**EX. NO: 7b ABSTRACT CLASSES & INTERFACES**

**(15/3/2017-19/3/2017)**

**Note: Part I and Part 2 should be recorded in your observation as directed by your mentor**

**Part-I (Who should do Part-I?)**

***Anyone who wants to clear java, hope everyone wants to…….***

**1. Abstract Class:**

Define an abstract class called **BasicShape** with the following members:

***Private Member Variable:***

area, a double used to hold the shape’s area.

***Public Methods:***

getArea - This concrete method should return the value in the member variable area.

calcArea - This method should be an abstract method.

Next, define a class named **Circle**. It should be derived from the **BasicShape** class.

It should have the following members:

***Private Member Variables:***

centerX, a long integer used to hold the x coordinate of the circle’s center.

centerY, a long integer used to hold the y coordinate of the circle’s center.

radius, a double used to hold the circle’s radius.

***Public Methods:***

constructor—accepts values for centerX, centerY, and radius. Should call the overridden

calcArea function described below.

getCenterX—returns the value in centerX.

getCenterY—returns the value in centerY.

calcArea—calculates the area of the circle (area = 3.14159 \* radius \* radius) and stores

the result in the inherited member area.

Next, define a class named **Rectangle**. It should be derived from the **BasicShape** class. It should have the following members:

***Private Member Variables:***

width, a long integer used to hold the width of the rectangle.

length, a long integer used to hold the length of the rectangle.

***Public Methods:***

constructor—accepts values for width and length. Should call the overridden calcArea

function described below.

getWidth—returns the value in width.

getLength—returns the value in length.

calcArea—calculates the area of the rectangle (area = length \* width) and stores the result

in the inherited member area.

After creating these classes, create a driver program that defines a Circle object and a Rectangle object. Demonstrate that each object properly calculates and reports its area.

**2.** Define an interface named **TVStation** with the following details:

***Data Members:***

satelliteName – String

cableTVName – String

signalFrequency – double

***Methods:***

show()- abstract method

Define another class named **Programme**, as shown below:

***Data Members:***

programmeName – String

sponsor - String

***Methods:***

Constructor

display()

Define a class named **Broadcast** that implements the interface **TVStation** and extends the class **Programme**.

Create an object for the Broadcast class and invoke the methods

3. Define an abstract class named **Shape** that contains an empty method named **numberOfSides()**. Define three classes named **Trapezoid, Triangle** and **Hexagon** such that each one of the classes extends the class **Shap**e. Each one of the classes contains only the method **numberOfSides()** that shows the number of sides in the given geometrical figure.

Demonstrate how the Trapezoid, Triangle and Hexagon classes can be instantiated and their methods can be tested.

**Part-II: MCQ (Who should do Part-II?)**

***Who will not love fun? If you love fun go ahead …and you can take this fun back home and continue there as well……………….***

1. Given these two definitions

interface I1 {}

interface I2 {}

which one of the following will compile without errors?

a) interface II implements I1, I2 {}

b) interface II implements I1 implements I2 {}

c) interface II implements I1 extends I2 {}

d) interface II extends I1, I2 {}

2. Given these three definitions

interface I1 {}

interface I2 {}

abstract class C {}

which one of the following will compile without errors?

a) class CI12 extends C, I1, I2 {}

b) class CI12 implements C extends I1, I2 {}

c) class CI12 implements C, I1, I2 {}

d) class CI12 extends C implements I1, I2 {}

e) class CI12 extends C implements I1 implements I2 {}

f) class CI12 implements C extends I1 extends I2 {}

3. Consider the following program:

interface Side { String getSide(); }

class Head implements Side {

public String getSide() { return "Head "; }

}

class Tail implements Side {

public String getSide() { return "Tail "; }

}

class Coin {

public static void overload(Head side) { System.out.print(side.getSide()); }

public static void overload(Tail side) { System.out.print(side.getSide()); }

public static void overload(Side side) { System.out.print("Side "); }

public static void overload(Object side) { System.out.print("Object "); }

public static void main(String []args) {

Side firstAttempt = new Head();

Tail secondAttempt = new Tail();

overload(firstAttempt);

overload((Object)firstAttempt);

overload(secondAttempt);

overload((Side)secondAttempt);

}

}

What is the output of this program when executed?

a) Head Head Tail Tail

b) Side Object Tail Side

c) Head Object Tail Side

d) Side Head Tail Side

4. Consider the following program:

abstract class AbstractBook {

public String name;

}

interface Sleepy {

public String name = "undefined";

}

class Book extends AbstractBook implements Sleepy {

public Book(String name) {

this.name = name; // LINE A

}

public static void main(String []args) {

AbstractBook philosophyBook = new Book("Principia Mathematica");

System.out.println("The name of the book is " + philosophyBook.name); // LINE B

}

}

Which one of the following options correctly describes the behavior of this program?

a) The program will print the output “The name of the book is Principia Mathematica”.

b) The program will print the output “The name of the book is undefined”.

c) The program will not compile and result in a compiler error “ambiguous reference to name” in line marked with

comment LINE A.

d) The program will not compile and result in a compiler error “ambiguous reference to name” in line marked with

comment LINE B.

**Part-III (Who should do Part-III?)**

***If you are a person who loves to challenge yourself, train yourself till you tire and in short for those who aspire to become extra intellect, this is for you***

1. Extend or Modify your Bank Application to incorporate the appropriate use Abstract classes and Interfaces.