OOP

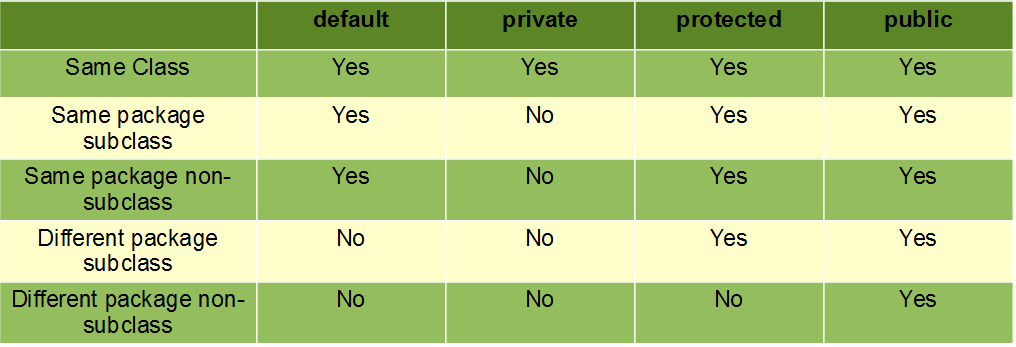
object-oriented programming

Lập trình hướng đối tượng

4 tính chất: tính đóng gói – encapsulation, tính kế thừa – inheritance, tính đa hình – polymorphism, tính trừu tượng – abstraction

1. Tính đóng gói – encapsulation

Trong mỗi class muốn bảo vệ dữ liệu, ta sử dụng access modifier như private, public,…



Sử dụng những phương thức public để truy cập các dữ liệu private, ví dụ như lấy và sửa dữ liệu trong **java** thì có hàm **set** và **get**

1. Tính kế thừa – inheritance

Tái sử dụng code - những dòng code có khả năng bị lặp lại, rút gọn code và có tính chất phân tầng

employee

hr

programer

2 loại kế thừa: kế thừa 1 cấp, kế thừa đa cấp

Ví dự đơn kế thừa: animal có hàm ăn khi lớp chó kế thừa animal thì nó cũng sẽ có hàm ăn

Ví dụ đa kế thừa: animal có hàm ăn, lớp chó có hàm sủa và kế thừa animal , lớp chó con kế thừa lớp chó thì sẽ kế thừa hàm ăn và sủa

Mỗi class chỉ được kế thừa 1 class nhưng được implement rất nhiều interfaces

1. Tính đa hình – polymorphism

Overloading và overwriting

Overloading : hàm có tên giống nhau nhưng parameter – tham số đầu vào khác nhau thì các ngôn ngữ lập trình vẫn hỗ trợ

Overwriting – đi liền với tính kế thừa: lớp con định nghĩa lại các phương thức kế thừa ở lớp cha, muốn phương thức được cấp quyền overwriting lại thì cần thêm từ khóa ở mỗi ngôn ngữ là khác nhau

1. Tính trừu tượng – abstraction

Có 2 thể loại : abstrac class và interfaces

Đơn giản hóa vấn đề trừ tượng, abs class được dùng để tạo nên khung định nghĩa của 1 loại object nhưng nó không thể tạo 1 thực thể

**Interface**: chỉ có asbtrac method – không thể định nghĩa mà chỉ có khái báo

**Abstrac class**: vẫn là 1 class nhưng có thêm abstrac funtion như trong interface và không thể tạo ra 1 intance (vx có thể nhưng nó sẽ phải tạo ra 1 lớp cài đặt toàn bộ những abs method có trong abs class hay còn gọi là anonimus class), những lớp mà kế thừa abstrac class phải định nghĩa toàn bộ những abstrac funtion có trong abs class

Mỗi class chỉ được kế thừa 1 class nhưng được implement rất nhiều interfaces

Giải thích vì sao mà đã có abs class mà vx còn interfaces ( **quan hệ is-a và has-a**)

Interface dùng để tiện trong thay đổi

Trong access modifier : interface không thể có private😊, chỉ có public abs method; trong abs class thì chỉ có public và procted method abs method và không thể có private

Trong kế thừa: interface kế thừa n interface khác nhưng không thể kế thừ từ 1 lớp, abs class có thể kế thừa từ class và abs class