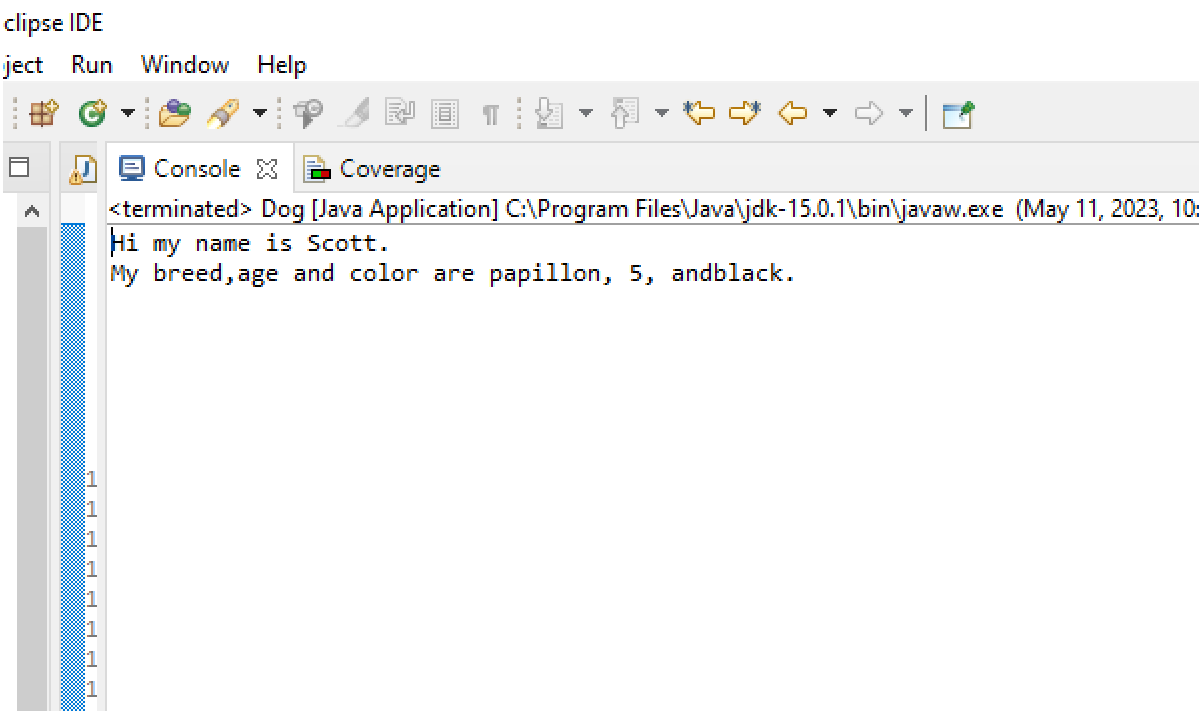
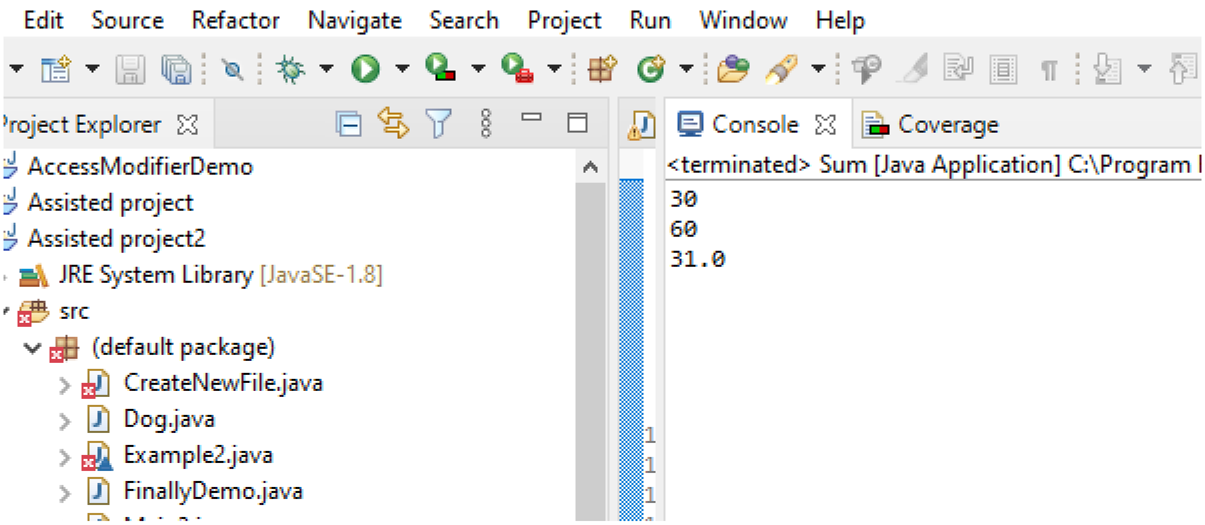
```java
public class Dog
{
    String name;
    String breed;
    int age;
    String color;
    public Dog(String name, String breed, int age, String color)
    {
        this.name = name;
        this.breed = breed;
        this.age = age;
        this.color = color;
    }
    public String getName()
    {
        return name;
    }
    public String getBreed()
    {
        return breed;
    }
    public int getAge()
    {
        return age;
    }
    public String getColor()
    {
        return color;
    }
    @Override
    public String toString()
    {
        return("Hi my name is "+ this.getName()+ ".\nMy breed,age and color are " + 
this.getBreed()+", " + this.getAge()+ ", and"+ this.getColor() + ".");
    }
    public static void main(String[] args)
    {
        Dog scott = new Dog("Scott","papillon", 5, "black");
        System.out.println(scott.toString());
    }
}
```



```
<terminated> Dog [Java Application] C:\Program Files\Java\jdk-15.0.1\bin\javaw.exe (May 11, 2023, 10:
Hi my name is Scott.
My breed,age and color are papillon, 5, andblack.
```

```
lass Sum
{
    public int sum(int x, int y)
    {
        return (x + y);
    }
    public int sum(int x, int y, int z)
    {
        return (x + y + z);
    }
    public double sum(double x, double y)
    {
        return (x + y);
    }
    public static void main(String args[])
    {
        Sum s = new Sum();
        System.out.println(s.sum(10, 20));
        System.out.println(s.sum(10, 20, 30));
        System.out.println(s.sum(10.5, 20.5));
    }
}
```



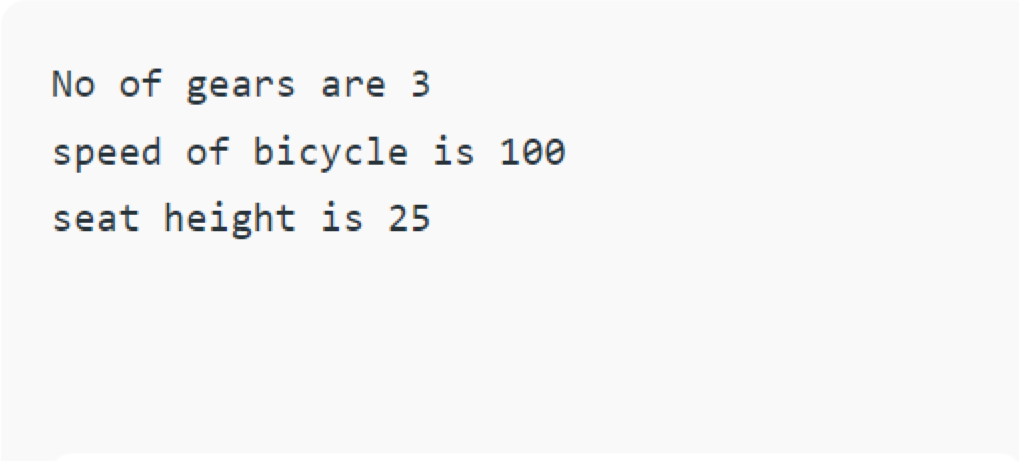
```
