Assignment No. 4
Assignment Name : Inheritance and Interface

Prerequisites:-

Constructor/ Destructor

Concept of Overloading and overriding

Inheritance

Inheritance is one of the feature of Object Oriented Programming System (OOPs) it allows the child class to acquire

the properties (the data members) and functionality (the member functions) of parent class.

What is child class?

A class that inherits another class is known as child class, it is also known as derived class or subclass.

What is parent class?

The class that is being inherited by other class is known as parent class, super class or base class.

Types of Inheritance

Access in subclass

The following members can be accessed in a subclass:

i) public or protected superclass members.

ii) Members with no specifier if subclass is in same package.

The use of super keyword

i) Invoking superclass constructor - super(arguments)

ii) Accessing superclass members – super.member

iii) Invoking superclass methods – super.method(arguments)

Example:

```
public class A
{
protected int num;
public A(int num)
{
this.num = num;
}
}
```

```
public class B extends A
{
private int num;
public B(int a, int b)
{
super(a); //should be the first line in the subclass constructor
this.num = b;
}
public void display()
{
System.out.println("In A, num = " + super.num);
System.out.println("In B, num = " + num);
} }
```

Overriding methods

Redefining superclass methods in a subclass is called overriding. The signature of the subclass method should be the

same as the superclass method.

Example:

```
public class A
{
public void display(int num) {//code}
}
public class B extends A
{
public void display(int x) { //code }
}
```

Dynamic binding

When over-riding is used, the method call is resolved during run-time i.e. depending on the object type, the

corresponding method will be invoked.

Example:

```
A ref;
```

ref = new A();

ref.display(10); //calls method of class A

ref = new B();

ref.display(20); //calls method of class B

Final Keyword in Java

The **final keyword** in java is used to restrict the user. The java final keyword can be used in many context. final

keyword can be for:

1. variable

2. method

3. class

The final keyword can be applied with the variables, a final variable that have no value it is called blank final

variable or uninitialized final variable. It can be initialized in the constructor only. The blank final variable can be

static also which will be initialized in the static block only. We will have detailed learning of these. Let's first learn

the basics of final keyword.

1) Java final variable

If any variable as final, value of final variable cannot change the (It will be constant).

2) Java final method

If any method as final, that method cannot be overridden in subclass.

3) Java final class

If any class as final, then that class cannot be extended ie creating subclass of final class is not allowed. **Abstract class**

An abstract class is a class which cannot be instantiated. It is only used to create subclasses. A class which has

abstract methods must be declared abstract. An abstract class can have data members, constructors, method

definitions and method declarations.

abstract accessspecifier class ClassName

{

...

}

Abstract method

An abstract method is a method which has no definition. The definition is provided by the subclass. abstract

accessspecifier returnType method(arguments);

Interface

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

In other words, you can say that interfaces can have abstract methods and variables. It cannot have a method body.

accessspecifier interface InterfaceName

{

//final variables

//abstract methods

}

Example:

public interface MyInterface

{

int size= 10; //final and static

void method1();

void method2();

}

public Class MyClass **implements** MyInterface

{

//define method1 and method2

}

Lab Assignment

SET A

1. Define a “Point” class having members – x,y(coordinates). Define default constructor and parameterized constructors. Define subclass “ColorPoint” with member as color .Write display method to display the details of different types of Points.

2. Write a Java program to create a super class Employee (members – name, salary). Derive a sub-class as Developer (member – projectname). Derive a sub-class Programmer (member – proglanguage) from Developer. Create object of Developer and display the details of it. Implement this multilevel inheritance with appropriate constructor and methods.

SET B

1. . Write a program for multilevel inheritance such that country is inherited from continent. State is inherited from country. Display the place, state, country and continent.

SET C

1. Define an interface “Operation” which has methods area(),volume(). Define a constant PI having a value 3.142. Create a class circle (member – radius), cylinder (members – radius, height) which implements this interface. Calculate and display the area and volume.

Signature of the instructor: _____ Date: _____

Assignment Evaluation

0: Not Done [ ] 1: Incomplete [ ] 2: Late Complete [ ]

3: Needs Improvement [ ] 4: Complete [ ] 5: Well done [ ]