Lab session

01)

Create an interface called "Shape" with two abstract methods: "double calculate Area0" and "double calculatePerimeter()". Implement the "Shape" interface in three classes: "Circle", "Rectangle", and in tt "Triangle". Each class should have private instance variables relevant to its shape, and provide public getter and setter methods for these variables. Additionally, each class should define a constructor that initializes the instance variables. Write the implementation code for the "Shape" interface, the getter and setter methods in each class, and the constructors in each class.

```
package com.mycompany.calshapearea;

interface shape

public interface Shape {
    public abstract double calculateArea();
    public abstract double calculatePerimeter()
}


package com.mycompany.calshapearea;

circle class

public class Circle implements Shape{//implement shpe with circle
    private double rad;
    static double pai=3.141;


    public void setRad(double rad){//set method
      this.rad=rad;
    }
```

```
    public double getRad(){//get method
      return rad;
    }

    public Circle(double rad){//cunstructor method
       this.rad=rad;
    }

    @Override
    public double calculateArea() {//override method in shape interface
      return Math.PI*rad*rad;
        }

    @Override
    public double calculatePerimeter() {//override method in shape interface
      return 2*Math.PI*rad;
    }

}
```

Rectangle class

```
package com.mycompany.calshapearea;
public class Rectangle implements Shape{
  private double width,height;

  public Rectangle(double width,double height){//constructor method
     this.width=width;
     this.height=height;
  }
  public void setWidth(double width){//set width value
```

```
    this.width=width;
  }
  public double getWidth(){//return width value
    return width;
  }
   public void setHeight(double height){//set height value
    this.height=height;
  }
  public double getHeight(){//return height value
    return height;
  }

    @Override
    public double calculateArea() {
        return width*height;
    }

    @Override
    public double calculatePerimeter() {
     return 2*(width+height);
    }

}
```

Triangle class

```
package com.mycompany.calshapearea;

public class Triangle implements Shape{
    private double base,height,side1,side2;
```

```
 public Triangle(double base,double side1,double side2,double height){
  this.base=base;
  this.height=height;
  this.side1=side1;
  this.side2=side2;
 }
 public void setBase(double base){//set base value
 this.base=base;
  }
 public double getBase(){//return base value
 return base;
 }

  public void setSide1(double side1){//set side1 value
 this.side1=side1;
 }
 public double getSide1(){//return side1 value
 return side1;
  }
  public void setSide2(double side2){//set side2 value
 this.side2=side2;
 }
 public double getSide2(){//return side2value
 return side2;
  }
 public void setHeight(double height){//set height value
  this.height=height;
}
public double getHeight(){//return height value
```

```
    return height;
  }
   @Override
   public double calculateArea() {
      return base*height/2;
   }

   @Override
   public double calculatePerimeter() {
      return base*side1*side2;
   }
    }
```

Main class

```
package com.mycompany.calshapearea;
public class CalShapeArea {

   public static void main(String[] args) {
      Circle circle1= new Circle(2);
      System.out.println("Area of Circle:"+ circle1.calculateArea());
      System.out.println("perimeter of Circle:"+ circle1.calculatePerimeter());


      Rectangle recltangle1=new Rectangle(2, 3);
      System.out.println("Area of Rectangle:"+ recltangle1.calculateArea());
       System.out.println("Perimeter of Rectangle:"+ recltangle1.calculatePerimeter());

      Triangle trangle1=new Triangle(2, 2, 3, 4);
      System.out.println("Area of Triangle:"+ trangle1.calculateArea());
      System.out.println("perimeter of Traingle:"+ trangle1.calculatePerimeter());
```

```
    }
}
```

02)

Create a class named "BankAccount" with private instance variables "accountNumber" and "balance." Implement encapsulation by providing public getter and setter methods for both variables. Additionally, create an abstract method called "calculateInterest" in the "BankAccount" class. Implement two subclasses, "Savings Account" and "Checking Account," that extend the "BankAccount" class and provide their own implementations of the "calculateInterest" method. Write the implementation code for the getter and setter methods in the "BankAccount" class, and the "calculateInterest" method in both the "Savings Account" and "CheckingAccount" classes. Assuming that the interest for saving is 12% and checking is 2% (both private variables), find out What will be the interest for a person with 1 million in his checking and 20 million in his saving account.

Bank account parent class/super class

```
package com.mycompany.calshapearea;

public abstract class BankAccount {
    private int accountNumber;
    private double balance;

    public void setAccountNumber(int accountNumber){
      this.accountNumber=accountNumber;
    }
```

```
    public int getAccountNumber(){
      return accountNumber;
    } public void setBalance(double balance){
      this.balance=balance;
    }

    public double getBalance(){
      return balance;
    }
    public abstract double calculateInsert();

}
```

Saving account sub class/child class

```
package com.mycompany.calshapearea;
public class SavingAccount extends BankAccount{
    private double savingInsert=.12;
    @Override
    public double calculateInsert() {
       return getBalance()*savingInsert;
    }

}
```

Checking balance sup class /child class

```
package com.mycompany.calshapearea;
public class CheckingAccount extends BankAccount{
    private double checkingInsert=.02;
    @Override
    public double calculateInsert() {
```

```
        return getBalance()*checkingInsert;
    }

}
```

Main class

```
package com.mycompany.calshapearea;
public class BankDetails {
    public static void main(String[] args) {
        SavingAccount saving=new SavingAccount();
        saving.setAccountNumber(2000025);
         saving.setBalance(20000000);
         System.out.println("Account Number:"+saving.getAccountNumber());
         System.out.println("Insert of saving account:"+saving.calculateInsert());

        CheckingAccount checking = new CheckingAccount();
        checking.setAccountNumber(2000027);
        checking.setBalance(1000000);
         System.out.println("Account Number:"+checking.getAccountNumber());
         System.out.println("Insert of checking account:"+checking.calculateInsert());



    }

}
```