Page No :
Date :

Yash Mahajan SE IT B 04

A) Extended Theory

1) Explain how to implement multiple inheritance in Java with syntax and example.

Multiple Inheritance is a feature of object oriented concept where a class can inherit properties of more than one parent class.
Java does not support multiple inheritance, however a class can implement one or more interface which has helped Java get rid of ~~interface~~ impossibility of multiple inheritance.

Eg) class A { }
class B { }
class C extends A, B { }

This inheritance using multiple class is wrong as two class cannot be inherited.

To achieve multiple inheritance, the correct way is extending only one class or implementing two or more interfaces

Eg:- interface A { }
interface B { }
class C implements ~~interface~~ A, B { }

Page No :
Date :

"Dash Mahajan SE IT B 04.

2) Differentiate between abstract class and interface.

Abstract Class	Interface.
1) Abstract class can have abstract & non-abstract methods	1) Interface can have only abstract methods.
2) Doesn't support multiple inheritance	2) supports multiple inheritance
3) Abstract class can have final, non-final, static, non static variable	3) Interface can only have static and final variable.
4) The abstract keyword is used to declare an abstract class.	4) The interface keyword is used to declare an interface.
5) A Java abstract class can have class members of all visibility access.	5) Members of Java interface are public by default.

Yash Mahajan SE IT B 04

D) Conclusion :-

In this experiment we have studied the concept about interfaces. To understand how interfaces are implemented we have written programs in which classes implement interfaces.

An interface in Java programming language is an abstract type that is used to specify a behaviour that classes must implement. One important advantage of using interfaces is that we can implement multiple inheritance.

Interfaces help us achieve total abstraction. It also helps us to achieve loose coupling. They also enable us to implement multiple inheritance.