ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG VIỆT - HÀN
Vietnam - Korea University of Information and Communication Technology

PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG PHẦN MỀM

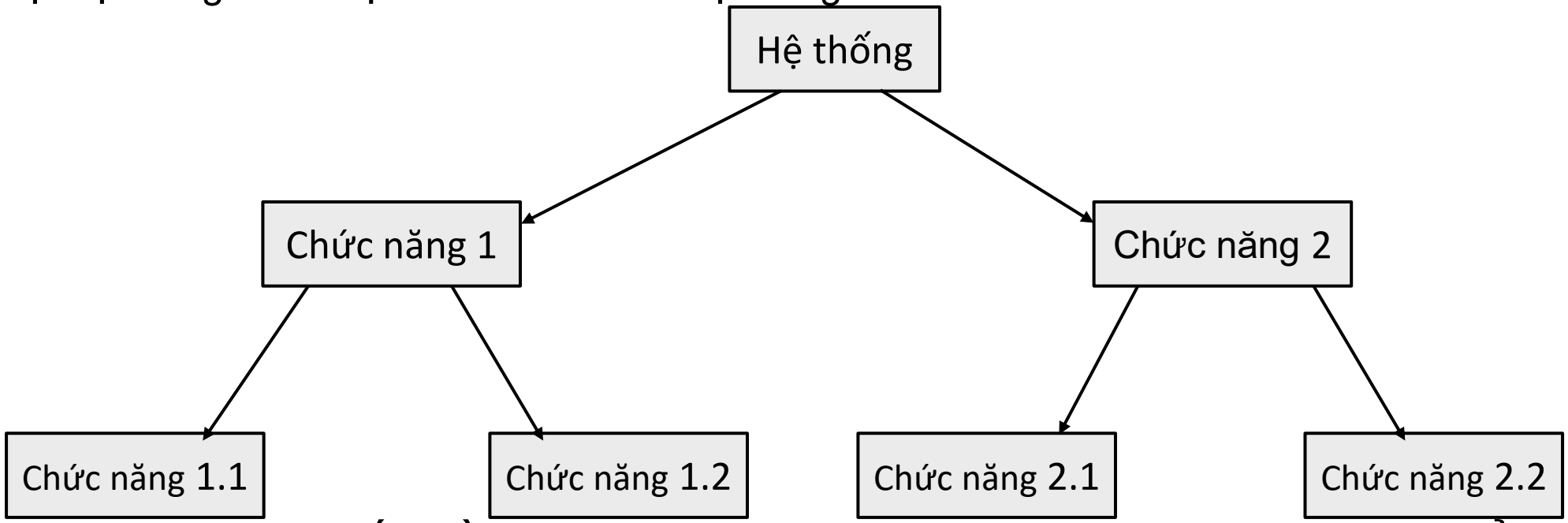
Lê Viết Trương
Khoa Khoa học máy tính

Giới thiệu về phát triển hướng đối tượng

- Tiếp cận hướng chức năng
- Tiếp cận hướng đối tượng
- Các khái niệm hướng đối tượng
  - Đối tượng/Objects
  - Lớp/Classes
  - Đóng gói/Encapsulation
  - Thừa kế/Inheritance
  - Tính đa hình/Polymorphism
  - Trừu tượng/Abstraction

Tiếp cận hướng chức năng/thủ tục

- Dựa vào các chức năng cụ thể của hệ thống
  - Một hệ thống bao gồm một số chức năng
- Phân rã các chức năng thành các chức năng con
  - Một hệ thống bao gồm các hệ thống con
  - Một hệ thống con được chia thành các hệ thống con nhỏ hơn



