CDAC Mumbai PG-DAC August 24

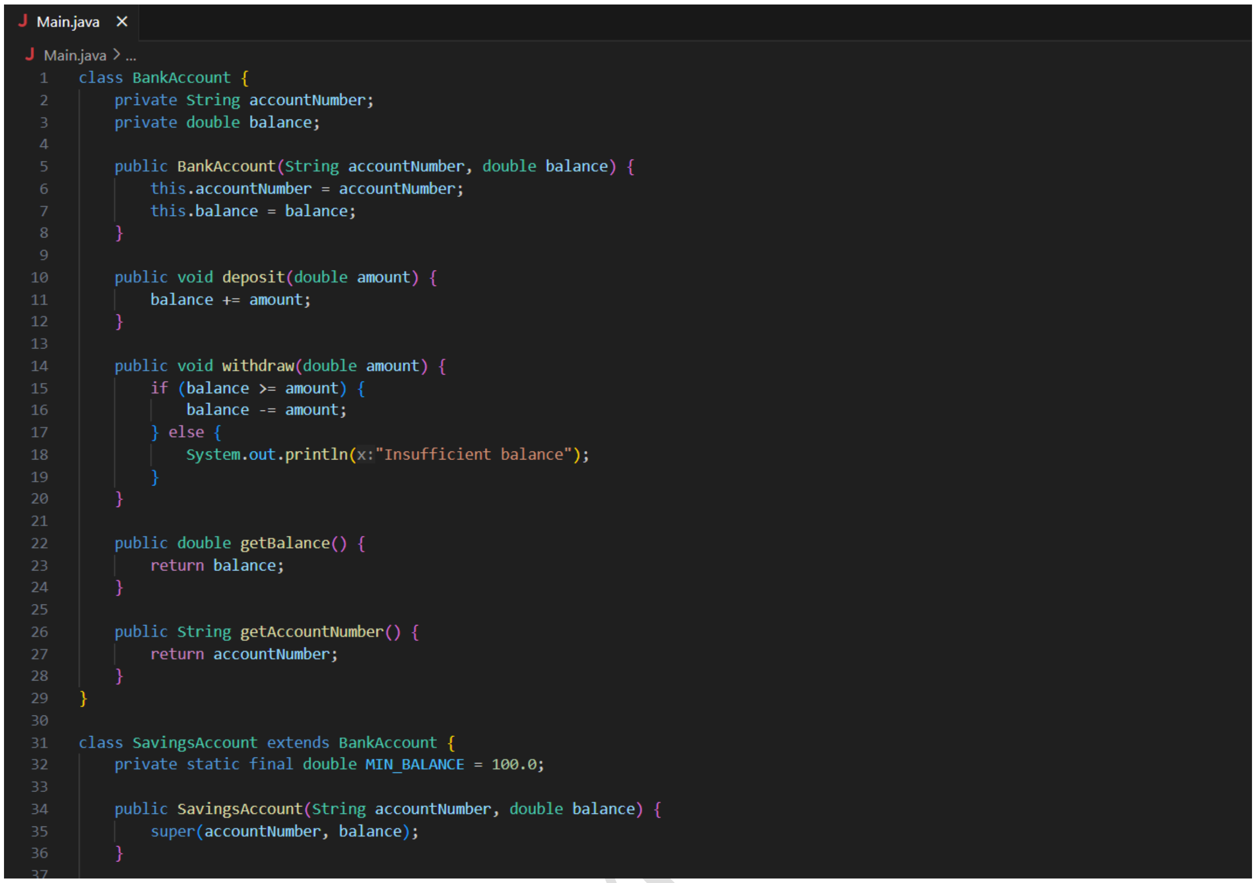
Assignment No- 5

1. Create a base class BankAccount with methods like deposit() and withdraw(). Derive a class SavingsAccount that overrides the withdraw() method to impose a limit on the withdrawal amount. Write a program that demonstrates the use of overridden methods and proper access modifiers & return the details.

   Ans:

