

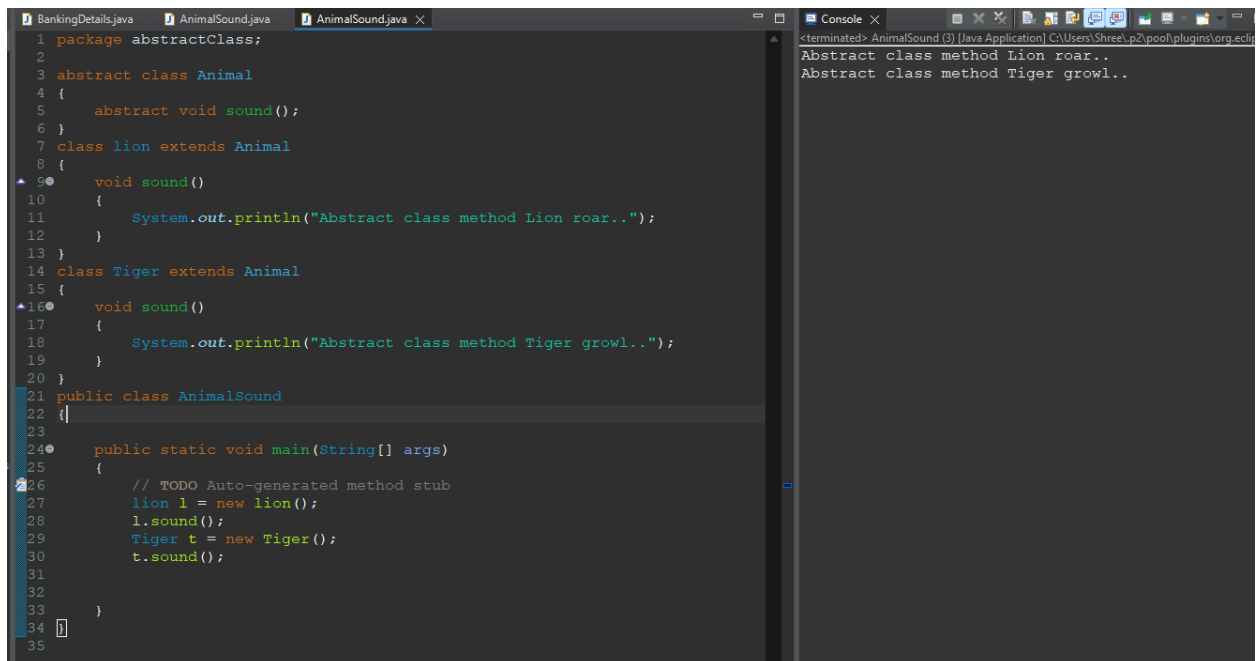
## Assignment No:-27

Name:-Suryawanshi Sangramsingh Sambhaji

Batch: - Delta - DCA (Java) 2024      Date:-12/6/2024

### Abstraction:

1. Write a Java program to create an abstract class **Animal** with an abstract method called **sound()**. Create subclasses **Lion** and **Tiger** that extend the **Animal** class and implement the **sound()** method to make a specific sound for each animal.

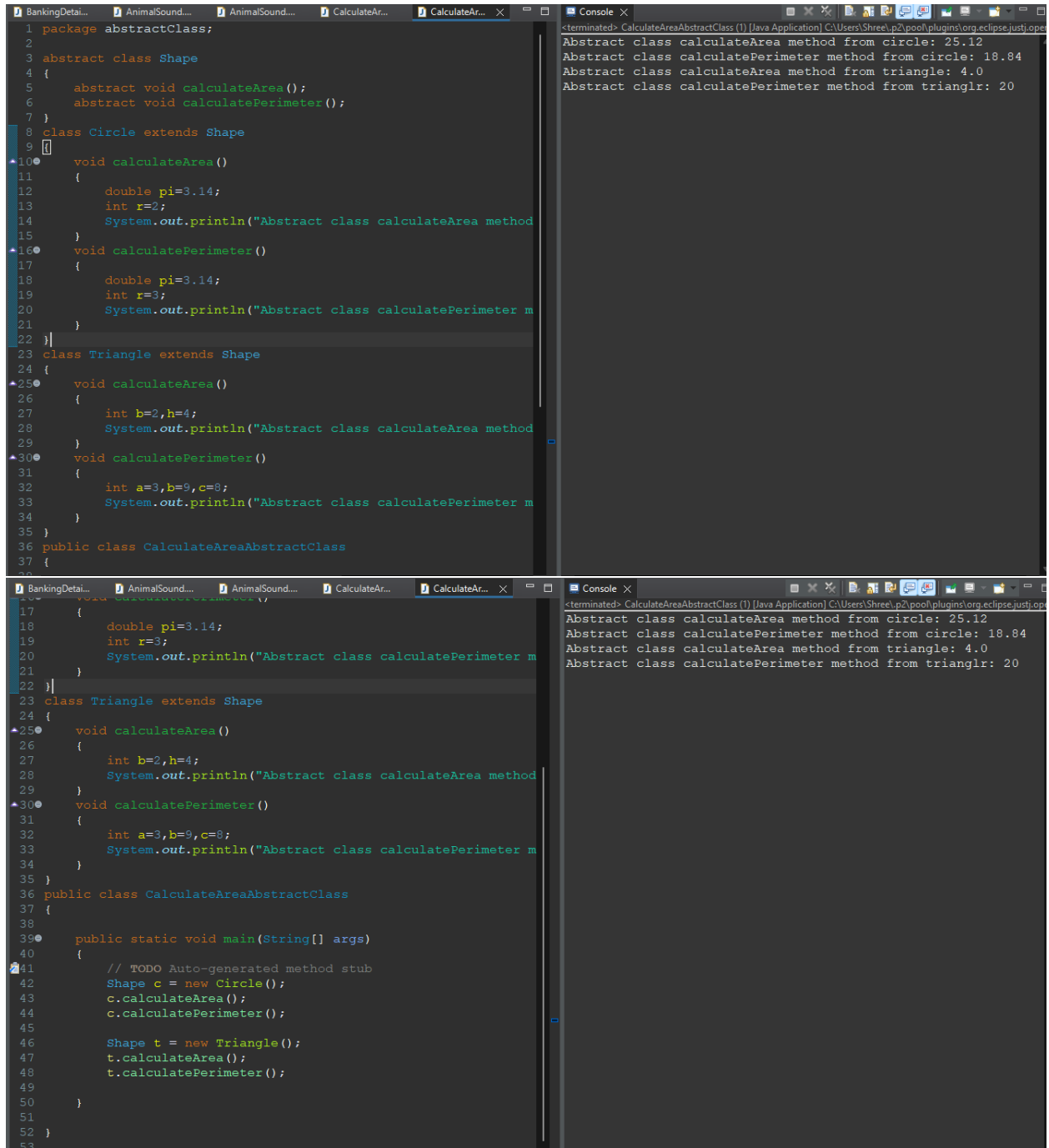


```
1 package abstractClass;
2
3 abstract class Animal
4 {
5     abstract void sound();
6 }
7 class lion extends Animal
8 {
9     void sound()
10    {
11        System.out.println("Abstract class method Lion roar..");
12    }
13 }
14 class Tiger extends Animal
15 {
16     void sound()
17     {
18         System.out.println("Abstract class method Tiger growl..");
19     }
20 }
21 public class AnimalSound
22 {
23
24     public static void main(String[] args)
25     {
26         // TODO Auto-generated method stub
27         lion l = new lion();
28         l.sound();
29         Tiger t = new Tiger();
30         t.sound();
31     }
32 }
33
34
35
```

