Object Oriented Analysis and Design with Java - UE19CS353

Prof. Sindhu R Pai
Department of Computer Science and Engineering

Object Oriented Analysis and Design with Java

Method Overloading

Prof. Sindhu R Pai
Department of Computer Science and Engineering

Object Oriented Analysis and Design with Java

Method Overloading - Agenda

1. Introduction

2. Coding examples - Demo

Object Oriented Analysis and Design with Java

Method Overloading

Introduction

- A feature that allows a class to have more than one method having the same name, if their argument lists are different.

- **Three ways:** In order to overload a method, the argument lists of the methods must differ in either of these:

  Number of parameters

  Data type of parameters

  Sequence of data type of parameters

Note: Method overloading has no relation with return-type

Method Overloading

Coding Example 1: Number of parameters and Data type of parameters

```
class Bird{
    void fly(){
        System.out.println("Bird is flying");
    }
    void fly(int height){
        System.out.println("Bird is flying "+height+" high");
    }
    void fly(String name,int height){
        System.out.println(name+" is flying "+height+" feet high");
    }
}
class P2_methodOverload {
public static void main(String[] args){
        Bird bird1=new Bird();
        bird1.fly();
        bird1.fly(10000);
        bird1.fly("Eagle",10000);
    }
}
```

PES UNIVERSITY ONLINE

Coding Example 2: Number of parameters and Sequence of Data type of parameters

```
class Addition {
    int add(int a, int b) {
        return a + b; }
    int add(int a, int b, int c) {
        return a + b + c; }
    double add(int a, double b) {
        return a + b;    }
    double add(double a, int b) {
        return a + b;    }
}
class P2_MethodOverload {
    public static void main(String[] args) {
        Addition a = new Addition();
        int intAdd1 = a.add(1, 2);
        System.out.println("1+2=" + intAdd1);
        int intAdd2 = a.add(1, 2, 3);
        System.out.println("1+2+3=" + intAdd2);
        double doubleAdd1 = a.add(1, 2.5);
        System.out.println("1+2.5=" + doubleAdd1);
        double doubleAdd2=a.add(3.5,2);
        System.out.println("3.5+2=" + doubleAdd2);
    }   }
```

THANK YOU

Prof. Sindhu R Pai
Department of Computer Science and Engineering

sindhurpai@pes.edu

+91 8277606459