CSE 1007- Java Programming

Lab Assignment-2

Name: Akshata A Bhat
Register No: 20BCE0923
Date: 20-09-2021

1. Write a Java program to create an abstract class named Shape that contains empty methods named numberOfSides( ) and calculateArea(). Derive classes named Triangle, Square, Pentagon, Hexagon, and Octagon from Shape. Each one of the derived classes contains the method definitions that shows the number of sides and the area calculations in the given geometrical figures. [Hint: Use Runtime polymorphism and refer the following images for area calculations]

Code:

```
abstract class Problem
{
   abstract int numberOfSides();
   abstract double calculateArea();

}
class Square extends Problem
{
   private static int sides=4;
   private static int l=4;
   int area=(l*l);
   int numberOfSides()
   {
      return sides;
   }
   double calculateArea()
   {
      return area;
   }
      public String toString()
   {
      return "Square";
   }
}

 class Triangle extends Problem
{
   private static int sides=3;
   float calculateAreaArea(float a, float b, float c)
   {
      if (a < 0 || b < 0 || c <0 || (a+b <= c) ||
          a+c <=b || b+c <=a)
```

```
        {
            System.out.println("Not a valid triangle");
            System.exit(0);
        }
        float s = (a+b+c)/2;
        return (float)Math.sqrt(s*(s-a)*(s-b)*(s-c));
    }
    int numberOfSides()
    {
        return sides;
    }
    public String toString()
    {
        return "Triangle";
    }
}
class Hexagon extends Problem
{
    private static int sides = 6;
    private static int a=4;
    private static double area=((a*a)*2.6);
    int numberOfSides()
    {
        return sides;
    }
    double calculateArea()
    {
        return area;
    }
    public String toString()
    {
        return "Hexagon";
    }
}
class Octagon extends Problem
{
    private static int sides = 8;
    private static int b=4;
    private static double area=((b*b)*4.8);
    int numberOfSides()
    {
        return sides;
    }
    double calculateArea()
    {
        return area;
    }
    public String toString()
    {
        return "Octagon";
```

```
    }
}
class Pentagon extends Problem
{
    private static int sides = 5;
    private static int c=5;
    private static double area=((c*c)*1.71);
    int numberOfSides()
    {
        return sides;
    }
    double calculateArea()
    {
        return area;
    }
    public String toString()
    {
        return "Pentagon";
    }
}

public class Akshata
{
    public static void main(String args[])
    {
    Problem[] shapes = new Problem[5];

    Square sq = new Square();
    Triangle tr = new Triangle();
    Hexagon hx = new Hexagon();
    Octagon oc=new Octagon();
    Pentagon pn=new Pentagon();
    shapes[0]=sq;
    shapes[1]=tr;
    shapes[2]=hx;
    shapes[3]=oc;
    shapes[4]=pn;
    for(int i=0 ; i<5 ;i++)
    {
        System.out.println(shapes[i].toString()+" sides:"+shapes[i].numberOfSides());
        System.out.println(shapes[i].toString()+" areas:"+shapes[i].calculateArea());
    }


    }
}
```

Output:

Square:4
Triangle3
Hexagon:6
Octagon: 8
Pentagon:5

Square:(Area)
Triangle:(Area)
Hexagon:(Area)
Octagon: (Area)
Pentagon:(Area)

Akshata.java

```java
abstract class Problem
{
    abstract int numberOfSides();
    abstract double calculateArea();

}
class Square extends Problem
{
    private static int sides=4;
    private static int l=4;
    int area=(l*l);
    int numberOfSides()
    {
        return sides;
    }
    double calculateArea()
    {
        return area;
    }
        public String toString()
    {
        return "Square";
    }
}

 class Triangle extends Problem
{
    private static int sides=3;
    float calculateAreaArea(float a, float b, float c)
    {
```

Akshata.java

```
 class Triangle extends Problem
{
    private static int sides=3;
    float calculateAreaArea(float a, float b, float c)
    {
        if (a < 0 || b < 0 || c <0 || (a+b <= c) ||
            a+c <=b || b+c <=a)
        {
            System.out.println("Not a valid triangle");
            System.exit(0);
        }
        float s = (a+b+c)/2;
        return (float)Math.sqrt(s*(s-a)*(s-b)*(s-c));
    }
    int numberOfSides()
    {
        return sides;
    }
    public String toString()
    {
        return "Triangle";
    }
}
class Hexagon extends Problem
{
    private static int sides = 6;
    private static int a=4;
    private static double area=((a*a)*2.6);
    int numberOfSides()
    {
```