Console Output:

```
<terminated> AnimalSound (3) [Java Application] C:\Users\Shree.p2\pcofi\plugins\org.eclipse
Abstract class method Lion roar..
Abstract class method Tiger growl..
```

2. Write a Java program to create an abstract class Shape with abstract methods calculateArea() and calculatePerimeter(). Create subclasses Circle and Triangle that extend the Shape class and implement the respective methods to calculate the area and perimeter of each shape.



```
1 package abstractClass;
2
3 abstract class Shape
4 {
5     abstract void calculateArea();
6     abstract void calculatePerimeter();
7 }
8 class Circle extends Shape
9 {
10     void calculateArea()
11     {
12         double pi=3.14;
13         int r=2;
14         System.out.println("Abstract class calculateArea method");
15     }
16     void calculatePerimeter()
17     {
18         double pi=3.14;
19         int r=3;
20         System.out.println("Abstract class calculatePerimeter method");
21     }
22 }
23 class Triangle extends Shape
24 {
25     void calculateArea()
26     {
27         int b=2,h=4;
28         System.out.println("Abstract class calculateArea method");
29     }
30     void calculatePerimeter()
31     {
32         int a=3,b=9,c=8;
33         System.out.println("Abstract class calculatePerimeter method");
34     }
35 }
36 public class CalculateAreaAbstractClass
37 {
38
39     public static void main(String[] args)
40     {
41         // TODO Auto-generated method stub
42         Shape c = new Circle();
43         c.calculateArea();
44         c.calculatePerimeter();
45
46         Shape t = new Triangle();
47         t.calculateArea();
48         t.calculatePerimeter();
49     }
50 }
51
52
53
```

```
<terminated> CalculateAreaAbstractClass (1) [Java Application] C:\Users\Shree\p2\pool\plugins\org.eclipse.justiopen...
Abstract class calculateArea method from circle: 25.12
Abstract class calculatePerimeter method from circle: 18.84
Abstract class calculateArea method from triangle: 4.0
Abstract class calculatePerimeter method from triangle: 20
```

3. Write a Java program to create an abstract class `BankAccount` with abstract methods `deposit()` and `withdraw()`. Create subclasses: `SavingsAccount` and `CurrentAccount` that extend the `BankAccount` class and implement the respective methods to handle deposits and withdrawals for each account type.

```

1 package abstractClass;
2
3 import java.util.Scanner;
4 abstract class BankAccount
5 {
6     abstract void deposit();
7     abstract void withdraw();
8 }
9 class SavingsAccount extends BankAccount
10 {
11     Scanner sc =new Scanner(System.in);
12     int d;
13     int s;
14     public void deposit()
15     {
16         System.out.println("Enter deposit Amount for SavingsAccount: ");
17         int d=sc.nextInt();
18         this.d=d;
19         System.out.println("Your amount is deposited successfully: "+d);
20         System.out.println("Enter choice(2)for checking withdrwal: ");
21         int ch=sc.nextInt();
22         switch(ch)
23         {
24             case 2:this.withdraw();
25             break;
26             default :System.out.println("Invalid number restart please..");
27             break;
28         }
29     }
30     public void withdraw()
31     {
32         System.out.println("Enter withdrwal Amount for SavingsAccount: ");
33         int s=sc.nextInt();
34         this.s=s;
35         System.out.println("Your amount withdrwal is successfully done: "+s);
36     }
37 }
38 class CurrentAccount extends BankAccount
39 {
40     Scanner sc =new Scanner(System.in);
41     int d;
42     int s;
43     public void deposit()
44     {
45         System.out.println("Enter deposit Amount for CurrentAccount: ");
46         int d=sc.nextInt();
47         this.d=d;
48         System.out.println("Your amount is deposited successfully: "+d);
49         System.out.println("Enter choice(2)for checking withdrwal: ");
50         int ch=sc.nextInt();
51         switch(ch)
52         {
53             case 2:this.withdraw();
54             break;
55             default :System.out.println("Invalid number restart please..");
56             break;
57         }
58     }
59     public void withdraw()
60     {
61         System.out.println("Enter withdrwal Amount for CurrentAccount: ");
62         int s=sc.nextInt();
63         this.s=s;
64         System.out.println("Your amount withdrwal is successfully done: "+s);
65     }
66 }
67 }
68 public class BankingDetails
69 {
70
71     public static void main(String[] args)
72     {
73         // TODO Auto-generated method stub
74     }
75 }

```

```

<terminated> BankingDetails (1) [Java Application] C:\Users\Shree\p2\pool\plugins\org.e
Enter deposit Amount for SavingsAccount:
1200
Your amount is deposited successfully: 1200
Enter choice(2)for checking withdrwal:
2
Enter withdrwal Amount for SavingsAccount:
122
Your amount withdrwal is successfully done: 122
Enter deposit Amount for CurrentAccount:
1000
Your amount is deposited successfully: 1000
Enter choice(2)for checking withdrwal:
388
Invalid number restart please..

```

```

<terminated> BankingDetails (1) [Java Application] C:\Users\Shree\p2\pool\plugins\org.e
Enter deposit Amount for SavingsAccount:
1200
Your amount is deposited successfully: 1200
Enter choice(2)for checking withdrwal:
2
Enter withdrwal Amount for SavingsAccount:
122
Your amount withdrwal is successfully done: 122
Enter deposit Amount for CurrentAccount:
1000
Your amount is deposited successfully: 1000
Enter choice(2)for checking withdrwal:
388
Invalid number restart please..

```

```

47     int d=sc.nextInt();
48     this.d=d;
49     System.out.println("Your amount is deposited successfully: "+d);
50     System.out.println("Enter choice(2)for checking withdrwal: ");
51     int ch=sc.nextInt();
52     switch(ch)
53     {
54         case 2:this.withdraw();
55         break;
56         default :System.out.println("Invalid number restart please..");
57         break;
58     }
59 }
60 public void withdraw()
61 {
62     System.out.println("Enter withdrwal Amount for CurrentAccount: ");
63     int s=sc.nextInt();
64     this.s=s;
65     System.out.println("Your amount withdrwal is successfully done: "+s);
66 }
67 }
68 public class BankingDetails
69 {
70
71     public static void main(String[] args)
72     {
73         // TODO Auto-generated method stub
74
75         BankAccount a1 = new SavingsAccount();
76         a1.deposit();
77
78         BankAccount a2 = new CurrentAccount();
79         a2.deposit();
80     }
81 }
82 }

```

```

<terminated> BankingDetails (1) [Java Application] C:\Users\Shree\p2\pool\plugins\org.ecl
Enter deposit Amount for SavingsAccount:
1200
Your amount is deposited successfully: 1200
Enter choice(2)for checking withdrwal:
2
Enter withdrwal Amount for SavingsAccount:
122
Your amount withdrwal is successfully done: 122
Enter deposit Amount for CurrentAccount:
1000
Your amount is deposited successfully: 1000
Enter choice(2)for checking withdrwal:
388
Invalid number restart please..

