Object Oriented Programming with Java (Subject Code: BCS-403)

Unit 1

Lecture 9

Lecture 9

- Polymorphism
- Overriding
- Overloading

Polymorphism

- Polymorphism is considered one of the important features of Object-Oriented Programming.
- Polymorphism allows us to perform a single action in different ways. In other words, polymorphism allows you to define one interface and have multiple implementations.
- The word “poly” means many and “morphs” means forms, So it means many forms.

Types of Java Polymorphism

In Java Polymorphism is mainly divided into two types

- Compile-time Polymorphism
- Runtime Polymorphism

Compile-Time Polymorphism in Java

It is also known as static polymorphism. This type of polymorphism is achieved by function overloading or operator overloading.

Note: *Java doesn’t support the Operator Overloading.*

Method Overloading

When there are multiple functions with the same name but different parameters then these functions are said to be overloaded.

Functions can be overloaded by changes in the number of arguments or/and a change in the type of arguments.

```
class  Overloading{
   // Method with 2 integer parameters
   static int Multiply(int a, int b)
   {
      // Returns product of integer numbers
      return a * b;
   }
    // Method 2
   // With same name but with 2 double parameters
   static double Multiply(double a, double b)
   {
      // Returns product of double numbers
      return a * b;
   }
}
```

Java Method Overloading

```
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;
}
}
```

Runtime Polymorphism

It is also known as Dynamic Method Dispatch. It is a process in which a function call to the overridden method is resolved at Runtime. This type of polymorphism is achieved by Method Overriding.

Method Overriding in Java

- If subclass (child class) has the same method as declared in the parent class, it is known as **method overriding in java**.
- Method overriding is used to provide specific implementation of a method that is already provided by its super class.
- Method overriding is used for runtime polymorphism.

Rules for Java Method Overriding

- method must have same name as in the parent class
- method must have same parameter as in the parent class.
- must be IS-A relationship (inheritance).

```
class Vehicle{
  void run(){System.out.println("Vehicle is runnin
    g");}
}
class Bike extends Vehicle{  void run(){System.o
    ut.println(“Bike  is running");}


  public static void main(String args[]){
  Bike obj = new Bike();
  obj.run();
  }
}
```

No.	Method Overloading	Method Overriding
1)	Method overloading is used to increase the readability of the program.	Method overriding is used to provide the specific implementation of the method that is already provided by its super class.
2)	Method overloading is performed within class.	Method overriding occurs in two classes that have IS-A (inheritance) relationship.
3)	In case of method overloading, parameter must be different.	In case of method overriding, parameter must be same.
4)	Method overloading is the example of compile time polymorphism.	Method overriding is the example of run time polymorphism.
5)	In java, method overloading can't be performed by changing return type of the method only. Return type can be same or different in method overloading. But you must have to change the parameter.	Return type must be same or covariant in method overriding.