Akshata.java

```
58      double calculateArea()
59      {
60          return area;
61      }
62      public String toString()
63      {
64          return "Hexagon";
65      }
66  }
67  class Octagon extends Problem
68  {
69      private static int sides = 8;
70      private static int b=4;
71      private static double area=((b*b)*4.8);
72      int numberOfSides()
73      {
74          return sides;
75      }
76      double calculateArea()
77      {
78          return area;
79      }
80      public String toString()
81      {
82          return "Octagon";
83      }
84  }
85  class Pentagon extends Problem
86  {
87      private static int sides = 5;
```

Akshata.java

```
100         return "Pentagon";
101     }
102 }
103
104 public class Akshata
105 {
106     public static void main(String args[])
107     {
108     Problem[] shapes = new Problem[5];
109
110     Square sq = new Square();
111     Triangle tr = new Triangle();
112     Hexagon hx = new Hexagon();
113     Octagon oc=new Octagon();
114     Pentagon pn=new Pentagon();
115     shapes[0]=sq;
116     shapes[1]=tr;
117     shapes[2]=hx;
118     shapes[3]=oc;
119     shapes[4]=pn;
120     for(int i=0 ; i<5 ;i++)
121     {
122         System.out.println(shapes[i].toString()+" sides:"+shapes[i].numberOfSides());
123         System.out.println(shapes[i].toString()+" areas:"+shapes[i].calculateArea());
124     }
125
126
127     }
128 }
129
```

input stderr

Compilation failed due to following error(s).

```
Akshata.java:26: error: Triangle is not abstract and does not override abstract method calculateArea() in Problem
 class Triangle extends Problem
 ^
1 error
```

---

2. Design a java interface ‘ArrayInterface’ with appropriate methods to read, sort, and to display all elements in the array. Create a class ‘ArrayClass’ by implementing the ArrayInterface for an integer array of ‘n’ integers. Write a Java program to test them.

Code:

```
import java.util.*;
interface ArrayInterface{
    void read_sort();

}
class Read_Sort extends ArrayInterface{
```

```
    public void read_sort(int n){
        int temp;
        Scanner s = new Scanner(System.in);
        System.out.print("Enter no. of elements you want in array:");
        n = s.nextInt();
        int a[] = new int[n];
        System.out.println("Enter all the elements:");
        for (int i = 0; i < n; i++)
        {
            a[i] = s.nextInt();
        }
        for (int i = 0; i < n; i++)
        {
            for (int j = i + 1; j < n; j++)
            {
                if (a[i] > a[j])
                {
                    temp = a[i];
                    a[i] = a[j];
                    a[j] = temp;
                }
            }
        }
        System.out.print("Ascending Order:");
        for (int i = 0; i < n - 1; i++)
        {
            System.out.print(a[i] + ",");
        }
        System.out.print(a[n - 1]);
    }
}


public class Main
{
        public static void main(String[] args) {
    Read_Sort r1=new Read_Sort();
    r1.read_sort(3);

}
}
```

Input:

Enter no. of elements you want in array:4

Enter all the elements:
7
6
3
4
Ascending Order:3,4,7,6

Output:

Akshata.java

```java
import java.util.*;
interface ArrayInterface{
    void read_sort();

}
class Read_Sort extends ArrayInterface{
    public void read_sort(int n){
        int temp;
        Scanner s = new Scanner(System.in);
        System.out.print("Enter no. of elements you want in array:");
        n = s.nextInt();
        int a[] = new int[n];
        System.out.println("Enter all the elements:");
        for (int i = 0; i < n; i++)
        {
            a[i] = s.nextInt();
        }
        for (int i = 0; i < n; i++)
        {
            for (int j = i + 1; j < n; j++)
            {
                if (a[i] > a[j])
                {
                    temp = a[i];
                    a[i] = a[j];
                    a[j] = temp;
                }
            }
        }
        System.out.print("Ascending Order:");
```

Akshata.java

```
19          {
20              for (int j = i + 1; j < n; j++)
21              {
22                  if (a[i] > a[j])
23                  {
24                      temp = a[i];
25                      a[i] = a[j];
26                      a[j] = temp;
27                  }
28              }
29          }
30          System.out.print("Ascending Order:");
31          for (int i = 0; i < n - 1; i++)
32          {
33              System.out.print(a[i] + ",");
34          }
35          System.out.print(a[n - 1]);
36      }
37  }
38
39
40  public class Akshata
41  {
42      public static void main(String[] args) {
43      Read_Sort r1=new Read_Sort();
44      r1.read_sort(3);
45
46  }
47  }
48
```

input

Compilation failed due to following error(s).

```
Akshata.java:6: error: no interface expected here
class Read_Sort extends ArrayInterface{
                        ^
1 error
```