```

**4. Write a Java program to create an abstract class Animal with abstract methods eat() and sleep(). Create subclasses Lion, Tiger, and Deer that extend the Animal class and implement the eat() and sleep() methods differently based on their specific behavior.**

```

1 package abstractClass;
2 abstract class Animal12
3 {
4     abstract void eat();
5     abstract void sleep();
6 }
7 class lion12 extends Animal12
8 {
9     void eat()
10    {
11        System.out.println("Abstract class method Lion eating..");
12    }
13    void sleep()
14    {
15        System.out.println("Abstract class method Lion sleeping..");
16    }
17 }
18 class Tiger12 extends Animal12
19 {
20     void eat()
21     {
22         System.out.println("Abstract class method Tiger eating..");
23     }
24     void sleep()
25     {
26         System.out.println("Abstract class method Tiger sleeping..");
27     }
28 }
29 class Deer12 extends Animal12
30 {
31     void eat()
32     {
33         System.out.println("Abstract class method Deer eating..");
34     }
35     void sleep()
36     {
37         System.out.println("Abstract class method Deer sleeping..");
38     }
39 }

```

```

<terminated> AnimalEatSleep [Java Application] C:\Users\Shree\p2\pool\plugins\org
Abstract class method Lion eating..
Abstract class method Lion sleeping..
Abstract class method Tiger eating..
Abstract class method Tiger sleeping..
Abstract class method Deer eating..
Abstract class method Deer sleeping..

```

```

22     System.out.println("Abstract class method Tiger eating..");
23 }
24 void sleep()
25 {
26     System.out.println("Abstract class method Tiger sleeping..");
27 }
28 }
29 class Deer12 extends Animal12
30 {
31     void eat()
32     {
33         System.out.println("Abstract class method Deer eating..");
34     }
35     void sleep()
36     {
37         System.out.println("Abstract class method Deer sleeping..");
38     }
39 }
40 public class AnimalEatSleep
41 {
42 }
43 public static void main(String[] args)
44 {
45     // TODO Auto-generated method stub
46     lion12 l = new lion12();
47     l.eat();
48     l.sleep();
49     Tiger12 t = new Tiger12();
50     t.eat();
51     t.sleep();
52     Deer12 d = new Deer12();
53     d.eat();
54     d.sleep();
55 }
56 }
57 }
58

```

```

<terminated> AnimalEatSleep [Java Application] C:\Users\Shree\p2\pool\plugins\org.eclipse
Abstract class method Lion eating..
Abstract class method Lion sleeping..
Abstract class method Tiger eating..
Abstract class method Tiger sleeping..
Abstract class method Deer eating..
Abstract class method Deer sleeping..

```

5. Write a Java program to create an abstract class Employee with abstract methods calculateSalary() and displayInfo(). Create subclasses Manager and Programmer that extend the Employee class and implement the respective methods to calculate salary and display information for each role.

```

1 package abstractClass;
2 abstract class Employee
3 {
4     abstract void calculateSalary();
5     abstract void displayInfo();
6 }
7 }
8 class manager extends Employee
9 {
10     void calculateSalary()
11     {
12         int sal=50000,bonus=5000;
13         System.out.println("manager sal is: "+(sal+bonus));
14     }
15 }
16 }
17 void displayInfo()
18 {
19     String name="ABC";
20     System.out.println("name of employee: "+name);
21 }
22 }
23 }
24 }
25 class programmer extends Employee
26 {
27     void calculateSalary()
28     {
29         int sal=30000,bonus=2500;
30         System.out.println("Programmer sal is: "+(sal+bonus));
31     }
32 }
33 void displayInfo()
34 {
35     String name="HP";
36     System.out.println("name of programmer: "+name);
37 }
38 }
39 }

```

```

<terminated> CalculateSphere [Java Application] C:\Users\Shree\p2\pool\plugins\org.ec
name of employee:ABC
manager sal is:55000
name of programmer:HP
Programmer sal is:32500

```

```

19 {
20     String name="ABC";
21     System.out.println("name of employee:"+name);
22 }
23
24 }
25 class programmer extends Employee
26 {
27     void calculateSalary()
28     {
29         int sal=30000,bonus=2500;
30         System.out.println("Programmer sal is:"+sal+bonus);
31     }
32     void displayInfo()
33     {
34         String name="HP";
35         System.out.println("name of programmer:"+name);
36     }
37 }
38 }
39 public class CalculateSphere
40 {
41
42     public static void main(String[] args)
43     {
44         // TODO Auto-generated method stub
45         Employee E=new manager();
46         E.displayInfo();
47         E.calculateSalary();
48         Employee E1=new programmer();
49         E1.displayInfo();
50         E1.calculateSalary();
51     }
52 }
53
54 }
55

```

```

<terminated> CalculateSphere [Java Application] C:\Users\Shree\p2\poo\plugins\org.
name of employee:ABC
manager sal is:55000
name of programmer:HP
Programmer sal is:32500

```

**6. Write a Java program to create an abstract class Shape3D with abstract methods calculateVolume() and calculateSurfaceArea(). Create subclasses Sphere and Cube that extend the Shape3D class and implement the respective methods to calculate the volume and surface area of each shape.**

```

1 package abstractClass;
2 abstract class shape3D
3 {
4     abstract void calculateVolume();
5     abstract void calculateSurfaceArea();
6
7
8 }
9 class sphere extends shape3D
10 {
11     void calculateVolume()
12     {
13         int r=5;
14         double result=(4/3)*(3.14)*r*r*r;
15         System.out.println("volume of sphere is:"+result);
16     }
17     void calculateSurfaceArea()
18     {
19         int r=6;
20         double result=4*(3.14)*r*r;
21         System.out.println("surface of sphere is:"+result);
22     }
23 }
24
25 }
26 class cube extends shape3D
27 {
28     void calculateVolume()
29     {
30         int v=6;
31         int result=v*v*v;
32         System.out.println("volume of cube is:"+result);
33     }
34     void calculateSurfaceArea()
35     {
36         int sa=3;
37         int result=6*sa*sa;
38         System.out.println("surface of cube is:"+result);
39     }
40 }