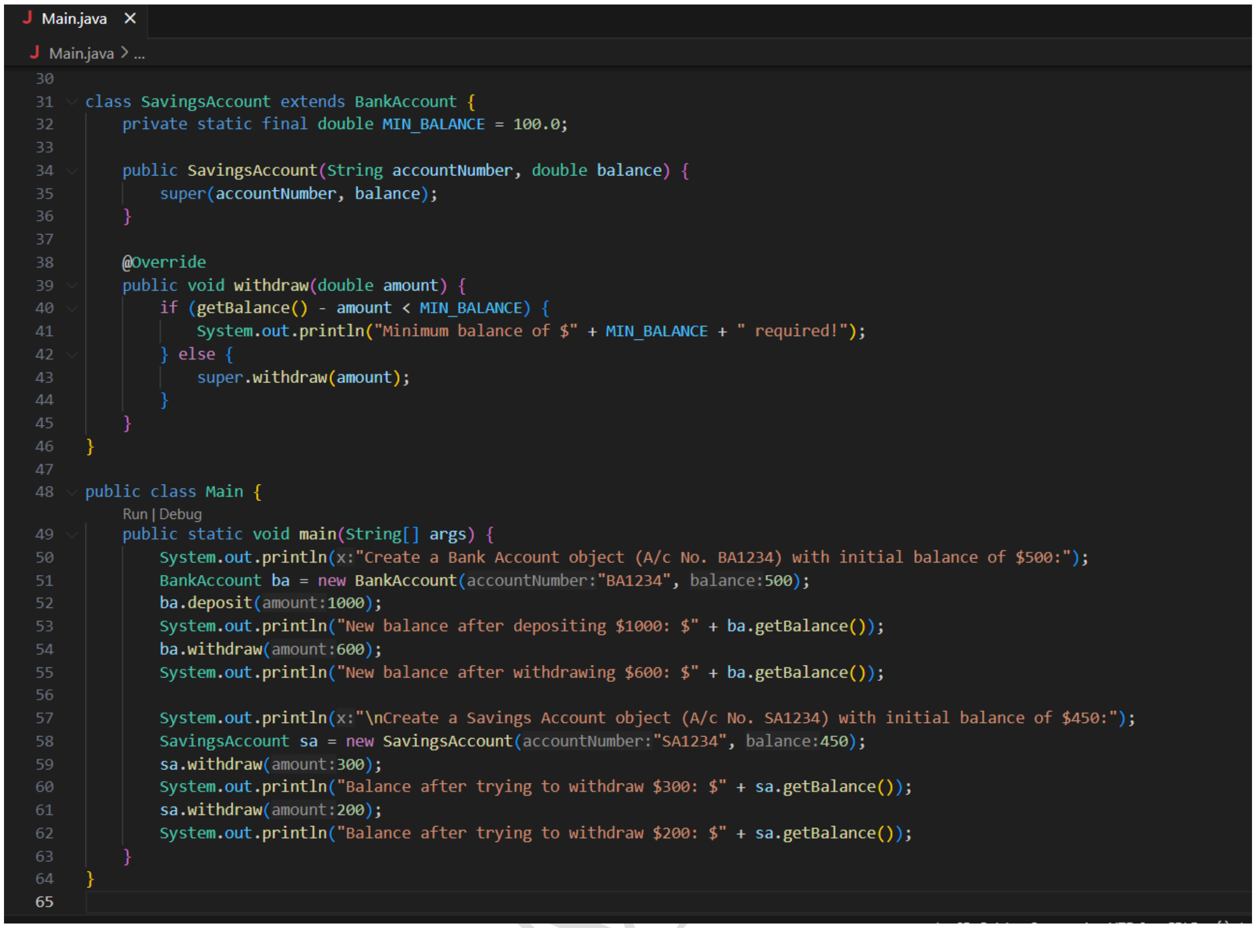
```
class BankAccount {
    private String accountNumber;
    private double balance;

    public BankAccount(String accountNumber, double balance) {
        this.accountNumber = accountNumber;
        this.balance = balance;
    }

    public void deposit(double amount) {
        balance += amount;
    }

    public void withdraw(double amount) {
        if (balance >= amount) {
            balance -= amount;
        } else {
            System.out.println("Insufficient balance");
        }
    }

    public double getBalance() {
        return balance;
    }

    public String getAccountNumber() {
        return accountNumber;
    }
}

class SavingsAccount extends BankAccount {
    private static final double MIN_BALANCE = 100.0;

    public SavingsAccount(String accountNumber, double balance) {
        super(accountNumber, balance);
    }
```

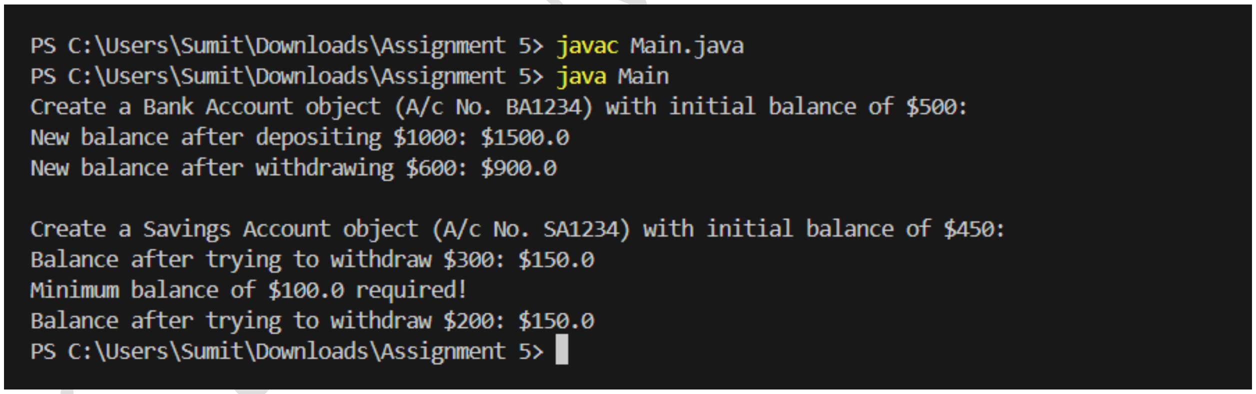
```
  @Override
  public void withdraw(double amount) {
    if (getBalance() - amount < MIN_BALANCE) {
      System.out.println("Minimum balance of $" + MIN_BALANCE + " required!");
    } else {
      super.withdraw(amount);
    }
  }
}

public class Main {
  public static void main(String[] args) {
    System.out.println("Create a Bank Account object (A/c No. BA1234) with initial balance of
$500:");
    BankAccount ba = new BankAccount("BA1234", 500);
    ba.deposit(1000);
    System.out.println("New balance after depositing $1000: $" + ba.getBalance());
    ba.withdraw(600);
    System.out.println("New balance after withdrawing $600: $" + ba.getBalance());

    System.out.println("\nCreate a Savings Account object (A/c No. SA1234) with initial balance of
$450:");
    SavingsAccount sa = new SavingsAccount("SA1234", 450);
    sa.withdraw(300);
    System.out.println("Balance after trying to withdraw $300: $" + sa.getBalance());
    sa.withdraw(200);
    System.out.println("Balance after trying to withdraw $200: $" + sa.getBalance());
  }
}
```


Main.java ×

Main.java > ...