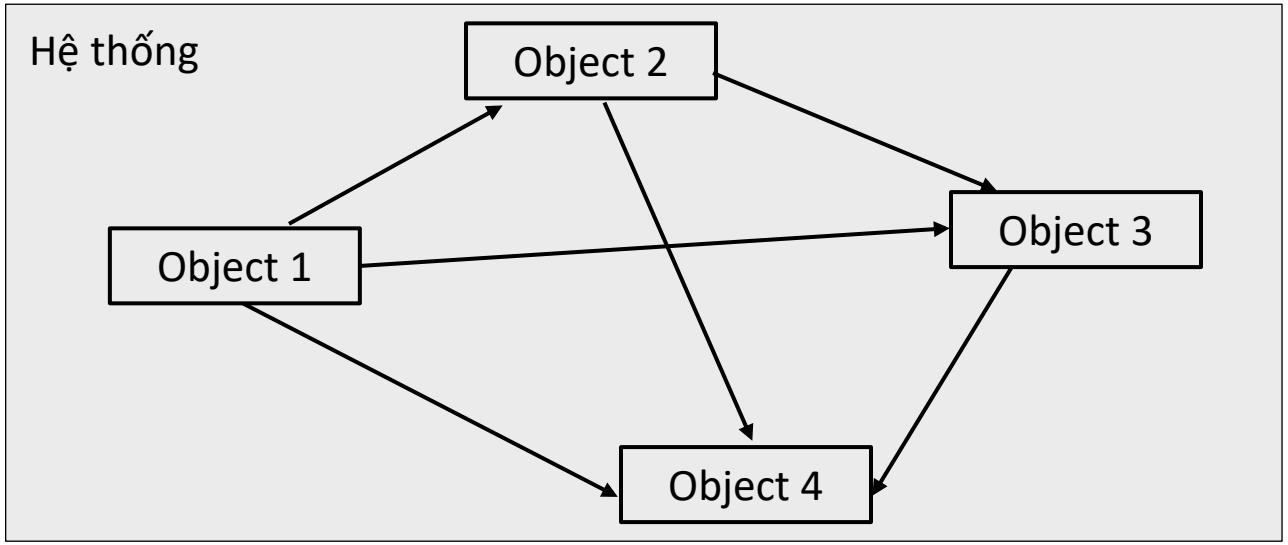
- Các chức năng giao tiếp bằng cách sử dụng dữ liệu được chia sẻ hoặc chuyển các tham số

Tiếp cận hướng chức năng

- Ưu điểm
  - Dễ áp dụng
  - Hoạt động tốt khi dữ liệu đơn giản
  - Giúp giảm bớt sự phức tạp
  - Đạt được kết quả mong đợi
- Nhược điểm
  - Các chức năng được tách biệt khỏi dữ liệu
  - Cấu trúc của hệ thống được xác định dựa trên các chức năng, do đó việc thay đổi chức năng sẽ gây khó khăn cho việc thay đổi cấu trúc
  - Hệ thống mở yếu
  - Khó sử dụng lại
  - Một chi phí bảo trì đáng kể

Tiếp cận hướng đối tượng

- Giải pháp của một vấn đề được tổ chức xung quanh khái niệm về các đối tượng
- Đối tượng là một trừu tượng của dữ liệu cũng chứa các hàm
- Một hệ thống bao gồm các đối tượng và mối quan hệ giữa chúng
- Các đối tượng được giao tiếp bằng cách trao đổi thông điệp để thực hiện một nhiệm vụ
- Không có biến toàn cục
- Đóng gói
- Thừa kế



Tiếp cận hướng đối tượng

- Ưu điểm
  - Rất gần với thế giới thực
  - Dễ dàng sử dụng lại
  - Ẩn thông tin (đóng gói)
  - Chi phí phát triển thấp hơn (kế thừa)
  - Thích hợp cho các hệ thống phức tạp
- Tiếp cận hướng chức năng v.s. tiếp cận hướng đối tượng
  - Tiếp cận hướng chức năng

