**SAIFULLAH**

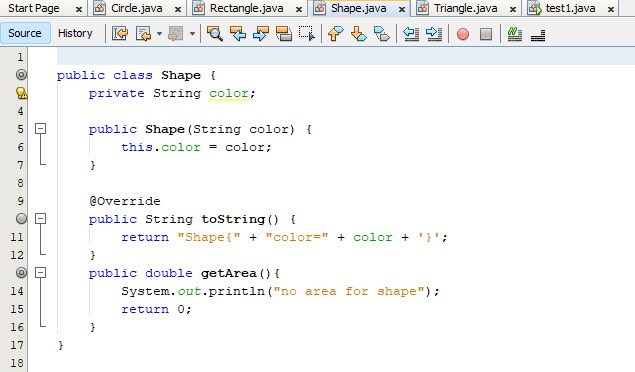
**SP21-BSE-022**

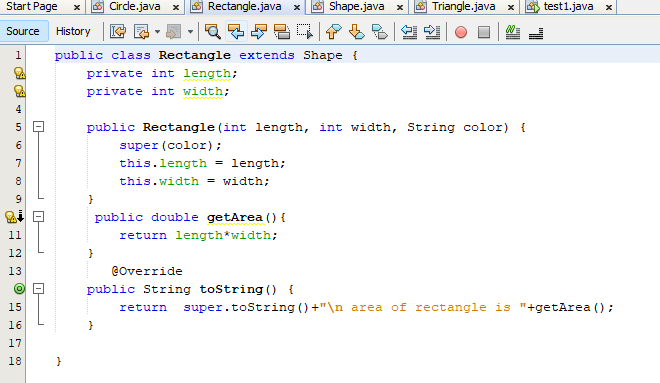
**BSE-3B**

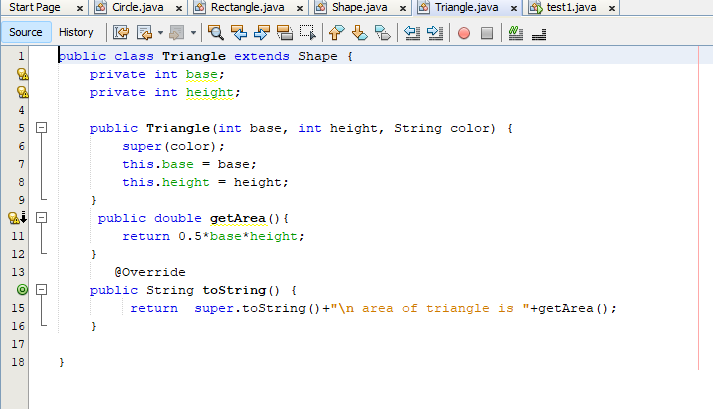
**LAB ASGNT 2 SOL:**

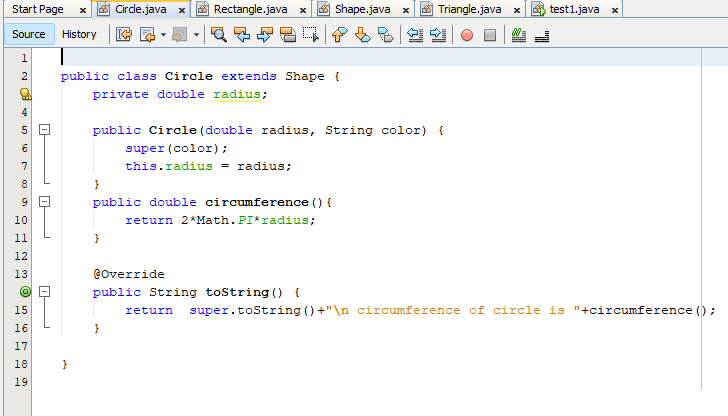
**SCREEN SHOT +CODE**

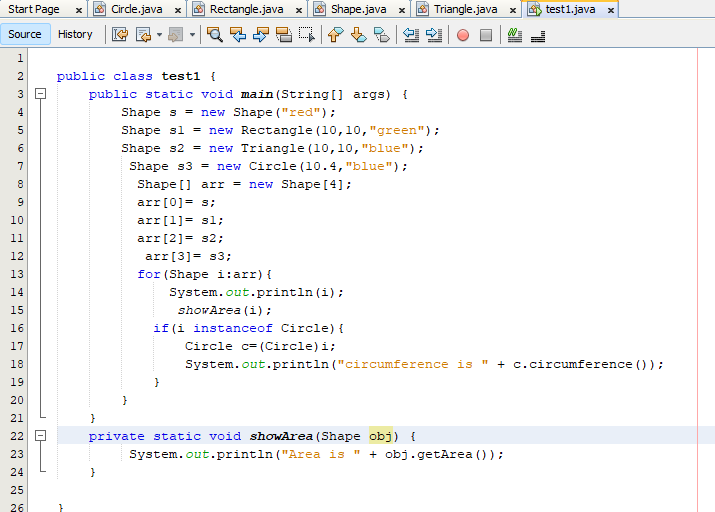
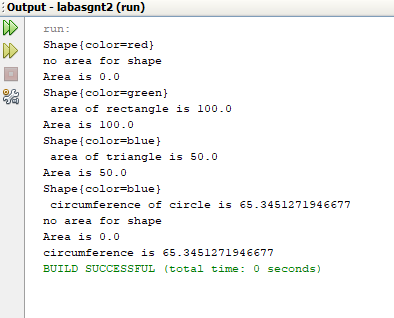
ANSWER 1:











**CODE:**

public class Shape {

private String color;

public Shape(String color) {

this.color = color;

}

@Override

public String toString() {

return "Shape{" + "color=" + color + '}';

}

public double getArea(){

System.out.println("no area for shape");

return 0;

}

}

public class Rectangle extends Shape {

private int length;

private int width;

public Rectangle(int length, int width, String color) {

super(color);

this.length = length;

this.width = width;

}

public double getArea(){

return length\*width;

}

@Override

public String toString() {

return super.toString()+"\n area of rectangle is "+getArea();

}

}

public class Triangle extends Shape {

private int base;

private int height;

public Triangle(int base, int height, String color) {

super(color);

this.base = base;

this.height = height;

}

public double getArea(){

return 0.5\*base\*height;

}

@Override

public String toString() {

return super.toString()+"\n area of triangle is "+getArea();

}

}

public class Circle extends Shape {

private double radius;

public Circle(double radius, String color) {

super(color);

this.radius = radius;

}

public double circumference(){

return 2\*Math.PI\*radius;

}

@Override

public String toString() {

return super.toString()+"\n circumference of circle is "+circumference();

}

}

public class test1 {

public static void main(String[] args) {

Shape s = new Shape("red");

Shape s1 = new Rectangle(10,10,"green");

Shape s2 = new Triangle(10,10,"blue");

Shape s3 = new Circle(10.4,"blue");

Shape[] arr = new Shape[4];

arr[0]= s;

arr[1]= s1;

arr[2]= s2;

arr[3]= s3;

for(Shape i:arr){

System.out.println(i);

showArea(i);

if(i instanceof Circle){

Circle c=(Circle)i;

System.out.println("circumference is " + c.circumference());