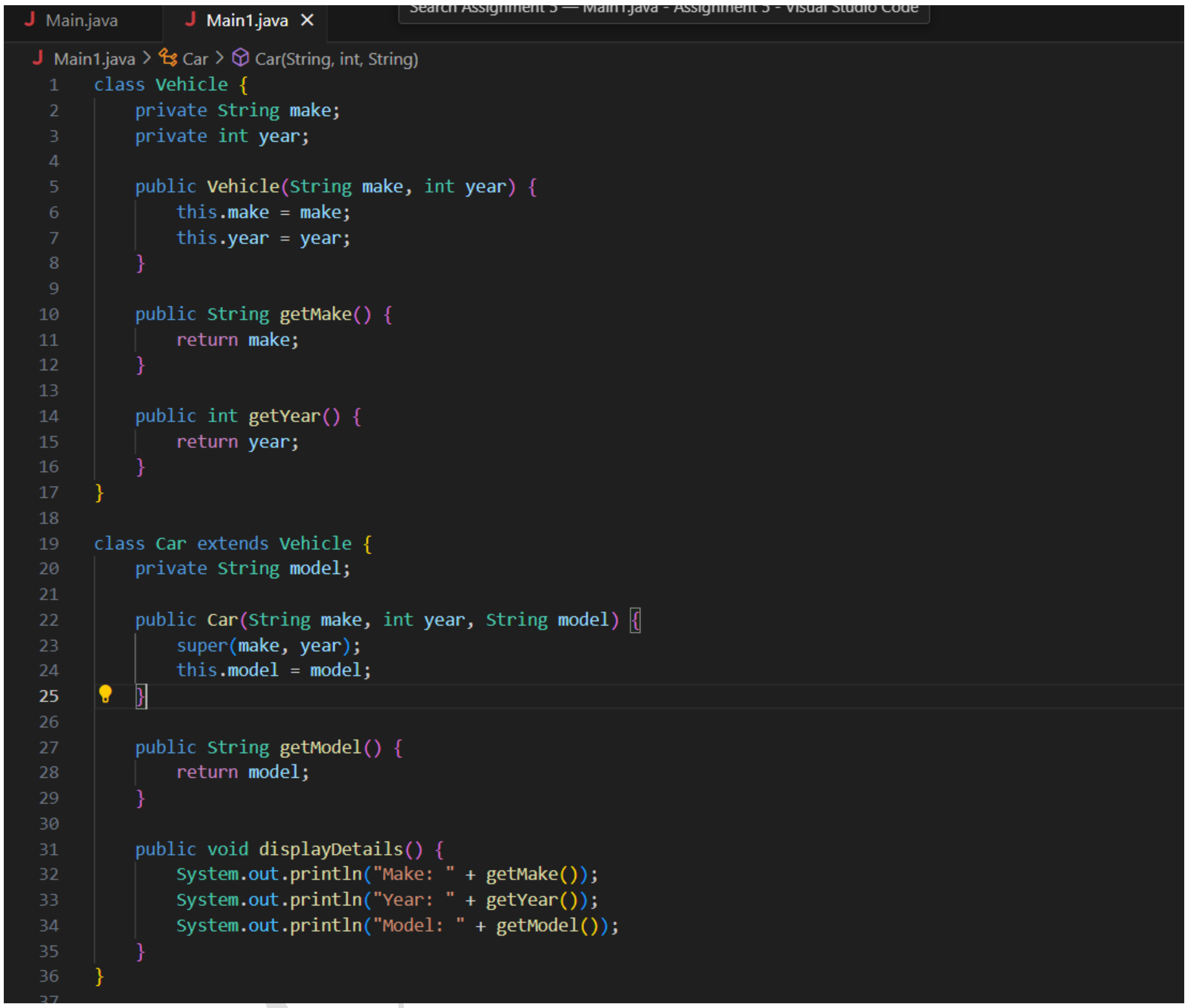
```java
class BankAccount {
    private String accountNumber;
    private double balance;

    public BankAccount(String accountNumber, double balance) {
        this.accountNumber = accountNumber;
        this.balance = balance;
    }

    public void deposit(double amount) {
        balance += amount;
    }

    public void withdraw(double amount) {
        if (balance >= amount) {
            balance -= amount;
        } else {
            System.out.println(x:"Insufficient balance");
        }
    }

    public double getBalance() {
        return balance;
    }

    public String getAccountNumber() {
        return accountNumber;
    }
}

class SavingsAccount extends BankAccount {
    private static final double MIN_BALANCE = 100.0;

    public SavingsAccount(String accountNumber, double balance) {
        super(accountNumber, balance);
    }
```


```
class SavingsAccount extends BankAccount {
    private static final double MIN_BALANCE = 100.0;

    public SavingsAccount(String accountNumber, double balance) {
        super(accountNumber, balance);
    }

    @Override
    public void withdraw(double amount) {
        if (getBalance() - amount < MIN_BALANCE) {
            System.out.println("Minimum balance of $" + MIN_BALANCE + " required!");
        } else {
            super.withdraw(amount);
        }
    }
}

public class Main {
    public static void main(String[] args) {
        System.out.println(x:"Create a Bank Account object (A/c No. BA1234) with initial balance of $500:");
        BankAccount ba = new BankAccount(accountNumber:"BA1234", balance:500);
        ba.deposit(amount:1000);
        System.out.println("New balance after depositing $1000: $" + ba.getBalance());
        ba.withdraw(amount:600);
        System.out.println("New balance after withdrawing $600: $" + ba.getBalance());

        System.out.println(x:"\nCreate a Savings Account object (A/c No. SA1234) with initial balance of $450:");
        SavingsAccount sa = new SavingsAccount(accountNumber:"SA1234", balance:450);
        sa.withdraw(amount:300);
        System.out.println("Balance after trying to withdraw $300: $" + sa.getBalance());
        sa.withdraw(amount:200);
        System.out.println("Balance after trying to withdraw $200: $" + sa.getBalance());
    }
}
```


```
PS C:\Users\Sumit\Downloads\Assignment 5> javac Main.java
PS C:\Users\Sumit\Downloads\Assignment 5> java Main
Create a Bank Account object (A/c No. BA1234) with initial balance of $500:
New balance after depositing $1000: $1500.0
New balance after withdrawing $600: $900.0

Create a Savings Account object (A/c No. SA1234) with initial balance of $450:
Balance after trying to withdraw $300: $150.0
Minimum balance of $100.0 required!
Balance after trying to withdraw $200: $150.0
PS C:\Users\Sumit\Downloads\Assignment 5> 
```

2. Create a base class Vehicle with attributes like make and year. Provide a constructor in Vehicle to initialize these attributes. Derive a class Car that has an additional attribute model and write a constructor that initializes make, year, and model. Write a program to create a Car object and display its details.
   Ans:

