: String (e.g., Camry, Mustang)
* year: int (e.g., 2023)
* fuelType: String (e.g., Petrol, Diesel, Electric)

Functions:

* displayDetails(): Prints car details.

#### **2. Subclass:** Sedan **(extends** Car**)**

Additional Attributes:

* bootSpace: double (in liters, e.g., 500.0)
* luxuryLevel: String (e.g., High, Medium, Low)

Additional Functions:

* enableCruiseControl(): Enables cruise control.
* disableCruiseControl(): Disables cruise control.

#### **3. Subclass:** SUV **(extends** Car**)**

Additional Attributes:

* fourWheelDrive: boolean (e.g., true or false)
* groundClearance: double (in cm, e.g., 20.5)

Additional Functions:

* activateFourWheelDrive(): Activates 4WD.
* deactivateFourWheelDrive(): Deactivates 4WD.

#### **4. Interface:** Drivable

Methods:

* startEngine(): Starts the car engine.
* stopEngine(): Stops the car engine.

# **Question 2: Mobile**

# **1. General Class:** Mobile

Attributes:

* brand: String (e.g., Apple, Samsung)
* model: String (e.g., iPhone 14, Galaxy S22)
* batteryCapacity: int (in mAh, e.g., 4000)

Functions:

* displaySpecifications(): Prints mobile specifications.

#### **2. Subclass:** Smartphone **(extends** Mobile**)**

Additional Attributes:

* os: String (e.g., Android, iOS)
* cameraResolution: double (in MP, e.g., 12.0)

Additional Functions:

* installApp(String appName): Installs an application.
* useTouchID(): Unlocks the phone using fingerprint or facial recognition.

#### **3. Subclass:** FeaturePhone **(extends** Mobile**)**

Additional Attributes:

* physicalKeyboard: boolean (e.g., true or false)
* dualSIM: boolean (e.g., true or false)

Additional Functions:

* playFM(): Starts the FM radio.
* sendSMS(String message): Sends an SMS.

#### **4. Interface:** Chargeable

Methods:

* plugIn(): Starts charging the mobile.
* unplug(): Stops charging the mobile.

### **Question 2: Books**

#### **1. General Class:** Book

Attributes:

* title: String (e.g., "The Great Gatsby")
* author: String (e.g., "F. Scott Fitzgerald")
* pages: int (e.g., 300)

Functions:

* readSynopsis(): Prints a brief summary of the book.

#### **2. Subclass:** FictionBook **(extends** Book**)**

Additional Attributes:

* genre: String (e.g., Romance, Mystery, Sci-Fi)
* isBestseller: boolean (e.g., true or false)

Additional Functions:

* recommendToFriends(): Recommends the book to a friend.

#### **3. Subclass:** TextBook **(extends** Book**)**

Additional Attributes:

* subject: String (e.g., Math, Physics)
* edition: int (e.g., 3 for the third edition)

Additional Functions:

* highlightText(String text): Highlights a given text.
* solveExercise(int exerciseNumber): Solves an exercise from the book.

#### **4. Interface:** Readable

Methods:

* open(): Opens the book.
* close(): Closes the book.

**Scenario Based Task**

**Task: Generalization and Specialization  
Instructions:**

* Create a parent class SmartDevice with methods turnOn and turnOff to represent common functionalities shared by all smart devices.
* Create two specialized subclasses: SmartLight and SmartFan. Each subclass should have its own additional method (setBrightness for SmartLight and setSpeed for SmartFan).
* In the main method, instantiate objects of both subclasses. Call the turnOn method for each device, then set the brightness for the light and the speed for the fan.
* Ensure that the program demonstrates the inheritance of common functionality and the addition of specific behavior in the subclasses.

|  |
| --- |
| **Sample Output:****Device is turned on.****Light brightness set to 80%****Device is turned on.****Fan speed set to level 3** |