|  |
| --- |
| **Question 1: Simple Inheritance Example**  **Problem Statement:** Create a Java program that demonstrates simple inheritance. Create a base class Animal with properties like name and a method makeSound(). Then, create a derived class Dog that extends Animal and overrides the makeSound() method.  **Example:**   * Input: Create an object of the Dog class and call its makeSound() method. * Output: "The dog barks"   **Hints:**   * Use the extends keyword for inheritance. * Override the method in the derived class.   class Animal {  void makeSound() {  System.out.println("Animal sound");  }  }  class Dog extends Animal {  void makeSound() {  System.out.println("Dog barks");  }  }  public class Main {  public static void main(String[] args) {  Dog dog = new Dog();  dog.makeSound();  }  } |
| **Question 2: Constructor Inheritance**  **Problem Statement:** Write a Java program that shows how constructors are inherited. Create a base class Person with a constructor that takes name and age as parameters. Then, create a derived class Student that adds a grade property and uses the base class constructor.  **Example:**   * Input: Create a Student object with name, age, and grade. * Output: Display the name, age, and grade of the student.   **Hints:**   * Use the super keyword to call the base class constructor.   class Person {  String name;  Person(String name) {  this.name = name;  }  }  class Student extends Person {  Student(String name) {  super(name);  }  void display() {  System.out.println("Name: " + name);  }  }  public class Main {  public static void main(String[] args) {  Student s = new Student("Alice");  s.display();  }  } |
| **Question 3: Multilevel Inheritance**  **Problem Statement:** Demonstrate multilevel inheritance by creating a base class Vehicle, a derived class Car that extends Vehicle, and another class ElectricCar that extends Car. Add relevant properties and methods to each class.  **Example:**   * Input: Create an ElectricCar object and set its properties like speed, fuelType, and batteryCapacity. * Output: Display all properties of the ElectricCar object.   **Hints:**   * Chain constructors using super() in each derived class.   class Vehicle {  int speed = 100;  }  class Car extends Vehicle {  String fuelType = "Petrol";  }  class ElectricCar extends Car {  int battery = 75;  }  public class Main {  public static void main(String[] args) {  ElectricCar car = new ElectricCar();  System.out.println("Speed: " + car.speed + ", Fuel: " + car.fuelType + ", Battery: " + car.battery);  }  } |
| **Question 4: Method Overriding in Inheritance**  **Problem Statement:** Create a Java program that demonstrates method overriding in inheritance. Create a base class Shape with a method draw() that prints "Drawing a shape". Then, create two derived classes Circle and Rectangle that override the draw() method.  **Example:**   * Input: Create objects of Circle and Rectangle and call their draw() methods. * Output: "Drawing a circle" for Circle and "Drawing a rectangle" for Rectangle.   **Hints:**   * Use the @Override annotation to override methods in derived classes.   class Shape {  void draw() {  System.out.println("Drawing shape");  }  }  class Circle extends Shape {  void draw() {  System.out.println("Drawing circle");  }  }  public class Main {  public static void main(String[] args) {  Circle circle = new Circle();  circle.draw();  }  } |
| **Question 5: Inheritance and Access Modifiers**  **Problem Statement:** Write a Java program that demonstrates how access modifiers affect inheritance. Create a base class Employee with private, protected, and public fields. Create a derived class Manager and try to access these fields.  **Example:**   * Input: Create a Manager object and try to access the private, protected, and public fields. * Output: Compilation errors for the private field, successful access for protected and public fields.   **Hints:**   * Understand how private, protected, and public affect inheritance.   class Employee {  protected int salary = 3000;  }  class Manager extends Employee {  void show() {  System.out.println("Salary: " + salary);  }  }  public class Main {  public static void main(String[] args) {  Manager m = new Manager();  m.show();  }  } |
|  |