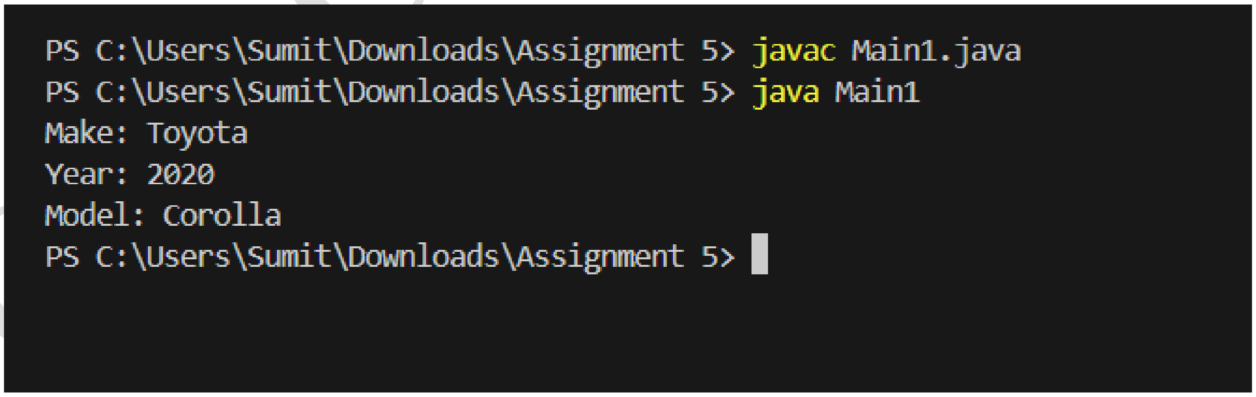
```
class Vehicle {
    private String make;
    private int year;
    public Vehicle(String make, int year) {
        this.make = make;
        this.year = year;
    }

    public String getMake() {
        return make;
    }

    public int getYear() {
        return year;
    }
}

class Car extends Vehicle {
    private String model;

    public Car(String make, int year, String model) {
        super(make, year);
        this.model = model;
    }

    public String getModel() {
        return model;
    }

    public void displayDetails() {
        System.out.println("Make: " + getMake());
        System.out.println("Year: " + getYear());
        System.out.println("Model: " + getModel());
    }
}

public class Main1 {
    public static void main(String[] args) {
        Car car = new Car("Toyota", 2020, "Corolla");
        car.displayDetails();
    }
}
```



Main.java | Main1.java ×

Main1.java > Car > Car(String, int, String)

```java
class Vehicle {
    private String make;
    private int year;

    public Vehicle(String make, int year) {
        this.make = make;
        this.year = year;
    }

    public String getMake() {
        return make;
    }

    public int getYear() {
        return year;
    }
}

class Car extends Vehicle {
    private String model;

    public Car(String make, int year, String model) {
        super(make, year);
        this.model = model;
    }

    public String getModel() {
        return model;
    }

    public void displayDetails() {
        System.out.println("Make: " + getMake());
        System.out.println("Year: " + getYear());
        System.out.println("Model: " + getModel());
    }
}
```

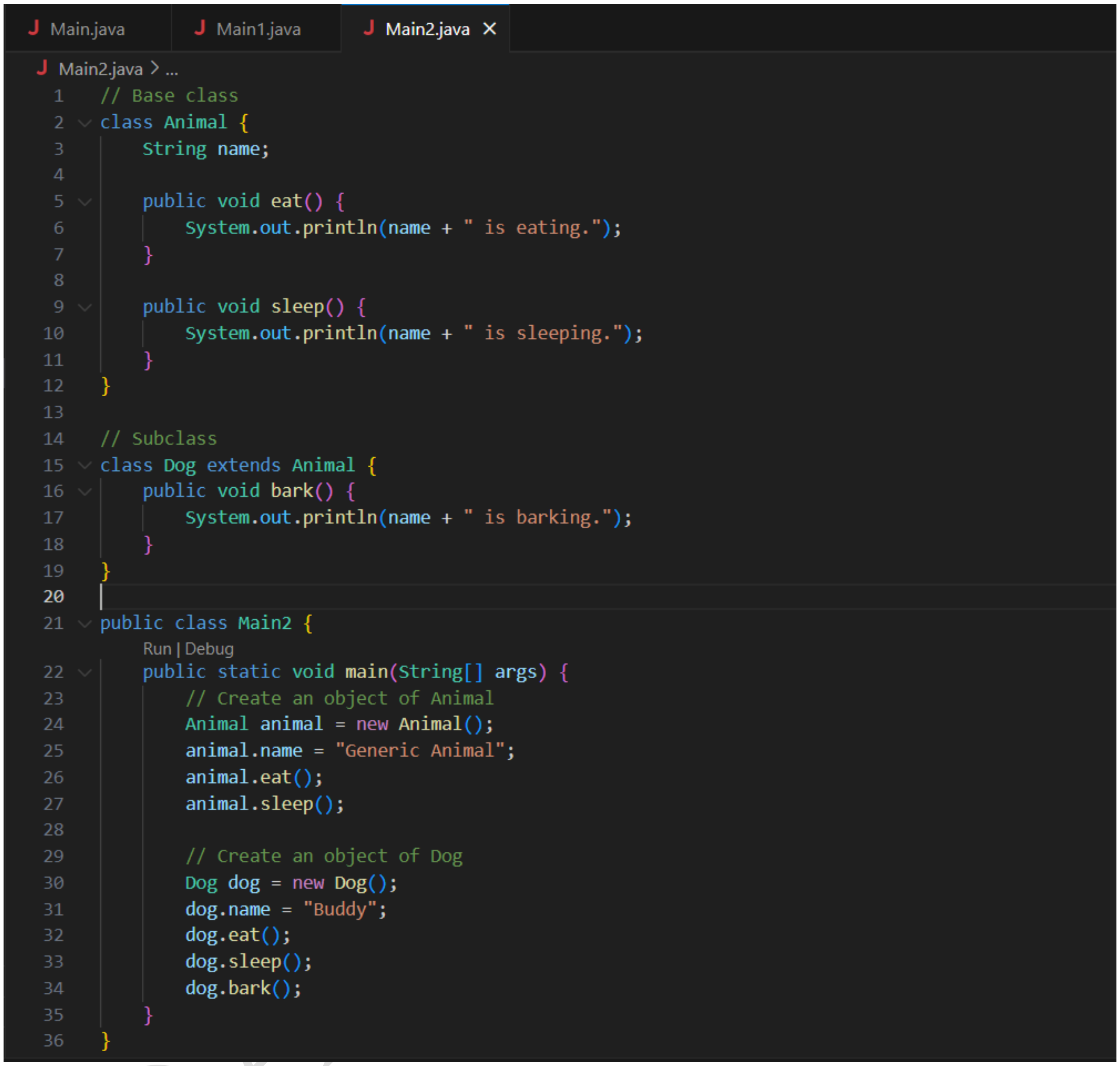


```
PS C:\Users\Sumit\Downloads\Assignment 5> javac Main1.java
PS C:\Users\Sumit\Downloads\Assignment 5> java Main1
Make: Toyota
Year: 2020
Model: Corolla
PS C:\Users\Sumit\Downloads\Assignment 5>
```

