



SNS COLLEGE OF ENGINEERING



Kurumbapalayam(Po), Coimbatore – 641 107

Accredited by NAAC-UGC with 'A' Grade

Approved by AICTE, Recognized by UGC & Affiliated to Anna University, Chennai

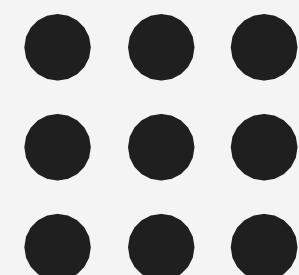
Department of Information Technology

19CS204 OBJECT ORIENTED PROGRAMMING

I YEAR /II SEMESTER

Unit 2- BASIC FEATURES OF JAVA

Overloading



Overloading /R.KAMALAKANNAN/ CSE-
IOT /SNSCE

Overloading

Overloading Methods

- In Java, it is possible to define two or more methods within the same class that share the same name.
- To achieve this parameters must be different
- When this is the case, the methods are said to be overloaded, and the process is referred to as method overloading.
- Method overloading is one of the ways that Java supports polymorphism.

Overloading

Overloading Methods

- When an overloaded method is invoked, Java uses the type and/or number of arguments as its guide to determine which version of the overloaded method to actually call.
- Thus, overloaded methods must differ in the type and/or number of their parameters.
- When Java encounters a call to an overloaded method, it simply executes the version of the method whose parameters match the arguments used in the call.

Overloading

Example

```
class OverloadDemo {  
    void test() {  
        System.out.println("No parameters");  
    }  
  
    // Overload test for one integer parameter.  
    void test(int a) {  
        System.out.println("a: " + a);  
    }  
  
    // Overload test for two integer parameters.  
    void test(int a, int b) {  
        System.out.println("a and b: " + a + " " + b);  
    }  
  
    // Overload test for a double parameter  
    double test(double a) {  
        System.out.println("double a: " + a);  
        return a*a;  
    }  
}
```

```
class Overload {  
    public static void main(String args[]) {  
        OverloadDemo ob = new OverloadDemo();  
        double result;  
        // call all versions of test()  
        ob.test();  
        ob.test(10);  
        ob.test(10, 20);  
        result = ob.test(123.25);  
        System.out.println("Result of ob.test(123.25): " + result);  
    }  
}
```

Overloading

Overloading Constructors

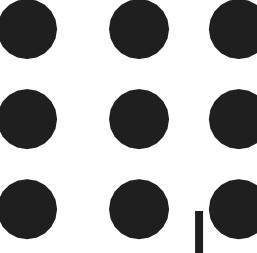
- In addition to overloading normal methods, you can also overload constructor methods.
- The constructor overloading can be defined as the concept of having more than one constructor with different parameters so that every constructor can perform a different task.
- Sometimes there is a need of initializing an object in different ways. This can be done using constructor overloading.

Overloading

Overloading Constructors

```
class Box {  
    double width;  
    double height;  
    double depth;  
  
    // constructor used when all dimensions specified  
    Box(double w, double h, double d) {  
        width = w;  
        height = h;  
        depth = d;  
    }  
  
    // constructor used when no dimensions specified  
    Box() {  
        width = -1; // use -1 to indicate  
        height = -1; // an uninitialized  
        depth = -1; // box  
    }  
  
    // constructor used when cube is created  
    Box(double len) {  
        width = height = depth = len;  
    }  
}
```

```
// compute and return volume  
double volume() {  
    return width * height * depth;  
}  
  
class OverloadCons {  
    public static void main(String args[]) {  
        // create boxes using the various constructors  
        Box mybox1 = new Box(10, 20, 15);  
        Box mybox2 = new Box();  
        Box mycube = new Box(7);  
        double vol;  
  
        // get volume of first box  
        vol = mybox1.volume();  
        System.out.println("Volume of mybox1 is " + vol);  
  
        // get volume of second box  
        vol = mybox2.volume();  
        System.out.println("Volume of mybox2 is " + vol);  
  
        // get volume of cube  
        vol = mycube.volume();  
        System.out.println("Volume of mycube is " + vol);  
    }  
}
```



THANK YOU

Overloading /R.KAMALAKANNAN/ CSE-IOT /SNSCE