G. Sahitya, 192211567
CSA 0914 Java

Assignment

1. Simple inheritance

```
class Animal {
    String name;
    Public void makeSound() {
        System.out.println("Animal make sound");
    }
}

class Dog Extends Animal {
    Public void makeSound() {
        System.out.println("Dog barks");
    }
}

Public class main {
    Public Static Void main(String[] args){
        Animal animal = new Animal();
        Animal.name = "Generic Animal";
        animal.makeSound();
        Dog dog = new Dog();
        dog.name = "Buddy";
        dog.makeSound();
    }
```

2. Constructor inheritance

```
class Person {
    String name;
    int age;
    Public Person(String name, int age) {
        this.name = name;
        this.age = age;
    }
    Public void displayInfo() {
        System.out.println("Name: "+name+", Age: "+
                           age)
    }
}

class Student extends Person {
    String grade;
    Public Student(String name, int age) {
        Super(name, age);
        this.grade = grade;
    }
    Public void displayInfo() {
        Super.displayInfo();
        System.out.println("grade: "+grade);
    }
}
```

3)

Multilevel inheritance

```
Class Vehicle {
    String brand;
    Public Vehicle(String brand) {
        this.brand = brand;
    }
    Public void displayInfo() {
        System.out.println("Brand:" + brand);
    }
}
class Car Extends Vehicle {
    String Model;
    Public Car(String brand, String Model) {
    Super(brand);
    this Model = Model;
    }
    Public Void displayInfo() {
        Super.displayInfo();
    System.Out.println("Battery Capacity:" +
            battery Capacity + "KWh");
    }
}
```

4. Method overriding in inheritance

```
class Shape {
    Public void draw() {
    System.out.println("Drawing a Shape");
    }
}

class Circle Extends Shape {
    Public void draw() {
    System.out.println("Drawing a Circle");
    }
}

Public class Main {
    Public Static void main(String[] args){
    Shape shape = new Shape();
    Shape.draw();
    Circle circle = new Circle();
    circle.draw();
    Rectangle rectangle = new Rectangle();
    Rectangle.draw();
    }
}
```

5. Inheritance and Access Modifiers

```
Class Employee {
    private int id;
Protected String name;
Public double Salary;
Public Employee (int id, String name, double
                 Salary) {

    this.id = id;
    this.name = name;
    this.Salary = Salary;
}
    Public Void displayManagementInfo () {
System.out.println ("ID:" + id);
System.out.println ("Name:" + Name);
System.out.println ("Salary:" + salary);
}
Public class Main {
    Public Static void main (String[] args) {
Manager manager = new Manager (101, "Alice,"
                               75000.0);

Manager.displayManagerInfo ();
Manager.displayEmployeeInfo ();
}
}
```