**Task 7 - Tính kế thừa và đa hình trong OOP**

1. **Kế thừa (Inheritance)**

- Kế thừa là 1 tính năng của lập trình hướng đối tượng (OOP), cho phép một lớp mới tiếp nhận các thuộc tính và phương thức của một lớp khác (trừ các thuộc tính và phương thức private).

- Kế thừa giúp tái sử dụng mã nguồn, giảm lặp lại và tăng tính mô-đun của chương trình.

- Kế thừa bao gồm:

* Đơn kế thừa: một lớp con duy nhất kế thừa từ một lớp cha duy nhất
* Kế thừa đa cấp: lớp con kế thừa từ nhiều lớp cha khác nhau
* Kế thừa phân cấp: một lớp cha được kế thừa bởi nhiều lớp con
* Đa kế thừa: thực hiện thông qua Interface
* Kế thừa hỗn hợp: thực hiện thông qua Interface

1. **Extends**

- Extends là từ khóa dùng để chỉ rõ lớp con kế thừa từ lớp cha (biểu diễn mối quan hệ cha - con).

- Lớp con kế thừa các thuộc tính và phương thức được khai báo với access modifier là public hoặc protected. Các thuộc tính, phương thức được khai báo với private không được kế thừa.

1. **Super**

- Super là từ khóa giúp gọi đến các thuộc tính và phương thức của lớp cha mà lớp đó kế thừa.

1. **Đa hình (Polymorphism)**

- Đa hình là khả năng của một đối tượng thể hiện nhiều hình dạng khác nhau.

1. **Ghi đè (Override)**

- Ghi đè là kỹ thuật cho phép lớp con cung cấp định nghĩa mới cho một phương thức đã được định nghĩa trong lớp cha.

- Giúp lớp con mở rộng hoặc thay đổi hành vi của lớp cha mà không ảnh hưởng đến mã nguồn của lớp cha.

- Khi một phương thức được ghi đè trong lớp con, phương thức của lớp con được gọi thay vì phương thức của lớp cha khi đối tượng của lớp con được sử dụng.

- Ghi đè cung cấp khả năng thực hiện tính đa hình trong Java: phương thức giống nhau thực hiện chức năng khác nhau.

1. **Lớp trừu tượng (Abstract class)**

- Cơ chế cho phép thực hiện tính trừu tượng trong Java.

- Lớp trừu tượng: không thể khởi tạo đối tượng trực tiếp từ lớp này. Nó chứa ít nhất một phương thức trừu tượng (khai báo với từ khóa abstract và không có thân hàm).

- Các lớp con phải triển khai tất cả các phương thức trừu tượng trong lớp cha (ghi đè) hoặc khai báo lại chúng là một phương thức trừu tượng.

- Khai báo lớp trừu tượng:

*<accessModifier> abstract class <className> {*

*//dataMembers;*

*//statements;*

*/methods;*

*}*

methods có thể là trừu tượng, không trừu tượng hoặc cả hai

- Khai báo phương thức trừu tượng:

*abstract <modifier> <returnType> <methodName>(<parameterList>*);

1. **Giao diện (Interface)**

- Là cách chỉ định hành vi của một lớp bằng cách cung cấp một tập hợp các phương thức trừu tượng. Giúp đạt được tính trừu tượng, đa hình và kế thừa trong Java.

- Đặc điểm:

* Không thể sử dụng để khởi tạo đối tượng
* Không chứa constructor
* Các phương thức phải được triển khai ở lớp triển khai
* Các phương thức được khai báo mặc định là abstract (có thể khai báo là public và private (từ Java 9)
* Các thuộc tính mặc định là public, static và final

- Mục đích:

* Trừu tượng hoàn toàn: tách biệt hoàn toàn các đối tượng với nhau
* Đa kế thừa: đạt được đa kế thừa mặc dù Java không hỗ trợ, một lớp triển khai từ nhiều interface
* Tính đa hình
* Kết nối lỏng (Loose Coupling): định nghĩa các lớp một cách độc lập, không phụ thuộc vào nhau

- Khai báo Interface:

*[accessModifier] interface <interfaceName> [extends SuperInterface1, SuperInterface2, …] {*

*// Attributes and methods*

*}*

access modifier: public, default, private (Java 9 trở lên)

- Triển khai Interface: sử dụng từ khóa **implements**, ví dụ:

*public class Circle implements Shape {*

*//code*

*}*

- Kế thừa Interface: sử dụng từ khóa **extends**