```

```

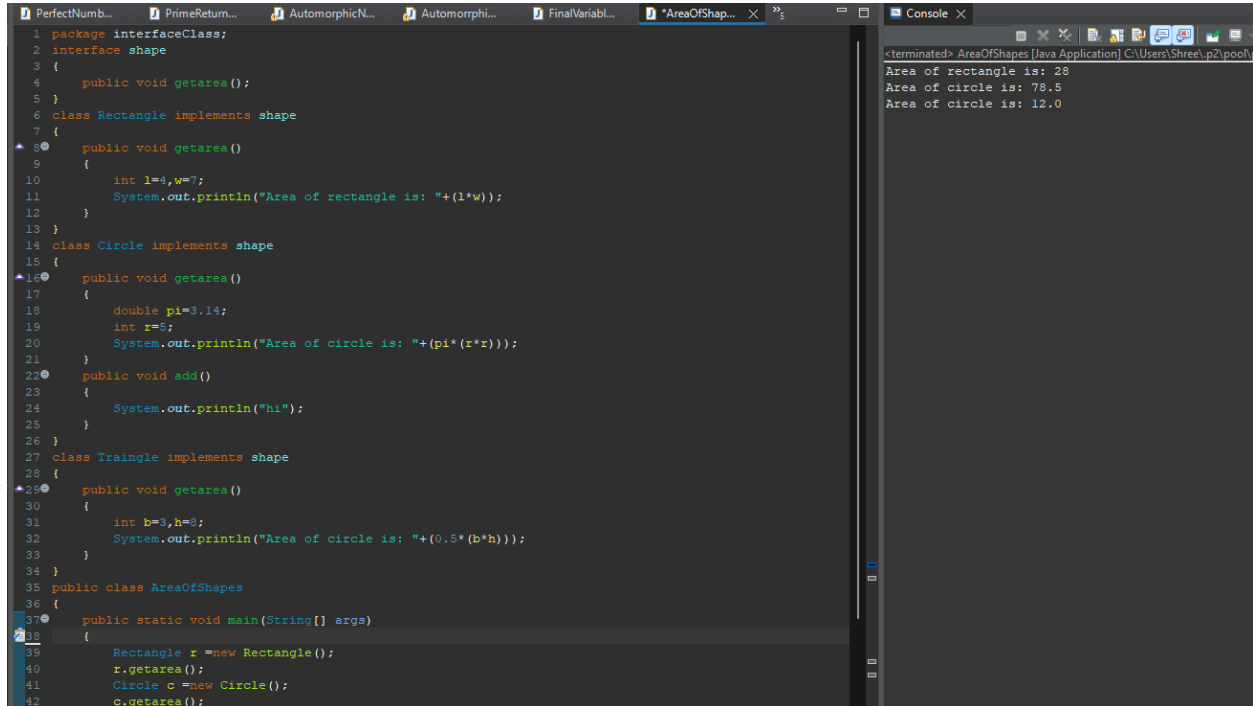
<terminated> EmployeeAbstract [Java Application] C:\Users\Shree\p2\poo\plugins\org.eclip
volume of sphere is:392.5
surface of sphere is:452.15999999999997
-----
volume of cube is:216
surface of cube is:54

```

```
BankingDeta... CalculateAre... BankingDeta... AnimalEatSle... CalculateSal... EmployeeAbst... Console X
23     }
24
25 }
26 class cube extends shape3D
27 {
28     void calculateVolume ()
29     {
30         int v=6;
31         int result=v*v*v;
32         System.out.println("volume of cube is:"+result);
33     }
34     void calculateSurfaceArea ()
35     {
36         int sa=3;
37         int result=6*sa*sa;
38         System.out.println("surface of cube is:"+result);
39     }
40 }
41
42 }
43 public class EmployeeAbstract
44 {
45
46     public static void main(String[] args)
47     {
48         // TODO Auto-generated method stub
49         shape3D s=new sphere();
50         s.calculateVolume();
51         s.calculateSurfaceArea();
52         System.out.println("-----");
53         shape3D s1=new cube();
54         s1.calculateVolume();
55         s1.calculateSurfaceArea();
56     }
57 }
58 }
59
<terminated> EmployeeAbstract [Java Application] C:\Users\Shree.p2\poo\plugins\org.eclipse
volume of sphere is:392.5
surface of sphere is:452.15999999999997
-----
volume of cube is:216
surface of cube is:54
```

## Interface:

1. Write a Java program to create an interface Shape with the getArea() method. Create three classes Rectangle, Circle, and Triangle that implement the Shape interface. Implement the getArea() method for each of the three classes.

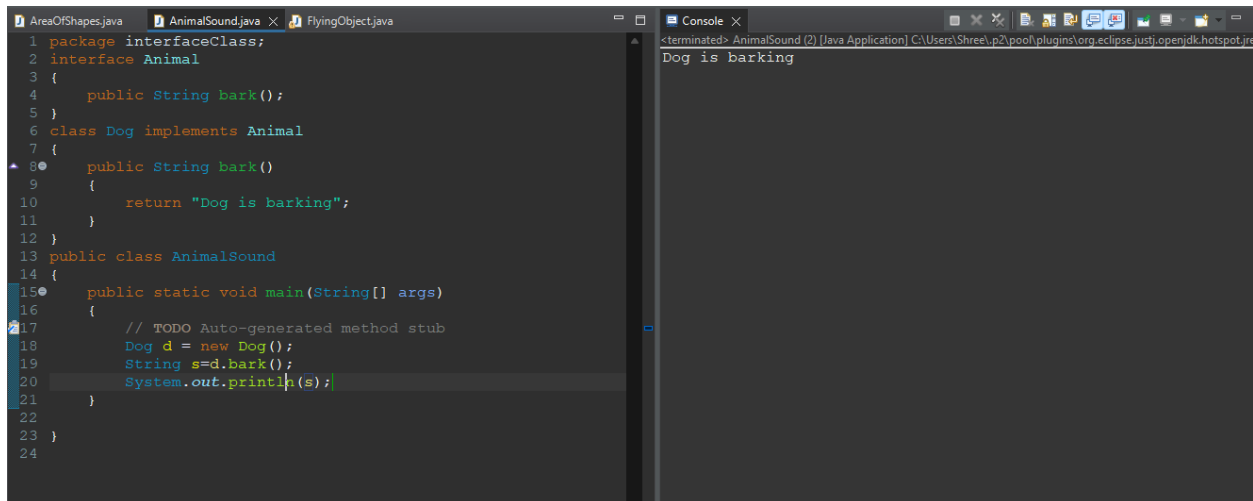


```
1 package interfaceClass;
2 interface shape
3 {
4     public void getarea();
5 }
6 class Rectangle implements shape
7 {
8     public void getarea()
9     {
10         int l=4,w=7;
11         System.out.println("Area of rectangle is: "+(l*w));
12     }
13 }
14 class Circle implements shape
15 {
16     public void getarea()
17     {
18         double pi=3.14;
19         int r=5;
20         System.out.println("Area of circle is: "+(pi*(r*r)));
21     }
22     public void add()
23     {
24         System.out.println("hi");
25     }
26 }
27 class Traingle implements shape
28 {
29     public void getarea()
30     {
31         int b=3,h=8;
32         System.out.println("Area of circle is: "+(0.5*(b*h)));
33     }
34 }
35 public class AreaOfShapes
36 {
37     public static void main(String[] args)
38     {
39         Rectangle r =new Rectangle();
40         r.getarea();
41         Circle c =new Circle();
42         c.getarea();
```

Console output:

```
<terminated> AreaOfShapes [Java Application] C:\Users\Shree\p2\pool\p...
Area of rectangle is: 28
Area of circle is: 78.5
Area of circle is: 12.0
```

2. Write a Java program to create a Animal interface with a method called bark() that takes no arguments and returns void. Create a Dog class that implements Animal and overrides speak() to print "Dog is barking".



```
1 package interfaceClass;
2 interface Animal
3 {
4     public String bark();
5 }
6 class Dog implements Animal
7 {
8     public String bark()
9     {
10         return "Dog is barking";
11     }
12 }
13 public class AnimalSound
14 {
15     public static void main(String[] args)
16     {
17         // TODO Auto-generated method stub
18         Dog d = new Dog();
19         String s=d.bark();
20         System.out.println(s);
21     }
22 }
23 }
24
```

Console output:

```
<terminated> AnimalSound (2) [Java Application] C:\Users\Shree\p2\pool\plugins\org.eclipse.jdti.openjdk.hotspot.jre
Dog is barking
```



**3. Write a Java program to create an interface Flyable with a method called fly obj(). Create three classes Spacecraft, Airplane, and Helicopter that implement the Flyable interface. Implement the fly\_obj() method for each of the three classes.**

The screenshot displays the Eclipse IDE with two tabs: 'FlyingObject.java' and 'Console'.

