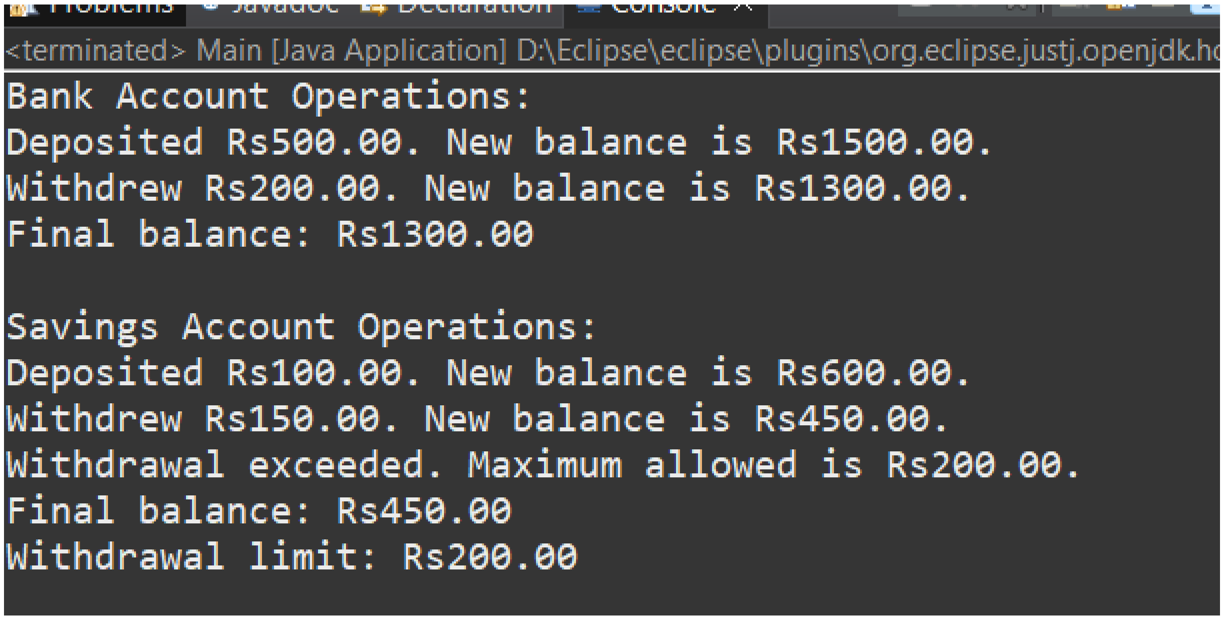
Create a base class BankAccount with methods like deposit() and withdraw(). Derive a class SavingsAccount that overrides the withdraw() method to impose a limit on the withdrawal amount. Write a program that demonstrates the use of overridden methods and proper access modifiers & return the details

```
package ques1;
public class BankAccount {
        private String accountNumber;
  private double balance;
  public BankAccount(String accountNumber, double initialBalance) {
     this.accountNumber = accountNumber;
     this.balance = initialBalance;
  }
  public void deposit(double amount) {
     if (amount > 0) {
        balance += amount;
        System.out.printf("Deposited Rs%.2f. New balance is Rs%.2f.%n", amount,
balance);
     } else {
        System.out.println("Deposit amount must be positive.");
     }
  }
  public void withdraw(double amount) {
     if (amount > 0 && amount <= balance) {
        balance -= amount;
        System.out.printf("Withdrew Rs%.2f. New balance is Rs%.2f.%n", amount,
balance);
     } else {
        System.out.println("Insufficient funds or invalid amount.");
     }
  }
  public double getBalance() {
     return balance;
  }
  public String getAccountNumber() {
     return accountNumber;
  }
}
package ques1;
public class SavingsAccount extends BankAccount {
        private double withdrawalLimit;
  public SavingsAccount(String accountNumber, double initialBalance, double
withdrawalLimit) {
     super(accountNumber, initialBalance);
     this.withdrawalLimit = withdrawalLimit;
  }
  @Override
  public void withdraw(double amount) {
     if (amount > 0 && amount <= getBalance()) {
        if (amount <= withdrawalLimit) {
           super.withdraw(amount);  // Call the parent
        } else {
           System.out.printf("Withdrawal exceeded. Maximum allowed is
Rs%.2f.%n", withdrawalLimit);
        }
     } else {
        System.out.println("Insufficient funds or invalid amount.");
     }
  }
  public double getWithdrawalLimit() {
     return withdrawalLimit;
  }
}
package ques1;
public class Main {

        public static void main(String[] args) {


     // Create a bank account


     BankAccount account = new BankAccount("123456789", 1000);
     SavingsAccount savings = new SavingsAccount("987654321", 500, 200);
     // Test deposit and withdrawal


     System.out.println("Bank Account Operations:");
     account.deposit(500);
```