  Hệ thống = thuật toán + cấu trúc dữ liệu
  - Tiếp cận hướng đối tượng

  Hệ thống = Σ đối tượng

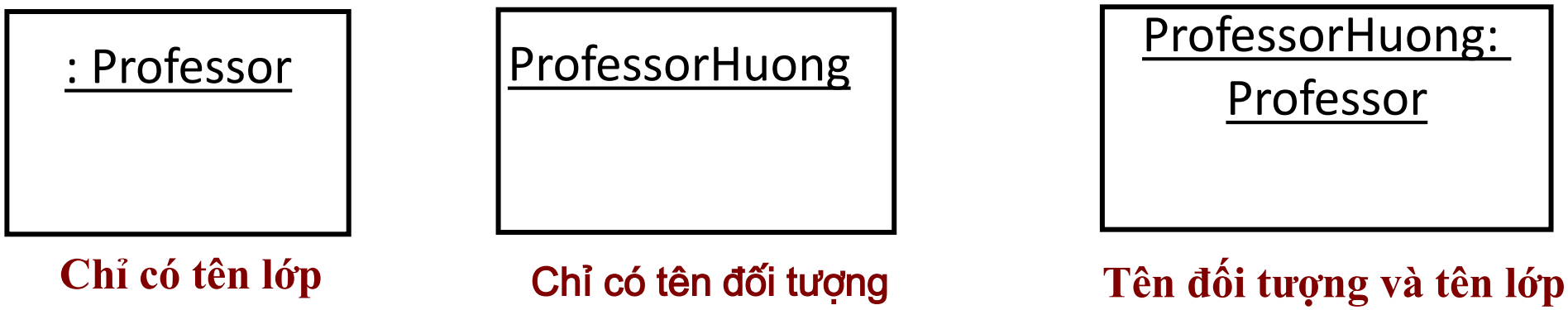
  Đối tượng = thuật toán + cấu trúc dữ liệu

Đối tượng/Objects

- Đối tượng là khái niệm mô tả một thực thể trong thế giới thực
- Có mối quan hệ giữa các đối tượng
- Thí dụ
  - Sinh viên “Hạnh” là một đối tượng
  - Sinh viên không thể là một đối tượng!
- Đối tượng = trạng thái + hành vi + định danh
  - Trạng thái (dữ liệu) mô tả các đặc điểm của một đối tượng tại một thời điểm nhất định và được lưu trong các biến
  - Hành vi được thể hiện bằng các chức năng của đối tượng
  - Mỗi đối tượng có một định danh riêng