**FlyingObject.java:**

```

1 package interfaceClass;
2 interface Flyable
3 {
4     public void flyObj();
5 }
6 class Spacecraft implements Flyable
7 {
8     public void flyObj()
9     {
10         System.out.println("Spacecraft implements Flyable with flyObj method in interface");
11     }
12 }
13 class Airplane implements Flyable
14 {
15     public void flyObj()
16     {
17         double pi=3.14;
18         int r=5;
19         System.out.println("Airplane implements Flyable with flyObj method in interface");
20     }
21 }
22 class Helicopter implements Flyable
23 {
24     public void flyObj()
25     {
26         int b=3,h=8;
27         System.out.println("Helicopter implements Flyable with flyObj method in interface");
28     }
29 }
30 public class FlyingObject
31 {
32     // TODO Auto-generated method stub
33     public static void main(String[] args) {
34         Flyable f = new Spacecraft();
35         f.flyObj();
36
37         Flyable f1 = new Airplane();
38         f1.flyObj();
39
40         Flyable f2 = new Helicopter();
41         f2.flyObj();
42     }
43 }

```

**Console:**

```

<terminated> FlyingObject [Java Application] C:\Users\Shree\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full\jre\bin\java.exe
Spacecraft implements Flyable with flyObj method in interface
Airplane implements Flyable with flyObj method in interface
Helicopter implements Flyable with flyObj method in interface

```

4. Write a Java programming to create a banking system with three classes - Bank, Account, SavingsAccount, and CurrentAccount. The bank should have a list of accounts and methods for adding them. Accounts should be an interface with methods to deposit, withdraw, calculate interest, and view balances. SavingsAccount and CurrentAccount should implement the Account interface and have their own unique methods.

```
1 package interfaceClass;
2
3 import java.util.Scanner;
4
5 interface Account
6 {
7     public void deposit();
8     public void withdraw();
9     public void calculateInterest();
10    public void viewBalances();
11 }
12 class BankAccount implements Account
13 {
14     Scanner sc =new Scanner(System.in);
15     int d;
16     int s;
17     public void deposit()
18     {
19         System.out.println("Enter deposite Amount for BankAccount: ");
20         int d=sc.nextInt();
21         this.d=d;
22         System.out.println("Your amount is deposited successfully: "+d);
23         System.out.println("Enter choice(1) (2)for checking calculateInterest/withdrwal: ");
24         int ch=sc.nextInt();
25         switch(ch)
26         {
27             case 1:this.calculateInterest();
28             break;
29             case 2:this.withdraw();
30             break;
31             default :System.out.println("Invalid number restart please..");
32             break;
33         }
34     }
35     public void withdraw()
36     {
37         System.out.println("Enter withdrwal Amount for BankAccount: ");
38         int s=sc.nextInt();
```

```

AreaOfShapes.java  AnimalSound.java  FlyingObject.java  BankingDetails.java X
37     System.out.println("Enter withdrwal Amount for BankAccount: ");
38     int s=sc.nextInt();
39     this.s=s;
40     System.out.println("Your amount withdrwal is successfully done: "+s);
41     System.out.println("Enter choice(1)for checking viewBalances: ");
42     int ch=sc.nextInt();
43     switch(ch)
44     {
45     case 1:this.viewBalances();
46     break;
47     default :System.out.println("Invalid number restart please..");
48     break;
49     }
50 }
51 public void calculateInterest()
52 {
53     System.out.println("Enter Principal amount (the beginning balance): ");
54     long b =sc.nextLong();
55     System.out.println("Enter R = Interest rate (usually per year, expressed as a decimal): ");
56     double d =sc.nextDouble();
57     System.out.println("Enter T = Number of time periods (generally one-year time periods): ");
58     int t =sc.nextInt();
59     double si=(b*d*t)/100;
60     System.out.println("Your simple intrest is: "+si);
61     System.out.println("Enter choice(1)for checking viewBalances: ");
62     int ch=sc.nextInt();
63     switch(ch)
64     {
65     case 1:this.viewBalances();
66     break;
67     default :System.out.println("Invalid number restart please..");
68     break;
69     }
70 }
71 public void viewBalances()
72 {
73     int c=this.d-=this.s;
74     System.out.println("Your BankAccount balance is: "+c);

```

```

AreaOfShapes.java  AnimalSound.java  FlyingObject.java  BankingDetails.java X
73     int c=this.d-=this.s;
74     System.out.println("Your BankAccount balance is: "+c);
75     System.out.println("Enter choice(1)(0)for checking calculateInterest/exit the BankAccount :
76     int ch=-1;
77     while(ch!=0)
78     {
79         ch = sc.nextInt();
80         switch(ch)
81         {
82         case 1:this.calculateInterest();
83         break;
84         default :System.out.println("Invalid number restart please..");
85         break;
86         }
87     }
88 }
89 }
90 class SavingsAccount implements Account
91 {
92
93     Scanner sc =new Scanner(System.in);
94     int d;
95     int s;
96     public void deposit()
97     {
98         System.out.println("Enter deposite Amount for SavingsAccount: ");
99         int d=sc.nextInt();
100        this.d=d;
101        System.out.println("Your amount is deposited successfuly: "+d);
102        System.out.println("Enter choice(1)(2)for checking calculateInterest/withdrwal: ");
103        int ch=sc.nextInt();
104        switch(ch)
105        {
106        case 1:this.calculateInterest();
107        break;
108        case 2:this.withdraw();
109        break;
110        default :System.out.println("Invalid number restart please..");

```

```

106     case 1: this.calculateInterest();
107     break;
108     case 2: this.withdraw();
109     break;
110     default : System.out.println("Invalid number restart please..");
111     break;
112 }
113 }
114 public void withdraw()
115 {
116     System.out.println("Enter withdrwal Amount for SavingsAccount: ");
117     int s=sc.nextInt();
118     this.s=s;
119     System.out.println("Your amount withdrwal is successfuly done: "+s);
120     System.out.println("Enter choice(1)for checking viewBalances: ");
121     int ch=sc.nextInt();
122     switch(ch)
123     {
124         case 1: this.viewBalances();
125         break;
126         default : System.out.println("Invalid number restart please..");
127         break;
128     }
129 }
130 public void calculateInterest()
131 {
132     System.out.println("Enter Principal amount (the beginning balance): ");
133     long b =sc.nextLong();
134     System.out.println("Enter R = Interest rate (usually per year, expressed as a decimal): ");
135     double d =sc.nextDouble();
136     System.out.println("Enter T = Number of time periods (generally one-year time periods): ");
137     int t =sc.nextInt();
138     double si=(b*d*t)/100;
139     System.out.println("Your simple intrest is: "+si);
140     System.out.println("Enter choice(1)for checking viewBalances: ");
141     int ch=sc.nextInt();
142     switch(ch)
143     {