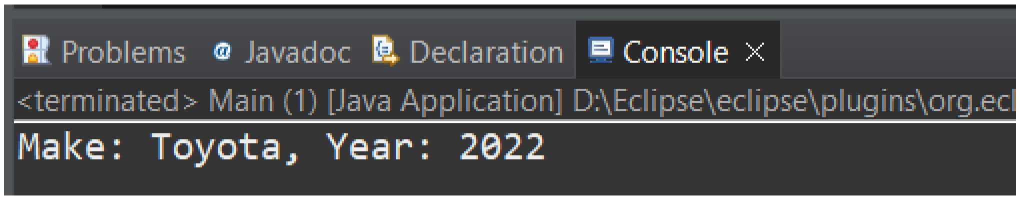
```java
        account.withdraw(200);
        System.out.printf("Final balance: Rs%.2f%n%n", account.getBalance());
        // Test deposit and withdrawal on the SavingsAccount


        System.out.println("Savings Account Operations:");
        savings.deposit(100);
        savings.withdraw(150);
        savings.withdraw(250);  // Should exceed withdrawal limit
        System.out.printf("Final balance: Rs%.2f%n", savings.getBalance());
        System.out.printf("Withdrawal limit: Rs%.2f%n",
savings.getWithdrawalLimit());
    }
}
```



```
<terminated> Main [Java Application] D:\Eclipse\eclipse\plugins\org.eclipse.justj.openjdk.hc
Bank Account Operations:
Deposited Rs500.00. New balance is Rs1500.00.
Withdrew Rs200.00. New balance is Rs1300.00.
Final balance: Rs1300.00

Savings Account Operations:
Deposited Rs100.00. New balance is Rs600.00.
Withdrew Rs150.00. New balance is Rs450.00.
Withdrawal exceeded. Maximum allowed is Rs200.00.
Final balance: Rs450.00
Withdrawal limit: Rs200.00
```

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.

```
package com.org.ques1;
//Base class
public class ques1 {
private String make;
private int year;
// Constructor
public ques1(String make, int year) {
    this.make = make;
    this.year = year;
}
// Method to display details
public String displayDetails() {
    return "Make: " + make + ", Year: " + year;
}


}
package com.org.ques1;
//Derived class
public class ques2 extends ques1 {
private String model;
// Constructor
public ques2(String make, int year, String model) {
    // Initialize the base class attributes
    super(make, year);
    // Initialize the additional attribute
    this.model = model;
}
// Overridden method to display details
@Override
public String displayDetails() {
    // Call the base class method and extend it with model
    return super.displayDetails() + ", Model: " + model;
}
}
package com.org.ques1;
public class Main {
  public static void main(String[] args) {
    // Create a Car object
    ques1 myCar = new ques1("Toyota", 2022);

    // Display the details of the Car object
    System.out.println(myCar.displayDetails());
  }
}
```

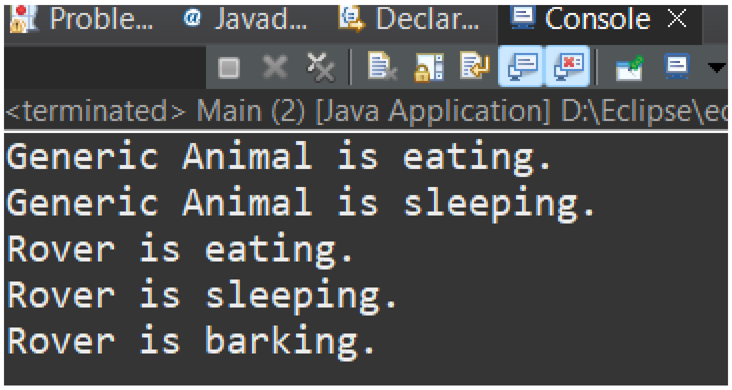


```
<terminated> Main (1) [Java Application] D:\Eclipse\eclipse\plugins\org.ecl
Make: Toyota, Year: 2022
```

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.

```
package ques3;
public class Animal {

        private String name;
  // Constructor for Animal
  public Animal(String name) {
    this.name = name;
  }
  // Getter for name
  public String getName() {
    return name;
  }
  // Method to simulate eating
  public void eat() {
    System.out.println(name + " is eating.");
  }
  // Method to simulate sleeping
  public void sleep() {
    System.out.println(name + " is sleeping.");
  }
```

```java
}
package ques3;
class Dog extends Animal {
  // Constructor for Dog
  public Dog(String name) {
     super(name); // Call the constructor of the base class Animal
  }
  // Method to simulate barking
  public void bark() {
     System.out.println(getName() + " is barking.");
  }
}
package ques3;
public class Main {
        public static void main(String[] args) {
                Animal myAnimal = new Animal("Generic Animal");
     myAnimal.eat();
     myAnimal.sleep();
     Dog myDog = new Dog("Rover");
     myDog.eat();  // Inherited method
     myDog.sleep(); // Inherited method
     myDog.bark();  // Dog-specific method
        }
}
```

