Module: Java + UI + PHP

Course: Core Java

Session 8: Inheritance and Polymorphism

Trainer Notes

1 Session Plan

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Time  (min) | Content | Methodology | Trainer  Approach | Learner  Activity | Learning  Outcome  (Bloom's) | Learning  Outcome  (Gardner's) |
| 15 | Inheritance | Reference to  Reading  Material and  Slides | Facilitate,  Elicit  responses | Think,  Respond,  Identify | Remember,  Understand | Intrapersonal,  interpersonal |
| 15 | Inheritance demo | Reference to  Reading  Material and  Slides | Facilitate,  Elicit  responses | Think,  Respond,  Identify | Remember,  Understand | Intrapersonal,  interpersonal |
| 15 | Polymorphism | Reference to  Reading  Material and  Slides | Facilitate,  Elicit  responses | Think,  Respond,  Identify | Remember,  Understand | Intrapersonal,  interpersonal |
| 15 | Polymorphism demo | Reference to  Reading  Material and  Slides | Facilitate,  Elicit  responses | Think,  Respond,  Identify | Remember,  Understand | Intrapersonal,  interpersonal |
| 25 | Guided Classroom  Activities | Group Activities | Facilitate | Work on guided activities | Remember,  Understand,  Coding | Intrapersonal,  interpersonal |
| 05 | Conclusion | Discussion | Question,  Facilitate,  Guides | Participates,  Recollect  concepts | Remember | Intrapersonal,  interpersonal |

2 Objectives

* Define inheritance
* Explain the need for Inheritance
* Discuss types of Inheritance
* Write Java code to create classes and subclasses in an inheritance hierarchy
* Explain and demonstrate use “super” keyword
* Implement constructor chaining in inheritance
* Explain Polymorphism and Overriding

3 Materials Needed

* Slides

1. Presentation Description

The Facilitator is expected to follow the Presentation Slides as a guideline for the flow of the session.

1. Classroom Activities

**Inheritance and Constructors**

To re organize the classes of the payroll system to use inheritance.

The payroll system of an organization involves calculating the gross salary of each type of employee and the tax applicable to each. The entity classes, their fields and methods are already given in your candidate project. Your task today is to use Inheritance and re-organize related classes in the payroll system application.

Classes given to you in the candidate project are: Employee, Trainer, Sourcing, Manager, Organization and TaxUtil. These classes already have fields, constructors and the respective methods.

Re-organize the given classes to use Inheritance as follows:

* Employee class should be made the base class. The class employee as defined has many common attributes with all the other classes. In fact, Trainer, Manager and Sourcing are types of Employees. The common attributes between these classes are already in the employee class. This will be your base class.
* Derive Manager, Trainer, and Sourcing from the Employee class. Remove any redundant field definitions in the derived classes.
* Constructors: The constructors of the derived classes need to change to use the Employee constructors.
* Override the method CalculateGrossSalary in the derived classes to provide implementations specific to the respective classes
* Do not make any changes to the TaxUtil, Employee or Organization classes.

**Polymorphism**

To use the advantage of polymorphism

The payroll system of an organization involves calculating the gross salary of each type of employee and the tax applicable to each. The entity classes, their fields and methods are already given in your candidate project. Your task today is to change the calculateTax method of TaxUtil class to use polymorphism.

Classes given to you in the candidate project are: Employee, Trainer, Sourcing, Manager, Organization and TaxUtil. These classes already have fields, constructors and the respective methods. The Trainer, Manager and sourcing classes are sub-classes of Employee.

In the taxUtil class, there are four calculateTax methods, one each for Employee, Manager, Trainer, Sourcing. This was declared earlier because the four classes method were not related. Now that we have tied them together using Inheritance, these four methods can be replaced with just one method.

Replace the four calculateTax methods with just one method calculateTax(Employee). Verify that the tax for all the four classes are calculated properly