```

```

142     switch(ch)
143     {
144         case 1: this.viewBalances();
145         break;
146         default : System.out.println("Invalid number restart please..");
147         break;
148     }
149 }
150 public void viewBalances()
151 {
152     int c=this.d==this.s;
153     System.out.println("Your SavingsAccount balance is: "+c);
154     System.out.println("Enter choice(1) (0)for checking calculateInterest/exit the SavingsAccount");
155     int ch=-1;
156     while(ch!=0)
157     {
158         ch = sc.nextInt();
159         switch(ch)
160         {
161             case 1: this.calculateInterest();
162             break;
163             default : System.out.println("Invalid number restart please..");
164             break;
165         }
166     }
167 }
168 }
169 class CurrentAccount implements Account
170 {
171     Scanner sc =new Scanner(System.in);
172     int d;
173     int s;
174     public void deposit()
175     {
176         System.out.println("Enter deposite Amount for CurrentAccount: ");
177         int d=sc.nextInt();
178         this.d=d;
179     }

```

```

AreaOfShapes.java  AnimalSound.java  FlyingObject.java  BankingDetails.java X
175● public void deposit ()
176 {
177     System.out.println("Enter deposite Amount for CurrentAccount: ");
178     int d=sc.nextInt();
179     this.d=d;
180     System.out.println("Your amount is deposited successfuly: "+d);
181     System.out.println("Enter choice(1) (2)for checking calculateInterest/withdrwal: ");
182     int ch=sc.nextInt();
183     switch(ch)
184     {
185         case 1:this.calculateInterest();
186         break;
187         case 2:this.withdraw();
188         break;
189         default :System.out.println("Invalid number restart please..");
190         break;
191     }
192 }
193● public void withdraw()
194 {
195     System.out.println("Enter withdrwal Amount for CurrentAccount: ");
196     int s=sc.nextInt();
197     this.s=s;
198     System.out.println("Your amount withdrwal is successfuly done: "+s);
199     System.out.println("Enter choice(1)for checking viewBalances: ");
200     int ch=sc.nextInt();
201     switch(ch)
202     {
203         case 1:this.viewBalances();
204         break;
205         default :System.out.println("Invalid number restart please..");
206         break;
207     }
208 }
209● public void calculateInterest()
210 {
211     System.out.println("Enter Principal amount (the beginning balance): ");
212     long b = sc.nextLong();

```

```

AreaOfShapes.java  AnimalSound.java  FlyingObject.java  BankingDetails.java X
205     default :System.out.println("Invalid number restart please..");
206     break;
207 }
208 }
209● public void calculateInterest()
210 {
211     System.out.println("Enter Principal amount (the beginning balance): ");
212     long b =sc.nextLong();
213     System.out.println("Enter R = Interest rate (usually per year, expressed as a decimal): ");
214     double d =sc.nextDouble();
215     System.out.println("Enter T = Number of time periods (generally one-year time periods): ");
216     int t =sc.nextInt();
217     double si=(b*d*t)/100;
218     System.out.println("Your simple intrest is: "+si);
219     System.out.println("Enter choice(1)for checking viewBalances: ");
220     int ch=sc.nextInt();
221     switch(ch)
222     {
223         case 1:this.viewBalances();
224         break;
225         default :System.out.println("Invalid number restart please..");
226         break;
227     }
228 }
229● public void viewBalances()
230 {
231     int c=this.d==this.s;
232     System.out.println("Your CurrentAccount balance is: "+c);
233     System.out.println("Enter choice(1) (0)for checking calculateInterest/exit the CurrentAccount: ");
234     int ch=-1;
235     while(ch!=0)
236     {
237         ch = sc.nextInt();
238         switch(ch)
239         {
240             case 1:this.calculateInterest();
241             break;
242             default :System.out.println("Invalid number restart please..");

```

```

AreaOfShapes.java  AnimalSound.java  FlyingObject.java  BankingDetails.java
226
227     }
228 }
229 public void viewBalances()
230 {
231     int c=this.d==this.s;
232     System.out.println("Your CurrentAccount balance is: "+c);
233     System.out.println("Enter choice(1) (0) for checking calculateInterest/exit the CurrentAccount");
234     int ch=-1;
235     while(ch!=0)
236     {
237         ch = sc.nextInt();
238         switch(ch)
239         {
240             case 1:this.calculateInterest();
241             break;
242             default :System.out.println("Invalid number restart please..");
243             break;
244         }
245     }
246 }
247 }
248 public class BankingDetails {
249
250     public static void main(String[] args) {
251         // TODO Auto-generated method stub
252         Account a = new BankAccount();
253         a.deposit();
254
255         Account a1 = new SavingsAccount();
256         a1.deposit();
257
258         Account a2 = new CurrentAccount();
259         a2.deposit();
260     }
261 }
262 }

```

<terminated> BankingDetails [Java Application] C:\Users\Shree\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86\_64\_17.0.9.v20231028

```
Enter deposit Amount for BankAccount:
10000
Your amount is deposited successfully: 10000
Enter choice(1)(2)for checking calculateInterest/withdrwal:
2
Enter withdrwal Amount for BankAccount:
3000
Your amount withdrwal is successfully done: 3000
Enter choice(1)for checking viewBalances:
1
Your BankAccount balance is: 7000
Enter choice(1)(0)for checking calculateInterest/exit the BankAccount :
0
Invalid number restart please..
Enter deposit Amount for SavingsAccount:
500000
Your amount is deposited successfully: 500000
Enter choice(1)(2)for checking calculateInterest/withdrwal:
1
Enter Principal amount (the beginning balance):
500000
Enter R = Interest rate (usually per year, expressed as a decimal):
5.5
Enter T = Number of time periods (generally one-year time periods):
5
Your simple intrest is: 137500.0
Enter choice(1)for checking viewBalances:
1
Your SavingsAccount balance is: 500000
Enter choice(1)(0)for checking calculateInterest/exit the SavingsAccount :
0
Invalid number restart please..
Enter deposit Amount for CurrentAccount:
4000
Your amount is deposited successfully: 4000
Enter choice(1)(2)for checking calculateInterest/withdrwal:
2
```

<terminated> BankingDetails [Java Application] C:\Users\Shree\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86\_64\_17.0.9.v20231028-0858

Your BankAccount balance is: 7000

Enter choice(1) (0)for checking calculateInterest/exit the BankAccount :

0

Invalid number restart please..

Enter deposit Amount for SavingsAccount:

500000

Your amount is deposited successfully: 500000

Enter choice(1) (2)for checking calculateInterest/withdrwal:

1

Enter Principal amount (the beginning balance):

500000

Enter R = Interest rate (usually per year, expressed as a decimal):

5.5

Enter T = Number of time periods (generally one-year time periods):

5

Your simple intrest is: 137500.0

Enter choice(1)for checking viewBalances:

1

Your SavingsAccount balance is: 500000

Enter choice(1) (0)for checking calculateInterest/exit the SavingsAccount :

0

Invalid number restart please..

