INHERITANCE

SLIPTEST-2

29-10-2024

|  |
| --- |
| How classes & Interfaces differ w.r.t multiple inheritance. Justify your answer. |
| Shape Hierarchy Define a class Shape with the following attributes and methods: • Attributes: color (String), filled (boolean). Methods: • Shape(): Default constructor initializing color to "green" and filled to true. • Shape(String color, boolean filled): Parameterized constructor. • getColor(): Getter method for color. • setColor(String color): Setter method for color. • isFilled(): Method to check if the shape is filled. • setFilled(boolean filled): Method to set whether the shape is filledDefine a subclass Circle that inherits from Shape with additional attributes and methods: • Attributes: radius (double). Methods: • Circle(): Default constructor initializing radius to 1.0. • Circle(double radius): Parameterized constructor. • Circle(double radius, String color, boolean filled): Parameterized constructor invoking superclass constructor. • getRadius(): Getter method for radius. • setRadius(double radius): Setter method for radius. • getArea(): Method to calculate and return the area of the circle (area = π \* radius \* radius). • getPerimeter(): Method to calculate and return the perimeter of the circle (perimeter = 2 \* π \* radius). |
| Illustrate The Use of Super keyword with suitable example.What is method overriding? Illustrate the concepts of method overriding and constructor overloading. |
| Illustrate with an example the concepts Nested Interfaces and Inheritance of Interfaces. |
| Explain about Dynamic Method Dispatch with an example |
| Explain about the Object class in Java with its methods. |
| Explain about different types of inheritance with examples |
| Explain the member access mechanism in the inheritance with examples. |
| Create an interface Shape with an abstract method draw(). Create two classes, circle and rectangle which implements interface Shape. Draw() should print the text indicating its shape. Create a reference to the shape and object of either circle or rectangle depending on the user ‘s choice and call the method draw(). |
| Illustrate with an example the concepts Default Methods in Interfaces and Static Methods in Interface. |
| Differentiate abstract classes & Interfaces. |
| Employee Hierarchy Define a superclass Employee with the following attributes and methods: • Attributes: name (String), age (int), designation (String). Methods: • Employee(String name, int age, String designation): Parameterized constructor to initialize attributes. • getName(), setName(String name): Getter and setter methods for name. • getAge(), setAge(int age): Getter and setter methods for age. • getDesignation(), setDesignation(String designation): Getter and setter methods for designation. • displayDetails(): Method to display details of the employee. Define subclasses Manager and Developer that inherit from Employee: Manager: • Additional attribute: department (String). Methods: • Manager(String name, int age, String designation, String department): Parameterized constructor invoking superclass constructor. • Override displayDetails() to include department information. Developer: • Additional attribute: programmingLanguage (String). Methods: • Developer(String name, int age, String designation, String programmingLanguage): Parameterized constructor invoking superclass constructor. • Override displayDetails() to include programming language information. |