CORE JAVA

MANUAL V8.3

MODULE CODE:

ANUDIP FOUNDATION


ICONS AND THEIR MEANING
HINTS:
Get ready for helpful insites on difficult topics and questions.
STUDENTS:
This icon symbolize important instrcutions and guides for the students.
TEACHERS/TRAINERS:
This icon symbolize important instrcutions and guides for the trainers.

Module 2: Object Oriented Programming and Package

Chapter 6

Objective: After completing this lesson you will be able to : * Understand the concepts of Java overloading and overriding * Gain knowledge about how the 'this' and 'super' keywords in Java * Learn the concept of dynamic binding in Java	**Materials Required:** 1. Computer 2. Internet access
Theory Duration: 60 minutes	**Practical Duration:** 60 minutes
Total Duration: 120 minutes	

Chapter 6

6.1 Overloading and Overriding

Overloading

Java overloading is the process that enables separate methods to have the same name. It enables programmers to have several methods within the same class. Overloaded methods can have the same names, but their arguments are different. Arguments can vary according to the type or number of parameters. Overloading is related to static (compile time) polymorphism. It can be performed within a class.

Example Method Overloading in Java

```
class OverloadingExample
{
static int add(int a,int b)
{
return a+b;
}
static int add(int a,int b,int c)
{
return a+b+c;
}
}
```

Overriding

Overriding in Java refers to having two methods that have the same arguments, but varying implementations. One of the methods should exist in a parent class while the other is in a child class. Parameters of methods have to be same for overriding. Through overriding, a child class can enable an implementation for a method, for which an implementation is already provided by its parent class. Java uses overriding for achieving runtime polymorphism. It occurs between two classes with an IS-A relationship.

Example Method Overriding in Java

```
class Animal{
void eat( ){System.out.println(‘eating...‘ );
}
}
class Cat extends Animal
{
void eat( )
{
System.out.println(‘eating fish...‘ );
}
}
```

Difference between method overloading and method overriding

Method overloading	Method overriding
It is a type of compile time polymorphism.	It is a type of run time polymorphism.
Method overloading raises programme readability.	It is used for enabling specific method implementation already specified by a parent class
May work with or without inheritance	It always requires inheritance
It can be performed inside a class	It is performed in two classes having an inheritance relationship.
The methods are required to have same names but different signatures.	The methods are required to have same names and same signatures.
Return types have to be different, but parameter must be changed.	Return type has to be the same.

Method overriding real-life example –

Let us consider a child class named Girl and a parent class named Human. The Human class is extended by the Girl

class. Both the parent and child classes have a common method called void walk( ). The Girl class is overriding the

walk( ) method in this example.

As the Girl class overrides the method, the output will show a result for it.

Code example –

```
class Human{
  //Overridden method
  public void walk( )
  {
    System.out.println(‘Human is walking‘ );
  }
}
class Girl extends Human{
  //Overriding method
  public void walk( ){
    System.out.println(‘Girl is walking‘ );
  }
  public static void main( String args[] ) {
    Girl obj = new Girl( );
    //It calls the child class’ s walk( ) method
    obj.walk( );
  }
}
```

Output:

Girl is walking

6.2 Use of this and super keywords

Use of this keyword in Java

‘

This is a Java reference variable used for referring to a current object.

The different uses of 'this' keyword in Java are –

- To refer to the instance variables of the current class
- To initiate or invoke the current class constructor
- To be passed as a form of argument in the method call
- To be passed as an argument within a constructor call
- To return the instance of the current class

Use of super keyword in Java

The Java 'super' keyword is a reference variable utilized for parent class object reference. Upon creating a subclass instance, a parent class is implicitly created and referred by a super keyword.

The different uses of 'super' keyword are

1. To refer immediate instance variable belonging to a parent class
2. To initiate or invoke immediate methods of a parent class
3. To initiate or invoke immediate constructor of a parent class

6.3 Dynamic binding

Binding in Java is the connection established between class and method, and to the connection between class and field. The two types of binding are –

* Static binding

* Dynamic binding

What is dynamic binding?

Dynamic binding is a Java binding method in which a compiler has no say regarding method call. Overriding is the ideal example of this type of binding. Methods are static and the binding occurs in the runtime stage. Dynamic binding utilizes objects for binding.

Dynamic binding example in Java –

```
class Mammal{
void eat(){System.out.println("mammal is sleeping...");}
}
Class Human extends Mammal{
 void eat(){System.out.println("human is sleeping...");}
 public static void main(String args[]){
  Mammal a=new Human();
  a.sleep();
 }
}
```

Output:

human is sleeping...

Practical (60 minutes)

a) See the example programme for Java method overloading below. Write the same programme by assigning values to integers c, d and f and add them using overloading. Show the resulting output.

```
class OverloadingExample
{
static int add(int a,int b)
{
return a+b;
}
static int add(int a,int b,int c)
{
```

```
return a+b+c;

}

}
```

b) See the example programme for Java method overriding below. Write the same programme for to override Animal is eating with dog is eating. Show the resulting output.

```
class Human{

  public void sleep( )

 {

     System.out.println(‘Human is sleeping‘ );

 }}class Girl extends Human{

 public void eat( ){

     System.out.println(‘Girl is sleeping‘ );

 }

 public static void main( String args[] ) {

   Girl obj = new Girl( );

     obj.eat( );

 }}
```

Instructions: The progress of students will be assessed with the exercises mentioned below.

MCQ

1. Java overloading enables separate methods to have the same _____.

a) property

b) name

c) characteristic

d) None of the mentioned

2. Programmers can use Java overloading to have several _____ within the same class.

a) methods

b) classes

c) options

d) None of the mentioned

3. Although overloaded methods have the same name, they have different _____.

a) arguments

b) non-arguments

c) categories

d) None of the mentioned

4. Overloading method arguments vary according to the type of _____.

a) arrays

b) parameters

c) properties

d) None of the mentioned

5. Overloading is related to _____ polymorphism.

a) default

b) dynamic

c) static

d) None of the mentioned

6. In Java overriding two methods have _____ arguments.

a) different

b) same

c) unique

d) None of the mentioned

7. Java overriding is possible when one method is in a parent class and the other is in a _____ class.

a) initial

b) child

c) primary

d) None of the mentioned

8. Overriding can be used to achieve _____ polymorphism.

a) static

b) runtime

c) compile time

d) None of the mentioned

9. A 'this' keyword in Java is used to refer to the instance _____ of the current class.

a) variable

b) code

c) modes

d) None of the mentioned

10. What type of variable is the Java 'super' keyword?

a) inference

b) reference

c) combination

d) None of the mentioned