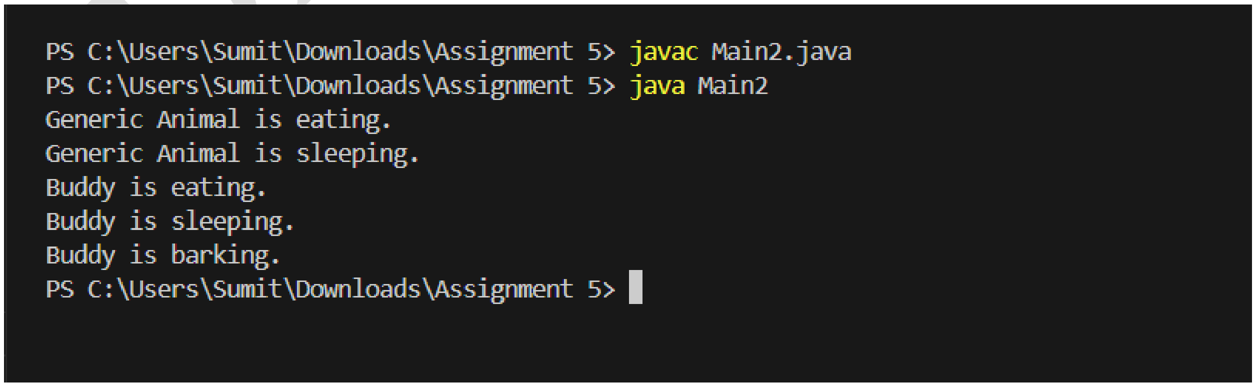
3. Create a base class Animal with attributes like name, and methods like eat() and sleep(). Create a subclass Dog that inherits from Animal and has an additional method bark(). Write a program to demonstrate the use of inheritance by creating objects of Animal and Dog and calling their methods.

Ans:

```
    // Base class
class Animal {
   String name;

   public void eat() {
      System.out.println(name + " is eating.");
   }

   public void sleep() {
      System.out.println(name + " is sleeping.");
   }
}

// Subclass
class Dog extends Animal {
   public void bark() {
      System.out.println(name + " is barking.");
   }
}

public class Main2 {
   public static void main(String[] args) {
      // Create an object of Animal
      Animal animal = new Animal();
      animal.name = "Generic Animal";
      animal.eat();
      animal.sleep();

      // Create an object of Dog
      Dog dog = new Dog();
      dog.name = "Buddy";
      dog.eat();
      dog.sleep();
      dog.bark();
   }
}
```



```java
// Base class
class Animal {
    String name;

    public void eat() {
        System.out.println(name + " is eating.");
    }

    public void sleep() {
        System.out.println(name + " is sleeping.");
    }
}

// Subclass
class Dog extends Animal {
    public void bark() {
        System.out.println(name + " is barking.");
    }
}

public class Main2 {
    public static void main(String[] args) {
        // Create an object of Animal
        Animal animal = new Animal();
        animal.name = "Generic Animal";
        animal.eat();
        animal.sleep();

        // Create an object of Dog
        Dog dog = new Dog();
        dog.name = "Buddy";
        dog.eat();
        dog.sleep();
        dog.bark();
    }
}
```