<terminated> Sum [Java Application] C:\Program I
30
60
31.0
```

```
// concept of inheritance

// base class
class Bicycle {
        // the Bicycle class has two fields
        public int gear;
        public int speed;

        // the Bicycle class has one constructor
        public Bicycle(int gear, int speed)
```

```
            {
                this.gear = gear;
                this.speed = speed;
            }

            // the Bicycle class has three methods
            public void applyBrake(int decrement)
            {
                speed -= decrement;
            }

            public void speedUp(int increment)
            {
                speed += increment;
            }

            // toString() method to print info of Bicycle
            public String toString()
            {
                return ("No of gears are " + gear + "\n"
                        + "speed of bicycle is " + speed);
            }
}

// derived class
class MountainBike extends Bicycle {

            // the MountainBike subclass adds one more field
            public int seatHeight;
```

```
                // the MountainBike subclass has one constructor
                public MountainBike(int gear, int speed,
                                                     int startHeight)
                {
                       // invoking base-class(Bicycle) constructor
                       super(gear, speed);
                       seatHeight = startHeight;
                }

                // the MountainBike subclass adds one more method
                public void setHeight(int newValue)
                {
                       seatHeight = newValue;
                }

                // overriding toString() method
                // of Bicycle to print more info
                @Override public String toString()
                {
                       return (super.toString() + "\nseat height is "
                                      + seatHeight);
                }
}
public class Test {
                public static void main(String args[])
                {MountainBike mb = new MountainBike(3, 100, 25);
                       System.out.println(mb.toString());
                }
```

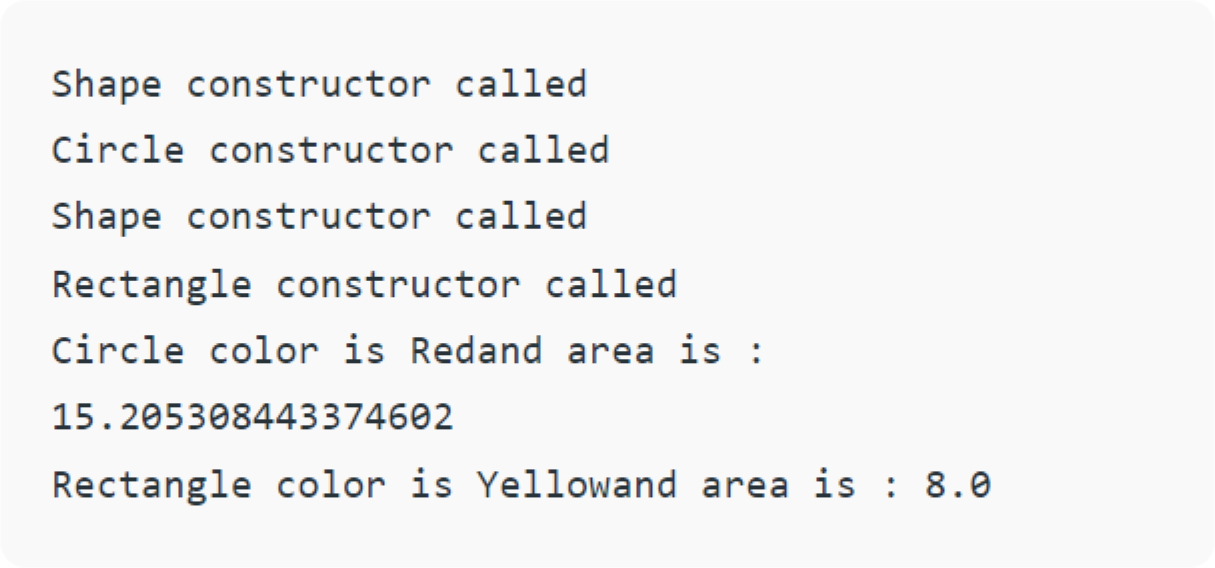
```
}
```



```
No of gears are 3
speed of bicycle is 100
seat height is 25
```

```
abstract class Shape
{
    String color;
    abstract double area();
    public abstract String toString();
    public Shape(String color)
    {
        System.out.println("Shape constructor called");
        this.color = color;
    }
    public String getColor()
    {
        return color;
    }
}
class Circle extends Shape
{
    double radius;
    public Circle(String color,double radius)
    {
        super(color);
        System.out.println("Circle constructor called");
        this.radius = radius;
    }
    @Override
    double area()
    {
        return Math.PI * Math.pow(radius, 2);
    }
    @Override
    public String toString()
    {
        return "Circle color is " + super.color + "and area is : " + area();
    }
}
class Rectangle extends Shape
{
    double length;
    double width;
    public Rectangle(String color,double length,double width)
    {
        super(color);
    }

        System.out.println("Rectangle constructor called");
        this.length = length;
        this.width = width;
