**Interface:**

Using the keyword **interface**, you can fully abstract a class interface from its implementation. Interfaces allow you to specify what a class must do, but not how it does it.

Interfaces lack instance variables and their methods are declared without anybody.

Any number of classes can implement an interface. Also, one class can implement any number of interfaces.

By providing the interface keyword, java allows you to fully utilize the **“one interface, multiple methods”** aspect of polymorphism.

**Interface implementation:**

To implement an interface, a class must create the complete set of methods defined by the interface. However, each class is free to determine the details of its own implementation.

**Defining an Interface:**

The general form of the interface is shown below.

access interface name{

return-type method-name1(parameter-list);

return-type method-name2(parameter-list);

type final-varname1=value;

}

**Implementing Interfaces:**

The general form of the class that includes the implements clause looks like this:

Access class classname [extends superclass]

[implements interface [,interface..]]{

//class body

}

**A practical example:**

An example of interface definition is shown below. It declares a simple interface which contains one method called callback() that takes a single integer parameter.

**Example 1:**

//defining an interface

**public** **interface** Callback {

**void** callback(**int** param);

}

//implementing interfaces

**public** **class** C\_callback **implements** Callback {

**public** **void** callback(**int** p)

{

System.*out*.println("callback called with =" +p);

}

**void** nonIfaceMeth()

{

System.*out*.println("classes that implement interfaces " + "may also define other members, too. ");

}}

//calling method via interface reference variable

**public** **class** TestIface {

**public** **static** **void** main(String[] args)

{

Callback c= **new** C\_callback();

c.callback(42);}}

**Output:**



**Example 2**

**public** **class** AC\_callback **implements** Callback{

**public** **void** callback(**int** p)

{ System.*out*.println("Another version of callback");

System.*out*.println("p squared is =" +(p\*p));

}}

//calling method using interface variable

**public** **class** AC\_callbackDemo {

**public** **static** **void** main(String args[])

{

//AC\_callback ac=new AC\_callback();

Callback ac=**new** AC\_callback();

ac.callback(25);

}}

**Output:**

Another version of callback

p squared is =625

**A class implementing two interfaces:**

**Aim: To write a Java program to show how a class implements two interfaces**

**Example 3:**

**public** **interface** one\_1 {

**int** *x*=12;

}

**public** **interface** two\_2 {

**int** *y*=10;

**void** display();

}

**public** **class** Demo\_1\_2 **implements** one\_1,two\_2{

**public** **void** display(){

System.*out*.println("X =" +*x*);

System.*out*.println("Y =" +*y*);

System.*out*.println("X+Y =" +(*x*+*y*));

} }

**public** **class** c\_one\_two {

**public** **static** **void** main(String args[])

{

Demo\_1\_2 d=**new** Demo\_1\_2();

d.display();

}

}

**Output:**



**Interfaces can be extended:**

**Example\_4:**

//One interface can extend another

**public** **interface** A {

**void** meth1();

**void** meth2();

}

//B now includes meth1() and meth2() -- it adds meth3().

**public** **interface** B **extends** A{

**void** meth3();

}

//This class implements all of A and B

**public** **class** C\_A\_B **implements** B{

**public** **void** meth1()

{

System.*out*.println("Implement meth1().");

}

**public** **void** meth2()

{

System.*out*.println("Implement meth2().");

}

**public** **void** meth3()

{

System.*out*.println("Implement meth3().");

}

}

//Main class

**public** **class** C\_ab\_main {

**public** **static** **void** main(String args[])

{

C\_A\_B cab=**new** C\_A\_B();

cab.meth1();

cab.meth2();

cab.meth3();

}

}

**Output:**



**Multiple Inheritance through interface in java**

Syntax

public interface **A**{

*//Do Something*

}

public interface **B** extends **A**{

*//Do Something*

}

public interface **C** extends **A**{

*//Do Something*

}

**Example\_ 5:**  
interface **vehicleone**{

int speed=90;

public void **distance**();

}

interface **vehicletwo**{

int distance=100;

public void **speed**();

}

class **Vehicle** implements **vehicleone**,**vehicletwo**{

public void **distance**(){

int distance=speed\*100;

System.out.println("distance travelled is "+distance);

}

public void **speed**(){

int speed=distance/100;

}

}

class **MultipleInheritanceUsingInterface**{

public static void **main**(String args[]){

System.out.println("Vehicle");

obj.distance();

obj.speed();

}

}