# Assignment: 5

## Problem Statement:

Design and develop a context for given case study and implement an interface for Vehicles Consider the example of vehicles like bicycle, car, and bike. All Vehicles have common functionalities such as Gear Change, Speed up and apply breaks. Make an interface and put all these common functionalities. Bicycle, Bike, Car classes should be implemented for all these functionalities in their own class in their own way.

## Aim :

To understand Interface in Java

## Theory

An interface in Java is a blueprint of a class. It has static constants and abstract methods.The interface in Java is a mechanism to achieve abstraction. There can be only abstract methods in the Java interface, not method body. It is used to achieve abstraction and multiple inheritance in Java.A programmer uses an abstract class when there are some common features shared by all the objects

1. programmer writes an interface when all the features have different implementations for different objects Interfaces are written when the programmer wants to leave the implementation to third party vendors An interface is a specification of method prototypes.

All the methods in an interface are abstract methods

* 1. An interface is a specification of method prototypes
  2. An interface contains zero or more abstract methods
  3. All the methods of interface are public, abstract by default
  4. An interface may contain variables which are by default public static final
  5. Once an interface is written any third party vendor can implement it
  6. All the methods of the interface should be implemented in its implementation classes
  7. If any one of the method is not implemented, then that implementation class should be declared as abstract
  8. We cannot create an object to an interface
  9. We can create a reference variable to an interface
  10. An interface cannot implement another interface
  11. An interface can extend another interface
  12. A class can implement multiple interfaces

## Syntax:

interface <interface\_name>{

// declare constant fields

// declare methods that abstract

// by default.

}

1. Java Interface Example

In this example, the Printable interface has only one method, and its implementation is provided in the A6 class.

interface printable{ void print();

}

class A6 implements printable{ public void print()

{

System.out.println("Hello");

}

public static void main(String args[]){ A6 obj = new A6();

obj.print();

}

}

1. Write an example program for interface

Interface Shape

{

void area (); void volume (); double pi = 314;

}

class Circle implements Shape

{

double r;

Circle (double radius)

{r = radius; } public void area ()

{

System.out.println ("Area of a circle is : " + pi\*r\*r );

}

public void volume ()

{

System.out.println ("Volume of a circle is : " + 2\*pi\*r);

}

}

class Rectangle implements Shape

{

double l,b;

Rectangle (double length, double breadth)

{

l = length; b = breadth;.

}

public void area ()

{

System.out.println ("Area of a Rectangle is : " + l\*b );

}

public void volume ()

{

System.out.println ("Volume of a Rectangle is : " + 2\*(l+b));

}

}

class InterfaceDemo

{

public static void main (String args[])

{

Circle ob1 = new Circle (102); ob1.area ();

ob1.volume ();

Rectangle ob2 = new Rectangle (126, 2355); ob2.area ();

ob2.volume ();

}

}

Interface vs Abstract Class

An interface is like having a 100% Abstract Class. Interfaces cannot have non-abstract Methods while abstract Classes can. A Class can implement more than one Interface while it can extend only one Class. As abstract Classes come in the hierarchy of Classes, they can extend other Classes while Interface can only extend Interfaces.

The relationship between classes and interfaces

As shown in the figure given below, a class extends another class, an interface extends another interface, but a class implements an interface.

Algorithm:

1 Create a vehicle class and initialize variables

2 Implement Bicycle class with vehicle class

3 Implement Bikers class with vehicle class

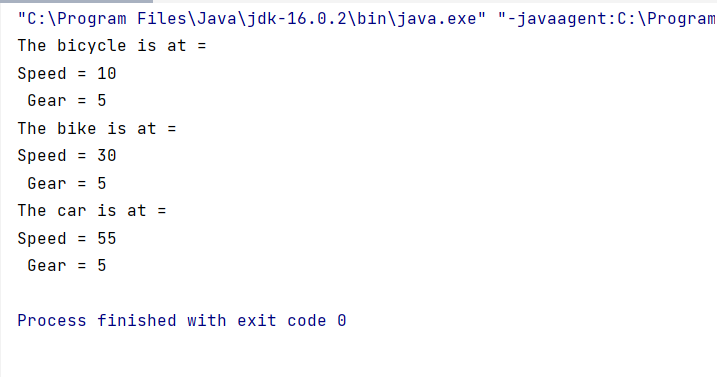
4 Implement cars class with vehicle class

5 And then in the main class call all the function.

**CODE:**

package com.company;  
  
interface Vehicles1{  
 void gearC(int a);  
 void speedU(int b);  
 void ApplyB(int c);  
}  
  
class Bicycles implements Vehicles1{  
 int speed ;  
 int gear ;  
 public void gearC(int newGear){  
 gear = newGear;  
  
 }  
  
 public void speedU(int increm){  
 speed = speed + increm;  
  
 }  
 public void ApplyB(int decrem){  
 speed = speed - decrem;  
 }  
  
 public void print(){  
 System.*out*.println("Speed = "+speed+"\n Gear = "+gear);  
 }  
}  
  
  
class Biker implements Vehicles1{  
 int speed ;  
 int gear ;  
 public void gearC(int newGear){  
 gear = newGear;  
  
 }  
  
 public void speedU(int increm){  
 speed = speed + increm;  
  
 }  
 public void ApplyB(int decrem){  
 speed = speed - decrem;  
 }  
  
 public void print(){  
 System.*out*.println("Speed = "+speed+"\n Gear = "+gear);  
 }  
}  
  
  
class Cars implements Vehicles1{  
 int speed ;  
 int gear ;  
 public void gearC(int newGear){  
 gear = newGear;  
  
 }  
  
 public void speedU(int increm){  
 speed = speed + increm;  
  
 }  
 public void ApplyB(int decrem){  
 speed = speed - decrem;  
 }  
  
 public void print(){  
 System.*out*.println("Speed = "+speed+"\n Gear = "+gear);  
 }  
}  
  
public class Assignmnet5 {  
  
 public static void main(String[] args) {  
 Bicycles bs = new Bicycles();  
 bs.gearC(5);  
 bs.speedU(15);  
 bs.ApplyB(5);  
 System.*out*.println("The bicycle is at = ");  
 bs.print();  
  
  
 Biker bi = new Biker();  
 bi.gearC(5);  
 bi.speedU(45);  
 bi.ApplyB(15);  
 System.*out*.println("The bike is at = ");  
 bi.print();  
  
  
 Cars ca = new Cars();  
 ca.gearC(5);  
 ca.speedU(65);  
 ca.ApplyB(10);  
 System.*out*.println("The car is at = ");  
 ca.print();  
  
  
  
 }  
}

**OUTPUT:**

****

**Conclusion:**

Thus, we have studies interface concept using java