Aror University of Art, Architecture, Design &

Heritage Sukkur

Department of AI-Multimedia and Gaming

BS-AI and MMG, Fall 2024 Batch

_____ Lab 10: Abstract Classes and Interfaces in Java Course: Object Oriented Programming

Date: 29 April, 2025

Objective:

- Understand the purpose and use of Abstract Classes and Interfaces in Java.
- Learn to implement abstraction, polymorphism, and multiple inheritance.
- Differentiate between Abstract Class and Interface with hands-on examples.

Task 1: Creating and Using an Abstract Class

Objective: Understand basic abstract class creation and use.

- Create an abstract class named `Animal`.
- Add an abstract method `makeSound()`.
- Add a concrete method `eat()` that prints `"Animal is eating"`.
- Create two subclasses `Dog` and `Cat` that extend `Animal` and implement `makeSound()` differently.
- In the main method, create objects of `Dog` and `Cat`, and call both `eat()` and `makeSound()`.

Task 2: Creating and Using an Interface

Objective: Learn basic Interface creation and implementation.

- Create an interface `Vehicle` with methods `start()` and `stop()`.
- Create two classes `Car` and `Bike` that implement the `Vehicle` interface.
- Implement `start()` and `stop()` methods differently in both classes.
- Create objects and call methods in the `main()`.

Task 3: Abstract Class vs Interface

Objective: Highlight key differences practically.

- Create an abstract class `Shape` with an abstract method `area()`.
- Create an interface `Printable` with method `print()`.
- Create a class `Rectangle` that **extends Shape** and **implements Printable**.
- Implement both `area()` and `print()` in `Rectangle`.
- Calculate and print area for given width and height, then print a custom message.

Task 4: Interface Inheritance (Multiple Interfaces)

Objective: Practice multiple inheritance using Interfaces.

- Create two interfaces: `Flyable` (with method `fly()`) and `Swimmable` (with method `swim()`).
- Create a class `Duck` that implements both `Flyable` and `Swimmable`.
- Implement both methods with simple print statements.
- Create an object of `Duck` and call both methods.

Task 5: Real-World Problem (Mini Project)

Objective: Apply abstract classes and interfaces in a real-world simulation.

- Create an abstract class `Employee` with:
  - Attributes: `name`, `id`.
  - Abstract method `calculateSalary()`.
- Create an interface `TaxPayer` with method `payTax()`.
- Create two classes `FullTimeEmployee` and `PartTimeEmployee` that:
  - Extend `Employee`
  - Implement `TaxPayer`
- Implement `calculateSalary()` differently (fixed monthly for full-time, hourly rate for part-time).
- In main:
  - Create objects for both employee types,
  - Calculate and display salaries,
  - Call `payTax()`.