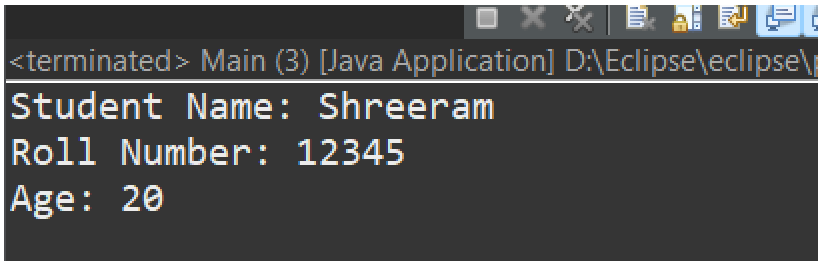


```
<terminated> Main (2) [Java Application] D:\Eclipse\ec
Generic Animal is eating.
Generic Animal is sleeping.
Rover is eating.
Rover is sleeping.
Rover is barking.
```

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

```java
package ques4;
public class Student {
        private String name;
  private int rollNumber;
  private int age;
  // Constructor to initialize the Student object
  public Student(String name, int rollNumber, int age) {
     this.name = name;
     this.rollNumber = rollNumber;
     this.age = age;
  }
  // Getter for name
  public String getName() {
     return name;
  }
  // Getter for rollNumber
  public int getRollNumber() {
     return rollNumber;
  }
  // Getter for age
  public int getAge() {
     return age;
  }
  // Method to display student details
  public void displayDetails() {
     System.out.println("Student Name: " + name);
     System.out.println("Roll Number: " + rollNumber);
     System.out.println("Age: " + age);
  }
}
package ques4;
public class Main {

        public static void main(String[] args) {
     // Create a Student object
     Student student1 = new Student("Shreeram", 12345, 20);

     // Display the details of the student
```

```java
    student1.displayDetails();
  }
}
```



```
<terminated> Main (3) [Java Application] D:\Eclipse\eclipse\
Student Name: Shreeram
Roll Number: 12345
Age: 20
```

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.

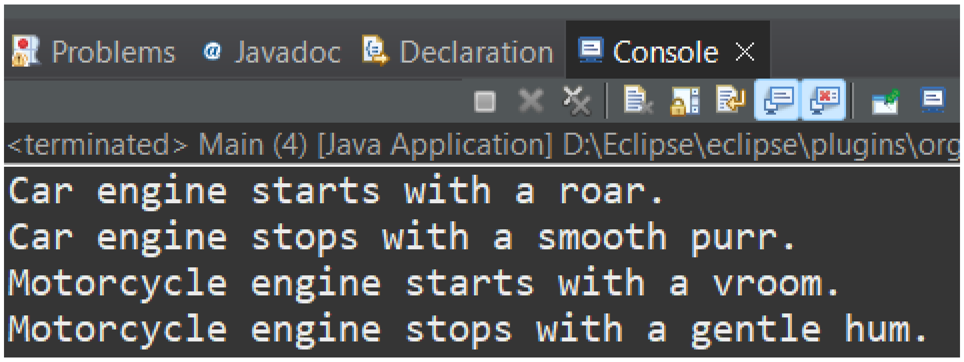
```java
package ques5;
abstract class Vehicle {
	// Method to start the engine
	public abstract void startEngine();
	// Method to stop the engine
	public abstract void stopEngine();
}
```

```java
package ques5;
class Car extends Vehicle {
	@Override
	public void startEngine() {
		System.out.println("Car engine starts with a roar.");
	}
	@Override
	public void stopEngine() {
		System.out.println("Car engine stops with a smooth purr.");
	}
}
```

```java
package ques5;
class Motorcycle extends Vehicle {
	// Override startEngine method for Motorcycle
```

```java
    @Override
    public void startEngine() {
        System.out.println("Motorcycle engine starts with a vroom.");
    }
    // Override stopEngine method for Motorcycle
    @Override
    public void stopEngine() {
        System.out.println("Motorcycle engine stops with a gentle hum.");
    }
}
```

```java
package ques5;
public class Main {
    public static void main(String[] args) {
        Vehicle myCar = new Car();
        myCar.startEngine();
        myCar.stopEngine();
        // Create a Motorcycle object
        Vehicle myMotorcycle = new Motorcycle();
        myMotorcycle.startEngine();
        myMotorcycle.stopEngine();
    }
}
```



```
<terminated> Main (4) [Java Application] D:\Eclipse\eclipse\plugins\org
Car engine starts with a roar.
Car engine stops with a smooth purr.
Motorcycle engine starts with a vroom.
Motorcycle engine stops with a gentle hum.
```