}

}

}

private static void showArea(Shape obj) {

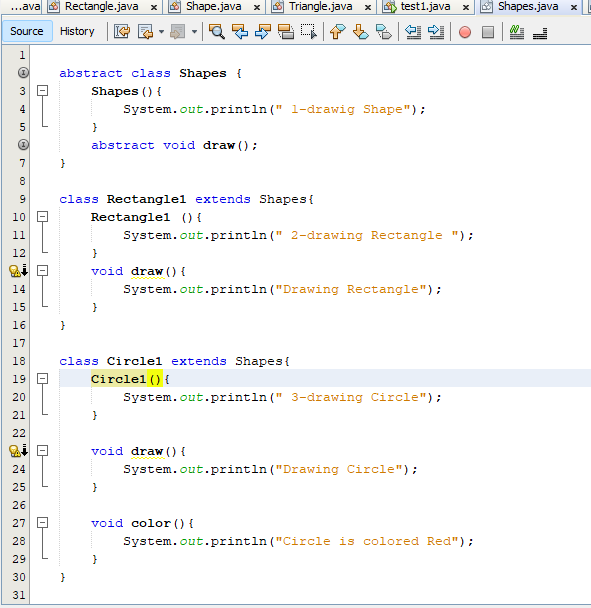
System.out.println("Area is " + obj.getArea());

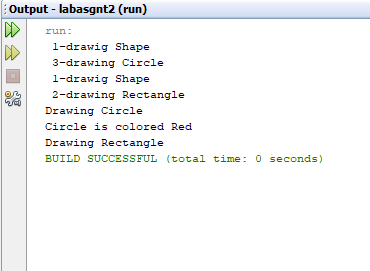
}

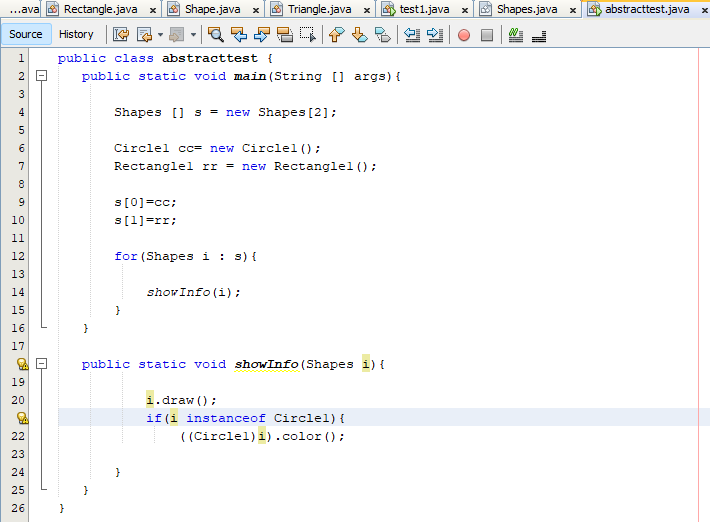
}

----------------------------------------------------------------------------------------------------------------------

