SNS COLLEGE OF ENGINEERING

Kurumbapalayam(Po), Coimbatore – 641 107
Accredited by NAAC-UGC with 'A' Grade
Approved by AICTE, Recognized by UGC & Affiliated to Anna University, Chennai

Department of Information Technology

19CS204 OBJECT ORIENTED PROGRAMMING

I YEAR /II SEMESTER

Topic – Protected Member

Inheritance – Protected Member

- The private members of a class cannot be directly accessed outside the class.
- Only methods of that class can access the private members directly.
- However, sometimes it may be necessary for a subclass to access a private member of a superclass.
- If you make a private member public, then anyone can access that member.
- So, if a member of a superclass needs to be (directly) accessed in a subclass and yet still prevent its direct access outside the class, you must declare that member protected.

The `protected` keyword is an access modifier used for attributes, methods and constructors,making them accessible in the same package and subclasses.

Inheritance – Protected Member

Example

```
class Shape
{
   protected double height;  // To hold height.
   protected double width;  //To hold width or base
   public void setValues(double height, double width)
   {
      this.height = height;
      this.width = width;
   }
}
class Rectangle extends Shape
{
   public double getArea()
   {
      return height * width; //accessing protected members
   }
}
class Triangle extends Shape
{
   public double getArea()
   {
      return height * width / 2; //accessing protected members
   }
}
```

```
class Square extends Shape
{
public double getArea()
{
return height * height;
}
}
public class TestProgram
{
   public static void main(String[] args)
    Rectangle rectangle = new Rectangle();
     Triangle triangle = new Triangle();
      Square square = new Square();
       square.setValues(4,10);
 rectangle.setValues(5,4);
 triangle.setValues(5,10);
  System.out.println("Area of rectangle : " +
                        rectangle.getArea());


      System.out.println("Area of triangle : " +
                        triangle.getArea());

    System.out.println("Area of Square:" +square.getArea());
   }
}
```

Inheritance – Constructor in Subclass

Example

```
class Box {
double width;
double height;
double depth;
Box(double w, double h, double d) {
width = w;
height = h;
depth = d;
}
Box() {
width = -1; // use -1 to indicate
height = -1; // an uninitialized
depth = -1; // box
}
Box(double len) {
width = height = depth = len;
}
double volume() {
return width * height * depth;
}
}
```

```
class BoxWeight extends Box {
double weight; // weight of box
BoxWeight(double w, double h, double d, double m) {
width = w;
height = h;
depth = d;
weight = m;
}
}
```

```
class DemoBoxWeight {
public static void main(String args[]) {
BoxWeight mybox1 = new BoxWeight(10, 20, 15, 34.3);
BoxWeight mybox2 = new BoxWeight(2, 3, 4, 0.076);
double vol;
vol = mybox1.volume();
System.out.println("Volume of mybox1 is " + vol);
System.out.println("Weight of mybox1 is " + mybox1.weight);
System.out.println();
vol = mybox2.volume();
System.out.println("Volume of mybox2 is " + vol);
System.out.println("Weight of mybox2 is " + mybox2.weight);
}
}
```

THANK YOU