Enter deposit Amount for CurrentAccount:

4000

Your amount is deposited successfully: 4000

Enter choice(1) (2)for checking calculateInterest/withdrwal:

2

Enter withdrwal Amount for CurrentAccount:

100

Your amount withdrwal is successfully done: 100

Enter choice(1)for checking viewBalances:

1

Your CurrentAccount balance is: 3900

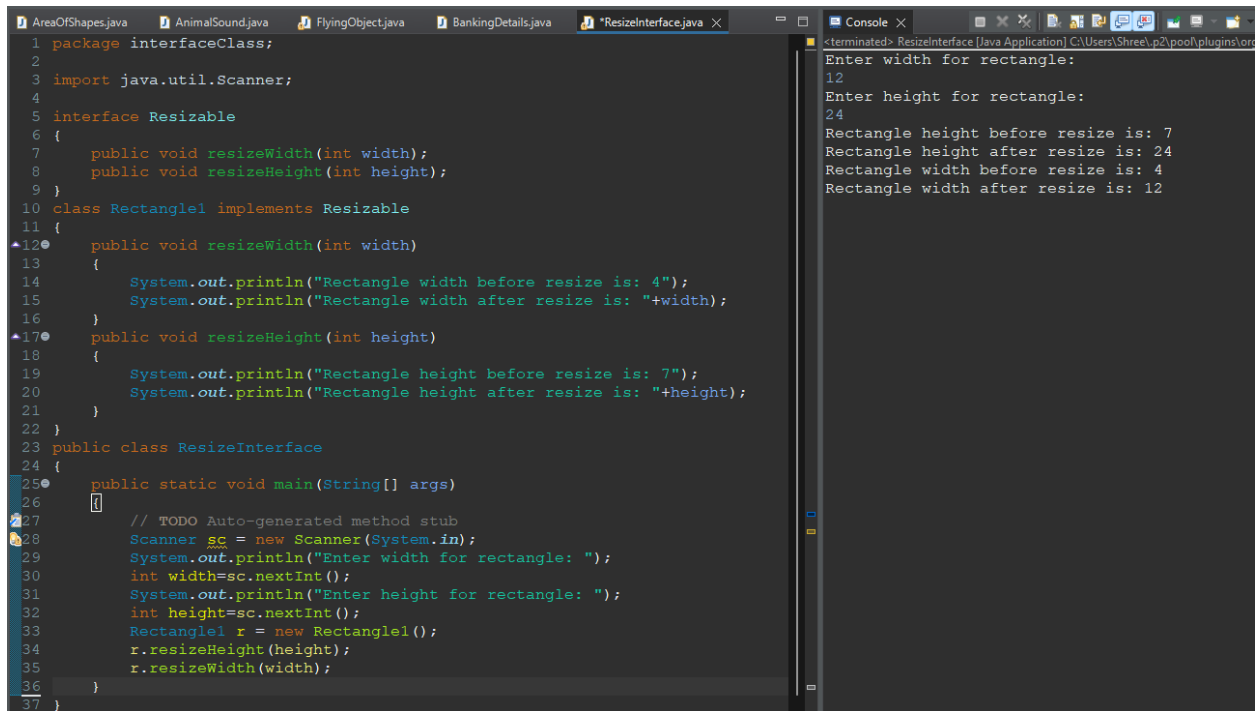
Enter choice(1) (0)for checking calculateInterest/exit the CurrentAccount :

0

Invalid number restart please..



5. Write a Java program to create an interface **Resizable** with methods **resizeWidth(int width)** and **resizeHeight(int height)** that allow an object to be resized. Create a class **Rectangle** that implements the **Resizable** interface and implements the resize methods.

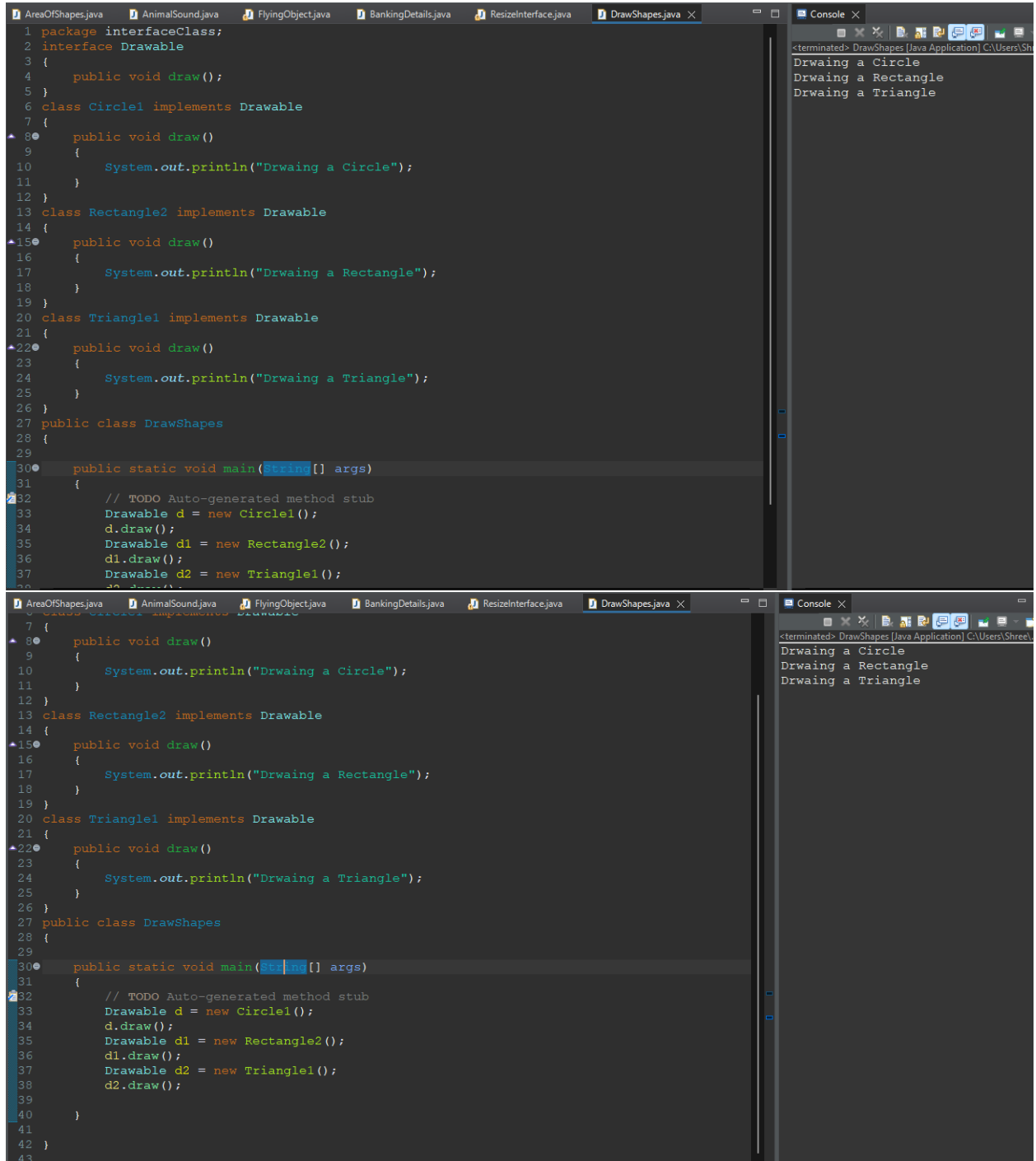


```
1 package interfaceClass;
2
3 import java.util.Scanner;
4
5 interface Resizable
6 {
7     public void resizeWidth(int width);
8     public void resizeHeight(int height);
9 }
10 class Rectangle1 implements Resizable
11 {
12     public void resizeWidth(int width)
13     {
14         System.out.println("Rectangle width before resize is: 4");
15         System.out.println("Rectangle width after resize is: "+width);
16     }
17     public void resizeHeight(int height)
18     {
19         System.out.println("Rectangle height before resize is: 7");
20         System.out.println("Rectangle height after resize is: "+height);
21     }
22 }
23 public class ResizeInterface
24 {
25     public static void main(String[] args)
26     {
27         // TODO Auto-generated method stub
28         Scanner sc = new Scanner(System.in);
29         System.out.println("Enter width for rectangle: ");
30         int width=sc.nextInt();
31         System.out.println("Enter height for rectangle: ");
32         int height=sc.nextInt();
33         Rectangle1 r = new Rectangle1();
34         r.resizeHeight(height);
35         r.resizeWidth(width);
36     }
37 }
```

