CS23304 JAVA PROGRAMMING

Course Instructor: V P Jayachitra

WEEK 6

Instruction:

- **Use meaningful variable names**
- **Consistent indentation**
- **Proper error handling**
- **Proper comment to follow the question requirement**

1. Write a program to show single inheritance

   Class LivingBeing
   Methods: Breath() and Response()
   Class Animal
   Methods: Walk() and NoOfLegs()

2. Write a program to show Multilevel inheritance
   Class LivingBeing

   Methods: Breath() and Response()
   Class Animal
   Methods: Walk() and NoOfLegs()
   Class Cat
   Methods: Meow()
   Class Dog
   Methods: Bark()

3. Write a Java program with:

Create a **Animal** base class with move() method and overloaded versions: move(String direction), move(int distance), move(String direction, int distance), Create a **Dog, Cat, Bird** classes extending Animal. Override all move methods with specific behaviors.

Write the Main method that demonstrates:

Create Animal array with Dog, Cat, Bird objects that shows runtime polymorphism(loop through array calling move method)

Create one Dog object and call all 4 move methods that shows compile-time polymorphism .

Show the output for different movement behaviors for each animal type.

5. Write a Java program to demonstrate Hierarchical inheritance:

Class LivingBeing (Constructor: takes String name Methods: breathe() and response())

Class LandAnimal extends LivingBeing (Constructor: takes String name - use super(name)
Methods: walk() and numberOfLegs())

Class WaterAnimal extends LivingBeing(Constructor: takes String name - use super(name)
Methods: swim() and waterType())

Class Dog extends LandAnimal (Constructor: takes String name - use super(name)
Methods: bark())

Class Cat extends LandAnimal (Constructor: takes String name - use super(name)
Methods: meow())

Create Dog and Cat objects. Call all inherited methods on each object. Show constructor chaining by printing messages in each constructor

6. Write a code to demonstrate the following

i. Trying to override with different return type,

ii. Trying to reduce visibility (public in parent and private in child class)

iii. Wrong overriding syntax (override the method with parameter in child class and without parameter in parent class)

iv. Wrong overloading syntax (same return type vs different return type for same method signature)

v. Covariant type

vi. Method Overloading (Same class, different signatures) and across classes (in inherited class, different signatures)

vii. Static Method Overloading (Same class, different signatures)

viii. Safe and unsafe right vs Wrong upcasting and right vs wrong downcasting