    }
    @Override
    double area()
    {
        return length*width;
    }
    @Override
    public String toString()
    {
        return "Rectangle color is " + super.color +
                           "and area is : " + area();
    }
}
public class Test
{
    public static void main(String[] args)
    {
        Shape s1 = new Circle("Red", 2.2);
        Shape s2 = new Rectangle("Yellow", 2, 4);
        System.out.println(s1.toString());
        System.out.println(s2.toString());
    }
```

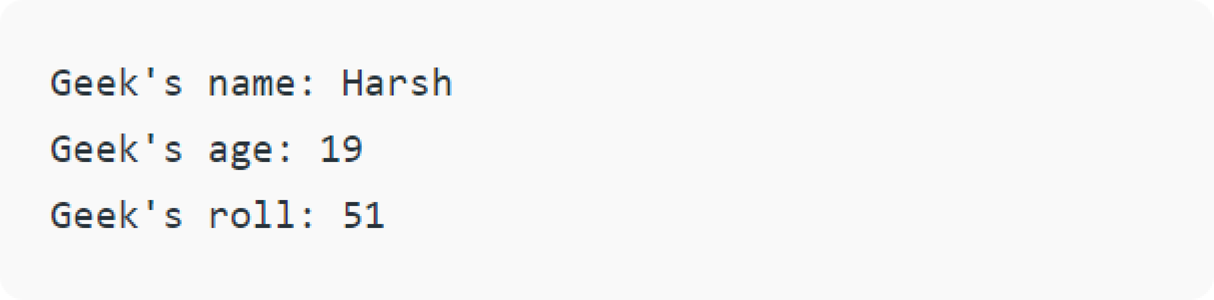
```
}
```



```
Shape constructor called
Circle constructor called
Shape constructor called
Rectangle constructor called
Circle color is Redand area is :
15.205308443374602
Rectangle color is Yellowand area is : 8.0
```

```
public class Encapsulate
{
  private String Name;
  private int Roll;
  private int Age;
  public int getAge()
  {
   return Age;
  }
  public String getName()
  {
   return Name;
  }
  public int getRoll()
```

```
    {
       return Roll;
    }
    public void setAge( int newAge)
    {
      Age = newAge;
    }
    public void setName(String newName)
    {
      Name = newName;
    }
    public void setRoll( int newRoll)
    {
      Roll = newRoll;
    }
}
public class TestEncapsulation
{
    public static void main (String[] args)
    {
        Encapsulate obj = new Encapsulate();
        obj.setName("Harsh");
        obj.setAge(19);
        obj.setRoll(51);
        System.out.println("My name: " + obj.getName());
        System.out.println("My age: " + obj.getAge());

        System.out.println("My roll: " + obj.getRoll());

    }

}
```



```
Geek's name: Harsh
Geek's age: 19
Geek's roll: 51
```