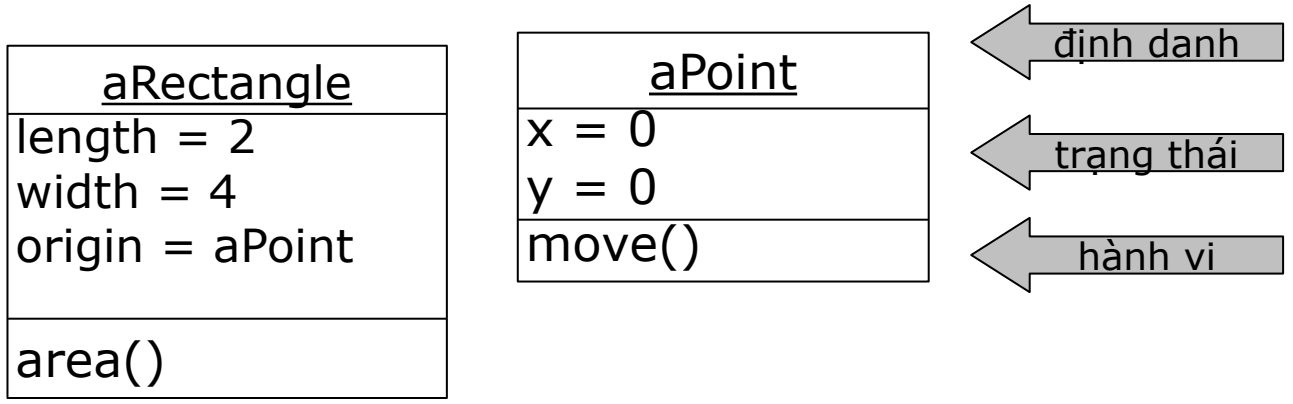
Đối tượng/Objects

- Một đối tượng được biểu diễn bằng hình chữ nhật với tên được gạch chân



Chỉ có tên lớp | **Chỉ có tên đối tượng** | **Tên đối tượng và tên lớp**

- Ví dụ

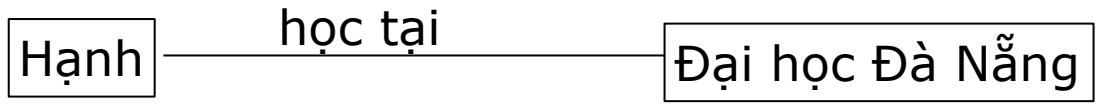


Đối tượng

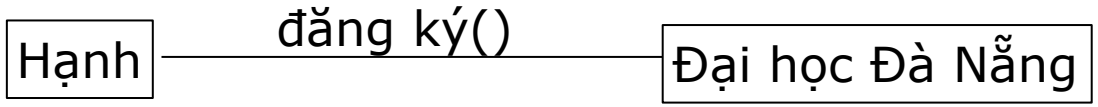
- Trạng thái = Tập hợp các thuộc tính
  - Thuộc tính mô tả một đặc tính của đối tượng
  - Tại mọi thời điểm, một thuộc tính có một giá trị trong một phạm v thuộc tính cụ thể
  - Thí dụ
    - Xe có các đặc tính: màu sắc, chiều dài, chiều rộng, trọng lượng, số km,…
    - Một chiếc Renault 207 nặng 1300 pound, màu đỏ,…
- Hành vi = Tập hợp các chức năng
  - Một chức năng / phương thức là khả năng của đối tượng để thực hiện một tác vụ
  - Hành vi phụ thuộc vào trạng thái
    - Ví dụ: Một chiếc ô tô có thể nổ máy rồi chạy,…

Đối tượng

- Liên kết
  - Giữa các đối tượng có thể có các liên kết
  - Thí dụ



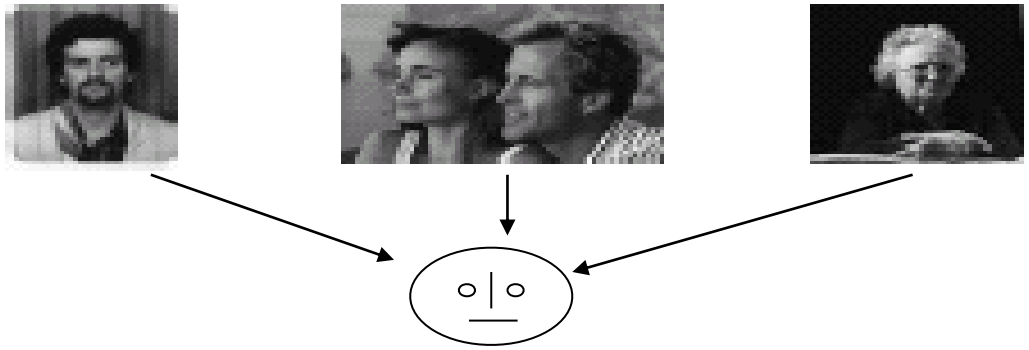
- Giao tiếp giữa các đối tượng
  - Gửi thông điệp



  - Các kiểu thông điệp
    - constructor
    - destructor
    - getter
    - setter
    - others

Lớp

- Một lớp là một mô tả trừu tượng của một tập hợp các đối tượng có
  - Thuộc tính tương tự
  - Hành vi chung
  - Mối quan hệ chung với các đối tượng khác
- Lớp là một sự trừu tượng
  - Sự trừu tượng: Tìm kiếm các khía cạnh chung và bỏ qua các điểm khác biệt



