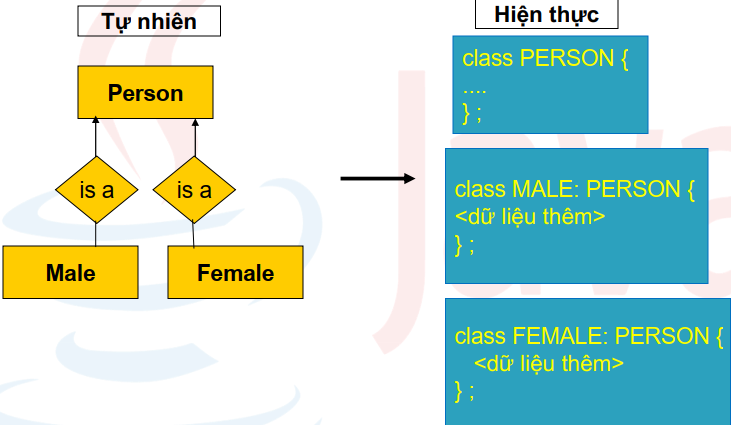
**Object Oriented Programming (OOP)**

* Chương trình: sự hoạt động của các đối tượng 🡪 Giống tự nhiên
* Đối tượng thực thi một hoạt động tức là đối tượng thực hiện một phương thức mà đối tượng này có khả năng
* Một chương trình: là một trật tự các lời yêu cầu đối tượng thực hiện phương thức của mình.

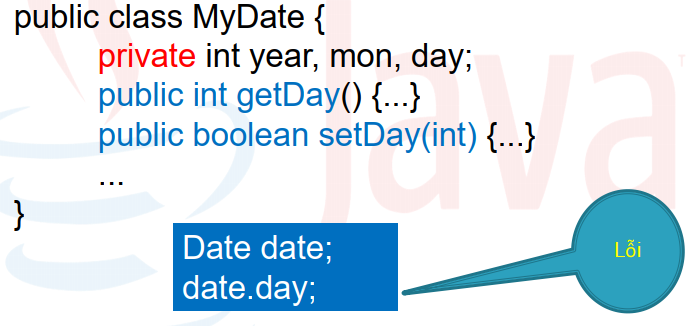
🡺 Chương trình là một kịch bản (script)

**Ưu điểm**

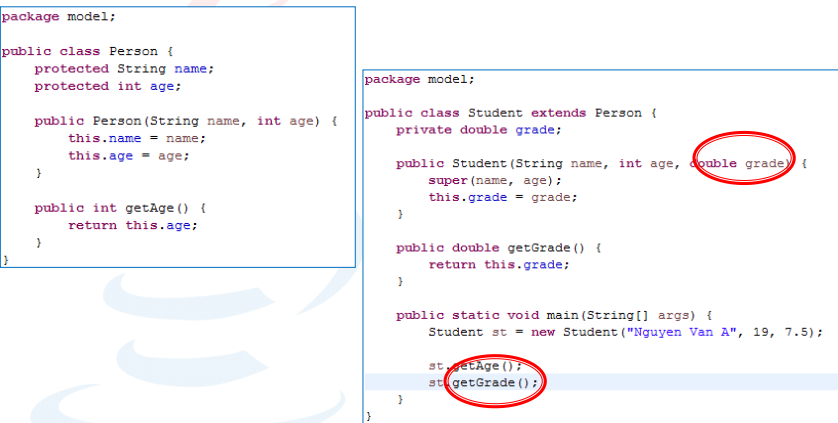
* Dễ mô tả các quan hệ phân cấp trong thế giới tự nhiên



* Có tính bảo mật cao
  + Bên ngoài không thể tuỳ tiện truy cập một dữ liệu thuộc tính



* Tái sử dụng code



**Tính đóng gói (Encapsulation)**

* Là cơ chế ràng buộc dữ liệu và thao tác trên dữ liệu đó thành một thể thống nhất, tránh được các tác động bất ngờ từ bên ngoài. Thể thống nhất này được gọi là đối tượng.
* Đối tượng (object):