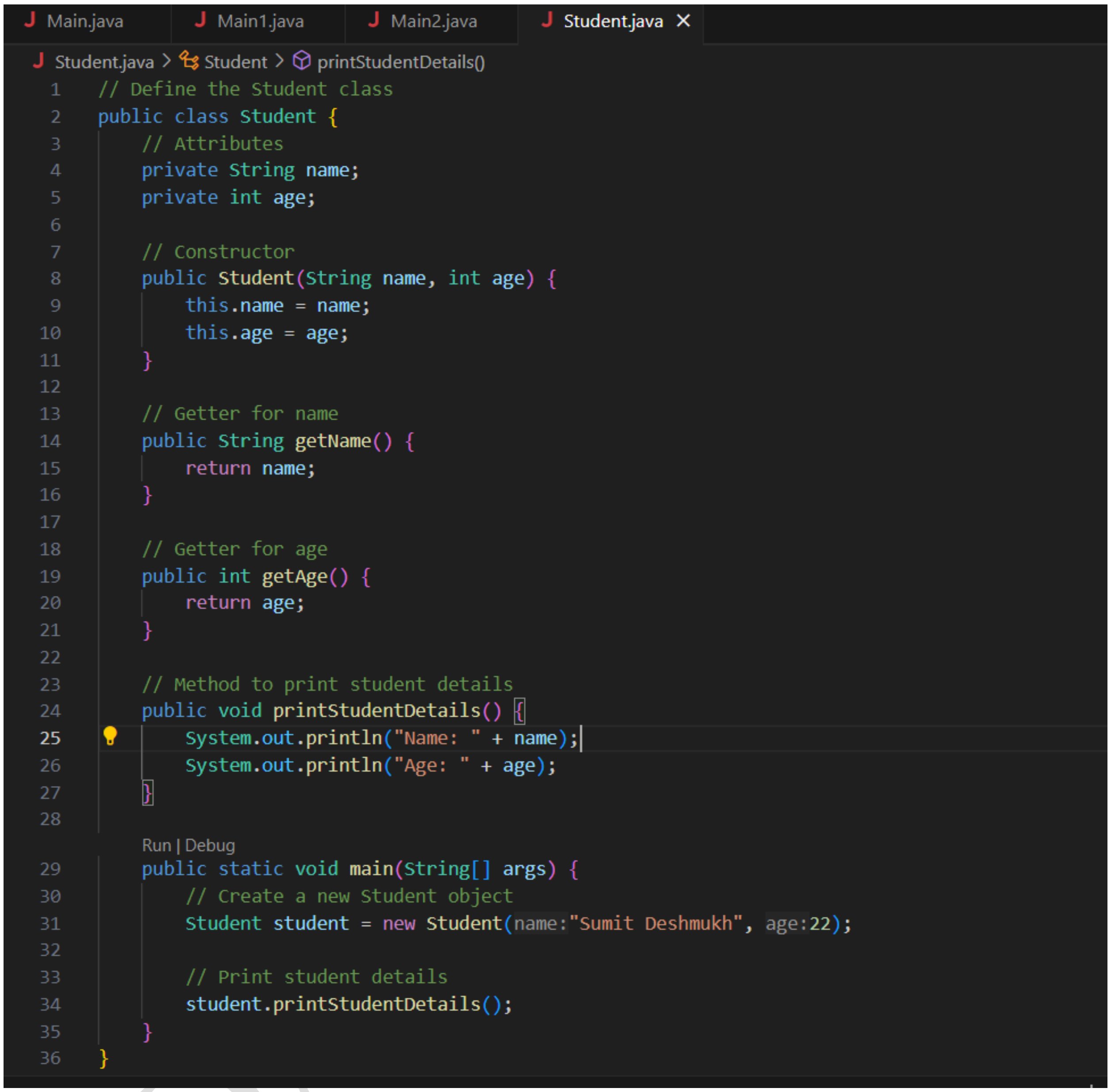


```
PS C:\Users\Sumit\Downloads\Assignment 5> javac Main2.java
PS C:\Users\Sumit\Downloads\Assignment 5> java Main2
Generic Animal is eating.
Generic Animal is sleeping.
Buddy is eating.
Buddy is sleeping.
Buddy is barking.
PS C:\Users\Sumit\Downloads\Assignment 5>
```

4. Build a class Student which contains details about the Student and compile and run its instance.

Ans:

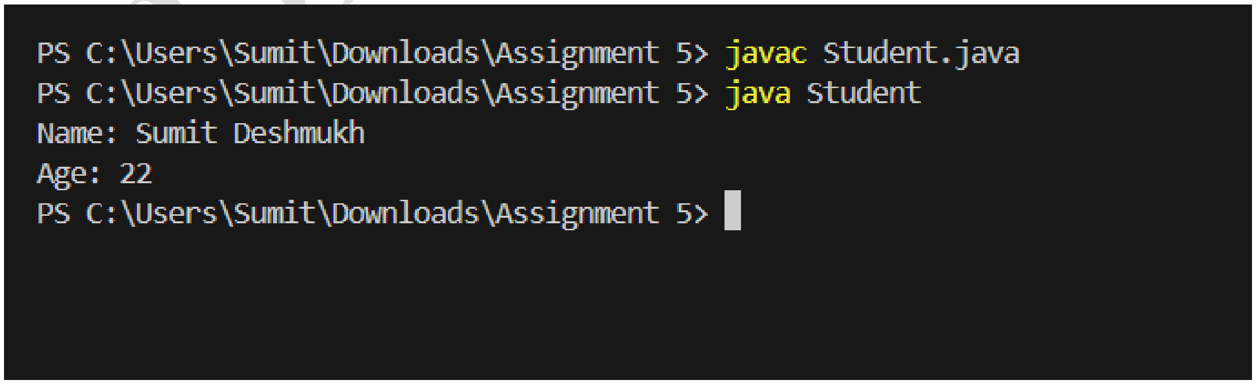
```
// Define the Student class
public class Student {
   // Attributes
   private String name;
   private int age;

   // Constructor
   public Student(String name, int age) {
      this.name = name;
      this.age = age;
   }

   // Getter for name
   public String getName() {
      return name;
   }

   // Getter for age
   public int getAge() {
      return age;
   }

   // Method to print student details
   public void printStudentDetails() {
      System.out.println("Name: " + name);
      System.out.println("Age: " + age);
   }

   public static void main(String[] args) {
      // Create a new Student object
      Student student = new Student("Sumit Deshmukh", 22);

      // Print student details
      student.printStudentDetails();
   }
}
```



Main.java | Main1.java | Main2.java | Student.java

Student.java > Student > printStudentDetails()

```java
// Define the Student class
public class Student {
    // Attributes
    private String name;
    private int age;

    // Constructor
    public Student(String name, int age) {
        this.name = name;
        this.age = age;
    }

    // Getter for name
    public String getName() {
        return name;
    }

    // Getter for age
    public int getAge() {
        return age;
    }

    // Method to print student details
    public void printStudentDetails() {
        System.out.println("Name: " + name);
        System.out.println("Age: " + age);
    }

    Run | Debug
    public static void main(String[] args) {
        // Create a new Student object
        Student student = new Student(name:"Sumit Deshmukh", age:22);

        // Print student details
        student.printStudentDetails();
    }
}
```

