**Lab 4:: 09.092020**

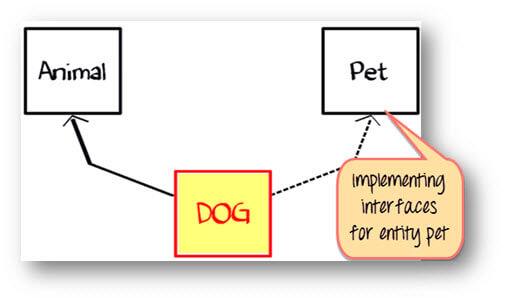
**Java Programming**

**Write a Java program to do the following.**

1. Define an abstract class by name Solid that has the data member radius of double type with default scope. The class Solid has three public abstract methods by name surfaceArea(), volume(), readRadius(). The class Solid also has a concrete method by name baseArea that computes πr2 by receiving r as an argument which is double type.
2. Two classes by name Cylinder and Sphere inherit the Solid class and implement the methods readRadius(), surfaceArea() and volume(). readRadius() accepts the radius, surfaceArea()computes surface area and volume() computes the volume of the concerned shape. Both the classes should make use of the baseArea method of the abstract class wherever possible.
3. Define another class by name MySolid with the main method that displays the surface area and volume of a cylinder and a sphere.

[Hint: surface area and volume of a cylinder are 2πr2+2πrh and πr2h respectively. surface area and volume of a sphere are 4πr2 and 4/3πr3 respectively.]

**Interfaces**



1. Class Dog can extend to class "Animal" and implement interface as "Pet".
2. Create an interface ILoan with homeLoan() and vehicleLoan() as method signature. Implement this interface in BankLoan class which is an abstract class with an abstract method calcInterst(). Derive two classes HomeLoan and VechicleLoan from the base class. A person is eligible for a home loan amount of Rs.20,00,000 if his income is above Rs. 50,000 with an interest rate of 9.5%. A person is eligible for a vehicle loan of 12,00,000 if his income is above Rs. 60,000 with an interest rate of 10.5%. Write a java program to demonstrate runtime polymorphism for the above scenario with an array of objects.
3. Consider an interface Shape with computeArea() as method signature. Implement this in an abstract class Shape with compCircumference() as abstract method. Derive Square, Rectangle and Circle classes from Shape class and override computeArea() member function. Create an array of Shape class objects and demonstrate runtime polymorphism.

**Exception Handling**

* Develop a Java program that demonstrates ArithmeticException and ArrayIndexOutOfBoundsException. (Both the exceptions should be handled in the same program using a single array.)
* Develop a Java program that accepts 10 integers in an array and sort them in ascending order (Use any sorting algorithm). The program should handle two exceptions namely NumberFormatException (or InputMismatchException) and ArrayIndexOutOfBoundsException. NumberFormatException is to be handled when you accept the contents of the array and ArrayIndexOutofBoundsException when you display the contents of the sorted array.
* Write a program that illustrates rethrowing an exception. Define methods someMethod and someMethod2. Method someMethod2 should initially throw an exception. Method someMethod should call someMethod2, catch the exception and rethrow it. Call someMethod from method main, and catch the rethrown exception. Print the stack trace of this exception.
* Use inheritance to create an exception superclass (called ExceptionA) and exception subclasses ExceptionB and ExceptionC, where ExceptionB inherits from ExceptionA and ExceptionC inherits from ExceptionB. Write a program to demonstrate that the catch block for type ExceptionA catches exceptions of types ExceptionB and ExceptionC.
* ***(Catching Exceptions Using Class Exception)*** Write a program that demonstrates how various exceptions are caught with
  + catch (Exception exception )

This time, define classes ExceptionA (which inherits from class Exception) and ExceptionB (which inherits from class ExceptionA). In your program, create try blocks that throw exceptions of types ExceptionA, ExceptionB, NullPointerException and IOException. All exceptions should be caught with catch blocks specifying type Exception.