**ANSWER 2:**







**CODE:**

**abstract class Shapes {**

**Shapes(){**

**System.out.println(" 1-drawig Shape");**

**}**

**abstract void draw();**

**}**

**class Rectangle1 extends Shapes{**

**Rectangle1 (){**

**System.out.println(" 2-drawing Rectangle ");**

**}**

**void draw(){**

**System.out.println("Drawing Rectangle");**

**}**

**}**

**class Circle1 extends Shapes{**

**Circle1(){**

**System.out.println(" 3-drawing Circle");**

**}**

**void draw(){**

**System.out.println("Drawing Circle");**

**}**

**void color(){**

**System.out.println("Circle is colored Red");**

**}**

**}**

**public class abstracttest {**

**public static void main(String [] args){**

**Shapes [] s = new Shapes[2];**

**Circle1 cc= new Circle1();**

**Rectangle1 rr = new Rectangle1();**

**s[0]=cc;**

**s[1]=rr;**

**for(Shapes i : s){**

**showInfo(i);**

**}**

**}**

**public static void showInfo(Shapes i){**

**i.draw();**

**if(i instanceof Circle1){**

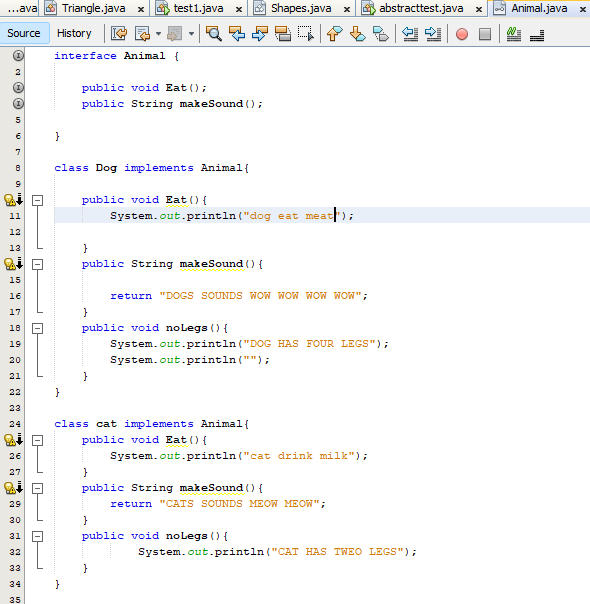
**((Circle1)i).color();**

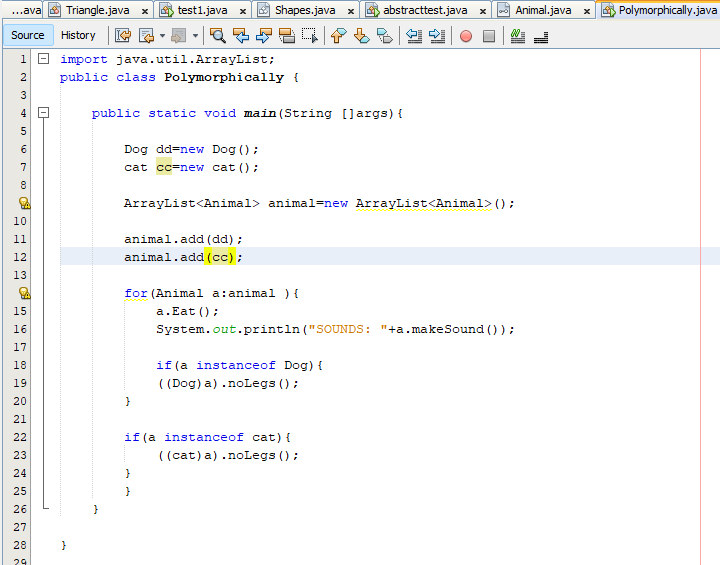
**}**

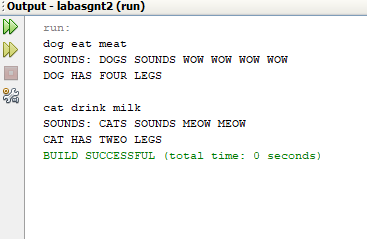
**}**

**}**

**ANSWER 3:**







---------------------------------------------------------------------------------------------------------------------------

**Code:**

interface Animal {

public void Eat();

public String makeSound();

}

class Dog implements Animal{

public void Eat(){

System.out.println("dog eat meat");

}

public String makeSound(){

return "DOGS SOUNDS WOW WOW WOW WOW";

}

public void noLegs(){

System.out.println("DOG HAS FOUR LEGS");

System.out.println("");

}

}

class cat implements Animal{

public void Eat(){

System.out.println("cat drink milk");

}

public String makeSound(){

return "CATS SOUNDS MEOW MEOW";

}

public void noLegs(){

System.out.println("CAT HAS TWEO LEGS");

}

}

import java.util.ArrayList;

public class Polymorphically {

public static void main(String []args){

Dog dd=new Dog();

cat cc=new cat();

ArrayList<Animal> animal=new ArrayList<Animal>();

animal.add(dd);

animal.add(cc);

for(Animal a:animal ){

a.Eat();

System.out.println("SOUNDS: "+a.makeSound());

if(a instanceof Dog){

((Dog)a).noLegs();

}

if(a instanceof cat){

((cat)a).noLegs();

}

}

}

}

-------------------------------------------------------------------