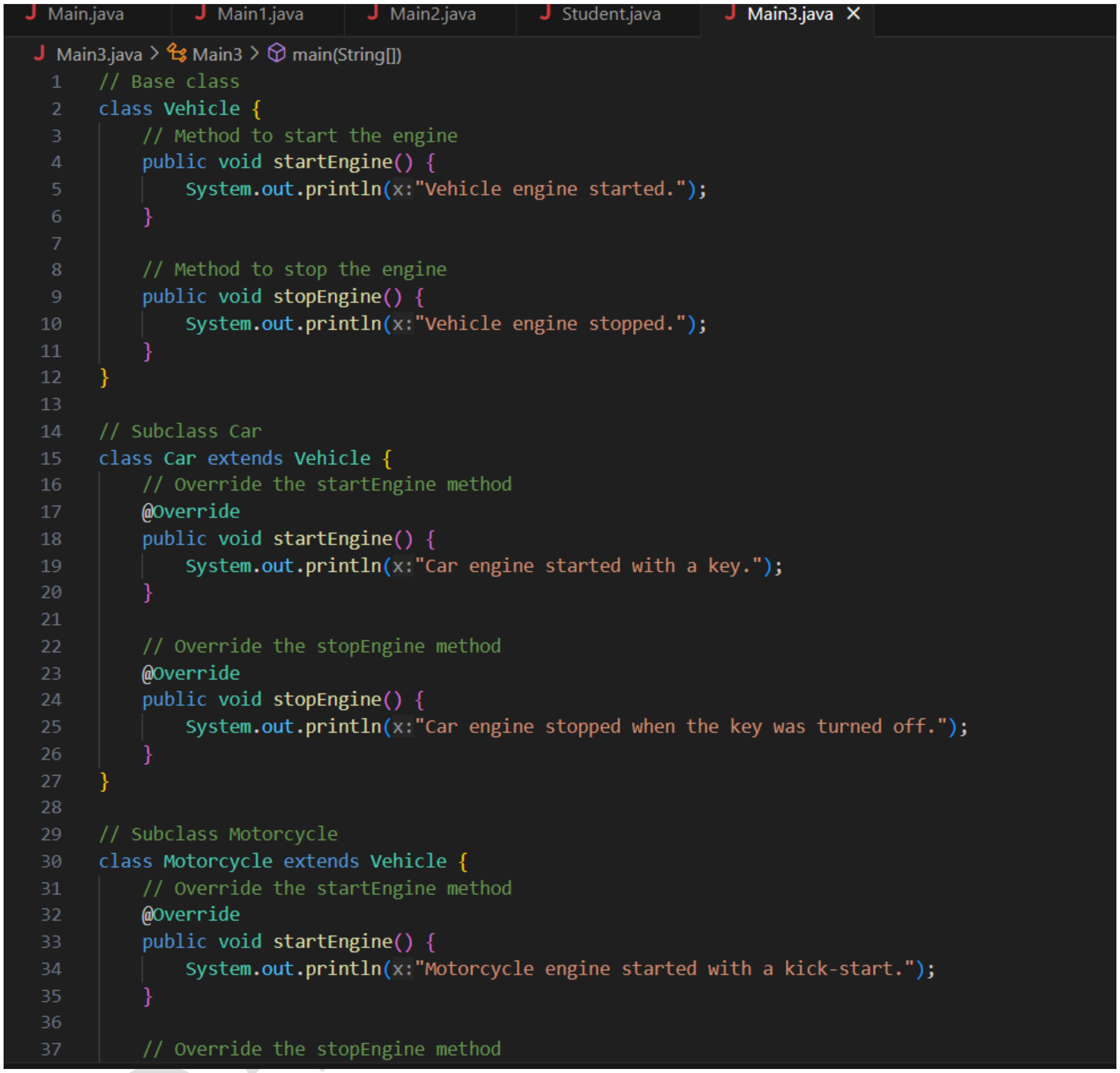


```
PS C:\Users\Sumit\Downloads\Assignment 5> javac Student.java
PS C:\Users\Sumit\Downloads\Assignment 5> java Student
Name: Sumit Deshmukh
Age: 22
PS C:\Users\Sumit\Downloads\Assignment 5>
```

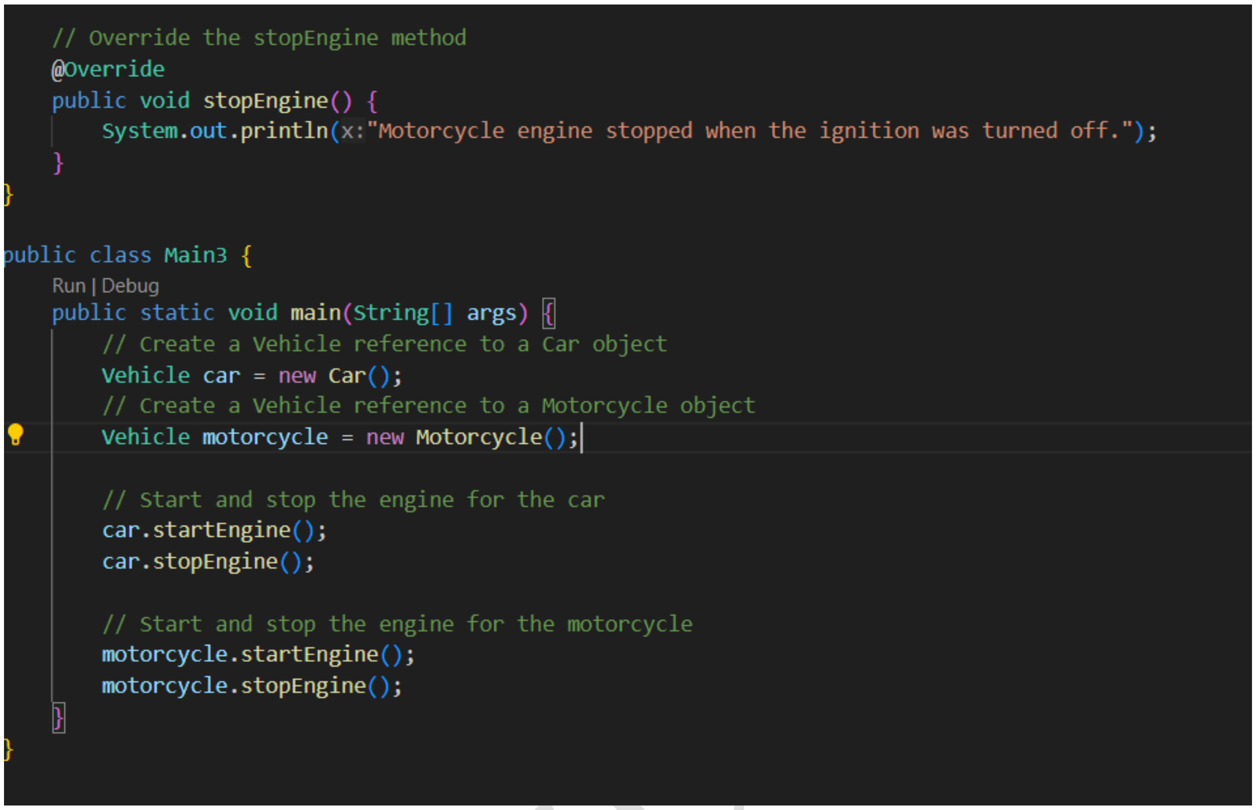
5. Write a Java program to create a base class Vehicle with methods startEngine() and stopEngine(). Create two subclasses Car and Motorcycle. Override the startEngine() and stopEngine() methods in each subclass to start and stop the engines differently.