Console Output:

```
<terminated> ResizeInterface [Java Application] C:\Users\Shree\p2\poo\plugins\org
Enter width for rectangle:
12
Enter height for rectangle:
24
Rectangle height before resize is: 7
Rectangle height after resize is: 24
Rectangle width before resize is: 4
Rectangle width after resize is: 12
```

6. Write a Java program to create an interface **Drawable** with a method **draw()** that takes no arguments and returns void. Create three classes **Circle**, **Rectangle**, and **Triangle** that implement the **Drawable** interface and override the **draw()** method to draw their respective shapes.



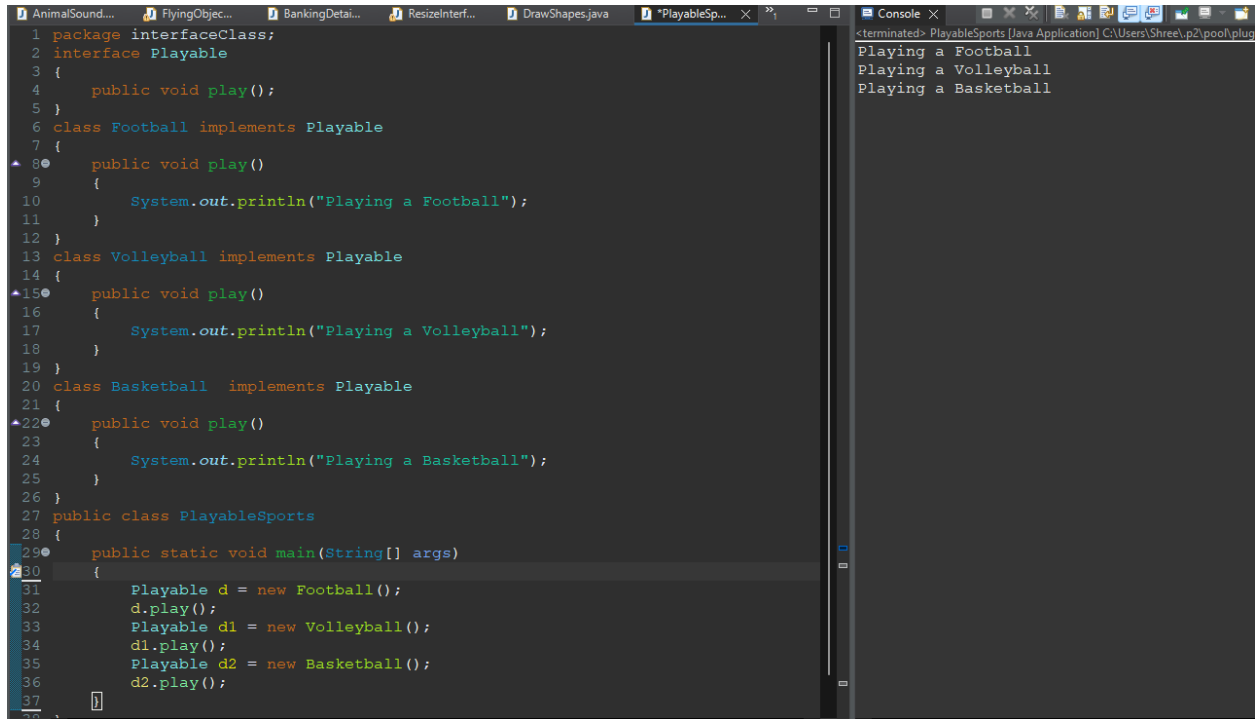
The screenshot displays an IDE with two tabs: `DrawShapes.java` and `Console`. The `DrawShapes.java` tab shows the following code:

```
1 package interfaceClass;
2 interface Drawable
3 {
4     public void draw();
5 }
6 class Circle1 implements Drawable
7 {
8     public void draw()
9     {
10         System.out.println("Drwaing a Circle");
11     }
12 }
13 class Rectangle2 implements Drawable
14 {
15     public void draw()
16     {
17         System.out.println("Drwaing a Rectangle");
18     }
19 }
20 class Triangle1 implements Drawable
21 {
22     public void draw()
23     {
24         System.out.println("Drwaing a Triangle");
25     }
26 }
27 public class DrawShapes
28 {
29
30     public static void main(String[] args)
31     {
32         // TODO Auto-generated method stub
33         Drawable d = new Circle1();
34         d.draw();
35         Drawable d1 = new Rectangle2();
36         d1.draw();
37         Drawable d2 = new Triangle1();
38         d2.draw();
39     }
40 }
41
42 }
43
```

The `Console` tab shows the output of the program:

```
<terminated> DrawShapes [Java Application] C:\Users\Shree\Sh...
Drwaing a Circle
Drwaing a Rectangle
Drwaing a Triangle
```

7. Write a Java program to create an interface Playable with a method play() that takes no arguments and returns void. Create three classes Football, Volleyball, and Basketball that implement the Playable interface and override the play() method to play the respective sports.



The screenshot shows an IDE with a Java file named "PlayableSp...". The code defines an interface "Playable" with a "play()" method. Three classes, "Football", "Volleyball", and "Basketball", implement this interface by overriding the "play()" method to print their respective names. A "PlayableSports" class contains a "main" method that creates instances of these three classes and calls their "play()" methods. The console on the right shows the output: "Playing a Football", "Playing a Volleyball", and "Playing a Basketball".

```
1 package interfaceClass;
2 interface Playable
3 {
4     public void play();
5 }
6 class Football implements Playable
7 {
8     public void play()
9     {
10         System.out.println("Playing a Football");
11     }
12 }
13 class Volleyball implements Playable
14 {
15     public void play()
16     {
17         System.out.println("Playing a Volleyball");
18     }
19 }
20 class Basketball implements Playable
21 {
22     public void play()
23     {
24         System.out.println("Playing a Basketball");
25     }
26 }
27 public class PlayableSports
28 {
29     public static void main(String[] args)
30     {
31         Playable d = new Football();
32         d.play();
33         Playable d1 = new Volleyball();
34         d1.play();
35         Playable d2 = new Basketball();
36         d2.play();
37     }
38 }
```

Console Output:

```
<terminated> PlayableSports [Java Application] C:\Users\Shree\p2\pool\plug
Playing a Football
Playing a Volleyball
Playing a Basketball
```