18-06-2024

-> package sreeja;

public class Demo

{

static void sayHello()

{

System.out.println("hello students");

}

}

-> package sreeja;

public class sumo extends Demo

{

public void saychello ()

{

System.out.println ("chello students");

}

public static void main (String () args)

{

sumo p = new sumo ();

p.sayHello ; p.saychello ;

}

}

o/p :-

hello students

chello students

-> package sreeja;

public class memo extends sumo

{

public void sayBye ()

{

System.out.println ("Bye students");

}

public static void main (String () args)

{

memo m = new memo; m.sayHello (); m.saychello ();

m.sayBye;

}

}

o/p :-

hello students

chello students

bye students

Differences between Method overloading & Method overriding:-

METHOD OVERLOADING	METHOD OVERRIDING
-> Re using same name for different methods with different number of arguments or with different type of arguments with / without same return type within a class is called Method overloading.	-> Re using same name for super & sub-class methods with same number of arguments, with same type of arguments & with same return type is called Method overriding.
-> No method hides no another method	-> Sub-class method hides super class method
-> Inheritance is not involved	-> Inheritance is involved
-> It is a static Polymorphism	-> It leads to dynamic Polymorphism

Super :- It is a keyword of Java used to access super class members & methods.

Ex:-
```
package creeja;
public class Demo {
    int a = 10;
    void sayHello() {
        System.out.println("Demo Hello");
    }
}
```

```
-> package Sreeja;
public class Sumo extends Demo
{
  int a=20;
  public void sayHello()
  {
    System.out.println("Sumo Hello" + a.super.a);
    super.sayHello();
  }
  public static void main(String[] args)
  {
    Sumo p = new Sumo();
    p.sayHello();
  }
}
```

"/p", - SumoHello 20 10
Demo Hello

CONSTRUCTORS ROLES IN INHERITENCE =>

-> When sub-class object is created not only sub-class constructor but also super class constructor is called.

-> 1st super class constructor word is executed after that sub-class constructor word is executed.

SUPER ( ) :- It is used to access super class constructor. It must be a 1st statement within a constructor.

super ( )

Demo.java :-

```
package Sruja
public class Demo {
    int a;
    public Demo (int a) {
        this ( );
    System.out.println ("Demo 1 arg con");
    }
    public Demo ( ) {
     System.out.println ("Demo default con");
    }
}
```

Sumo.java :-

```
package sraja;
public class sumo extends Demo{
    int x;
    public sumo (int x) {
        super (5);
        System.out.println ("sumo 1 arg cons");
    }
    public sumo ( ) {
        this (5);
        System.out.println ("sumo def constructor");
    }
    public static void main (String[ ] args)
    {
        sumo p = new sumo ( );
    }
}
```

arg -> argument
cons -> constructor

* abstract * concrete

ABSTRACT ( ) & ABSTRACT CLASS :-

A method without body is called abstract method

[abstract ( )]

SYNTAX :-

```
abstract class
abstract returntype name (---);
```

If a class containing any abstract method then the class is said to be abstract class & it should be declared as abstract

SYNTAX :-

```
abstract class name
{
  abstract returntype name ( );
}
```