Ans:

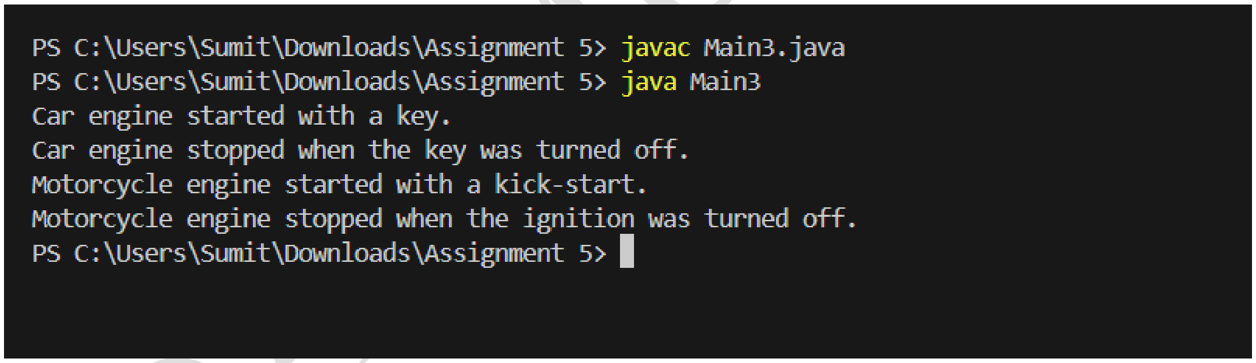
```
// Base class
class Vehicle {
   // Method to start the engine
   public void startEngine() {
      System.out.println("Vehicle engine started.");
   }

   // Method to stop the engine
   public void stopEngine() {
      System.out.println("Vehicle engine stopped.");
   }
}

// Subclass Car
class Car extends Vehicle {
   // Override the startEngine method
   @Override
   public void startEngine() {
      System.out.println("Car engine started with a key.");
   }

   // Override the stopEngine method
   @Override
   public void stopEngine() {
      System.out.println("Car engine stopped when the key was turned off.");
   }
}

// Subclass Motorcycle
class Motorcycle extends Vehicle {
   // Override the startEngine method
   @Override
   public void startEngine() {
      System.out.println("Motorcycle engine started with a kick-start.");
   }

   // Override the stopEngine method
   @Override
   public void stopEngine() {
      System.out.println("Motorcycle engine stopped when the ignition was turned off.");
   }
}

public class Main3 {
    public static void main(String[] args) {
        // Create a Vehicle reference to a Car object
        Vehicle car = new Car();
        // Create a Vehicle reference to a Motorcycle object
        Vehicle motorcycle = new Motorcycle();

        // Start and stop the engine for the car
        car.startEngine();
        car.stopEngine();

        // Start and stop the engine for the motorcycle
        motorcycle.startEngine();
        motorcycle.stopEngine();
    }
}
```



```java
// Base class
class Vehicle {
    // Method to start the engine
    public void startEngine() {
        System.out.println(x:"Vehicle engine started.");
    }

    // Method to stop the engine
    public void stopEngine() {
        System.out.println(x:"Vehicle engine stopped.");
    }
}

// Subclass Car
class Car extends Vehicle {
    // Override the startEngine method
    @Override
    public void startEngine() {
        System.out.println(x:"Car engine started with a key.");
    }

    // Override the stopEngine method
    @Override
    public void stopEngine() {
        System.out.println(x:"Car engine stopped when the key was turned off.");
    }
}

// Subclass Motorcycle
class Motorcycle extends Vehicle {
    // Override the startEngine method
    @Override
    public void startEngine() {
        System.out.println(x:"Motorcycle engine started with a kick-start.");
    }

    // Override the stopEngine method
```



```java
    // Override the stopEngine method
    @Override
    public void stopEngine() {
        System.out.println(x:"Motorcycle engine stopped when the ignition was turned off.");
    }
}

public class Main3 {
    Run | Debug
    public static void main(String[] args) {
        // Create a Vehicle reference to a Car object
        Vehicle car = new Car();
        // Create a Vehicle reference to a Motorcycle object
        Vehicle motorcycle = new Motorcycle();

        // Start and stop the engine for the car
        car.startEngine();
        car.stopEngine();

        // Start and stop the engine for the motorcycle
        motorcycle.startEngine();
        motorcycle.stopEngine();
    }
}
```



```
PS C:\Users\Sumit\Downloads\Assignment 5> javac Main3.java
PS C:\Users\Sumit\Downloads\Assignment 5> java Main3
Car engine started with a key.
Car engine stopped when the key was turned off.
Motorcycle engine started with a kick-start.
Motorcycle engine stopped when the ignition was turned off.
PS C:\Users\Sumit\Downloads\Assignment 5>
```