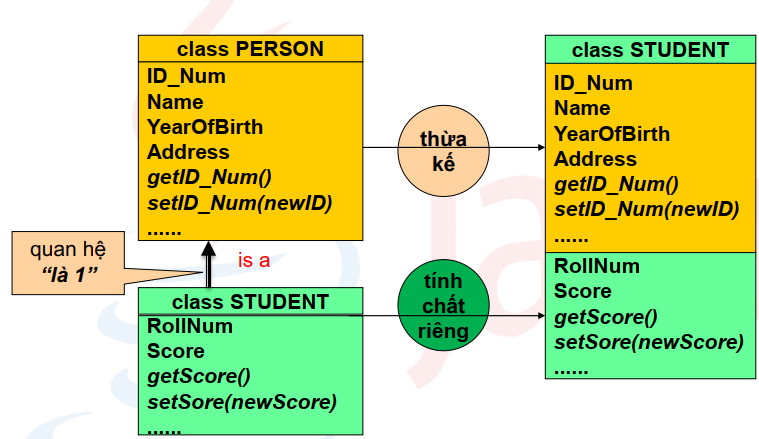
➔ Bao gói dữ liệu + phương thức.

* Đối tượng phải thuộc một lớp (class).
* Muốn làm việc trên đối tượng, ta phải xây dựng lớp

➔ Class = data (biến, thuộc tính)+ methods (code)

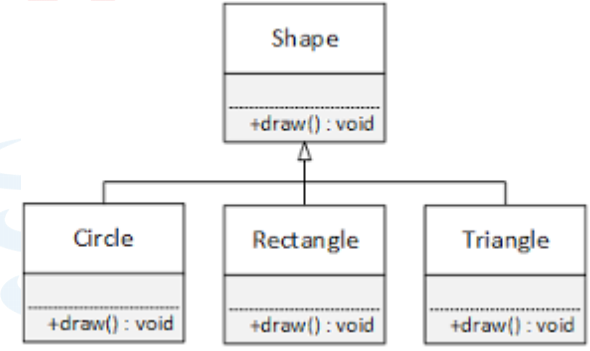
* Che giấu đi dữ liệu.
* Bên ngoài chỉ tương tác được với đối tượng qua một số phương thức.
* Phương thức giao tiếp (public, interface methods): phương thức giúp đối tượng giao tiếp với môi trường.
* Phương thức riêng/nội (private, internal): phương thức xử lý dữ liệu nội tại của đối tượng).

**Tính kế thừa (Inheritance)**



**Tính đa hình (Polymorphism)**

* Kỹ thuật cho phép thay đổi nội dung của cùng một phương thức trong 2 lớp cha con.
* Bản chất: Sửa code của cùng một phương thức ở 2 lớp cha-con để các code này khác nhau.
* Ví dụ: Lớp Shape có phương thức draw() nhưng ở ba lớp con Rectangle, Triangle, Circle phương thức này phải được thể hiện khác nhau



**Tính trừu tượng (Abstraction)**

* Trừu tượng trong thực tế còn có thể hiểu là cái gì đó không có thực.
* Trong OOP: giúp che giấu các hoạt động bên trong và chỉ hiển thị những tính năng thiết yếu của đối tượng tới người dùng
* Java trừu tượng hóa thông qua các lớp trừu tượng (*Abstract class*) và các giao diện (*Interface*)
  + Abstract class:
    - Không thể được dùng để tạo ra các đối tượng như những lớp bình thường khác; được xem như là một cái sườn để tạo ra các lớp con
  + Interface:
    - Một bản thiết kế của một lớp; chỉ có các phương thức trừu tượng; không có thuộc tính (attributes)

**Lớp và đối tượng**

* Lớp (class) là khuôn mẫu để sinh ra đối tượng
* Đối tượng là thể hiện (instance) của một lớp.
* Đối tượng có
  + Định danh
  + Thuộc tính (dữ liệu)
  + Hành vi (phương thức)

**Hệ thống hướng đối tượng**

* Bao gồm một tập các đối tượng
  + Mỗi đối tượng chịu trách nhiệm một công việc
* Các đối tượng tương tác thông qua trao đổi thông điệp (message)
* Các đối tượng có thể tồn tại phân tán/có thể hoạt động song song