- Giảm sự phức tạp

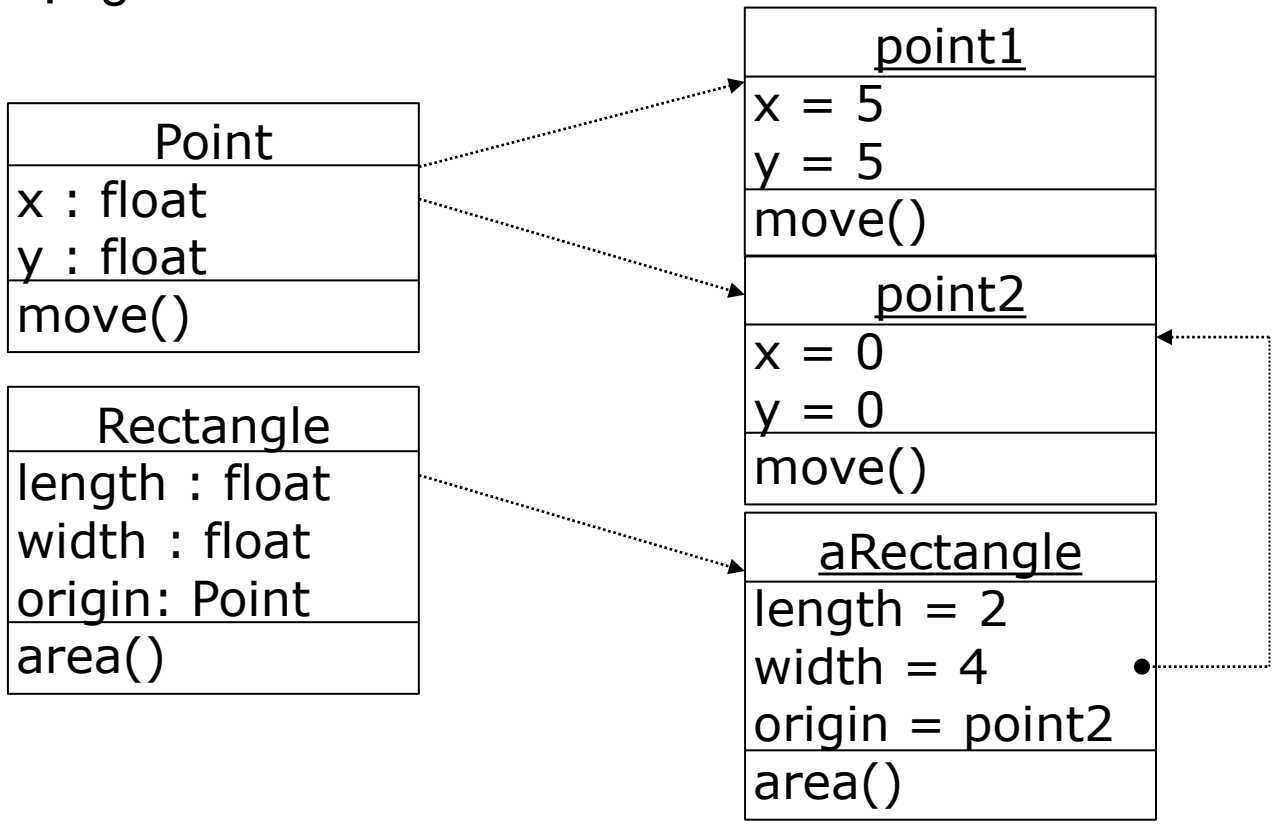
Lớp

- Quan hệ
  - Có thể có mối quan hệ giữa các lớp
  - Mối quan hệ giữa các lớp là tập hợp các liên kết giữa các đối tượng của chúng

- Lớp / Đối tượng
  - Một đối tượng là một thể hiện của một lớp
  - Giá trị là một thể hiện của một thuộc tính
  - Liên kết giữa các đối tượng là một thể hiện của mối quan hệ giữa các lớp

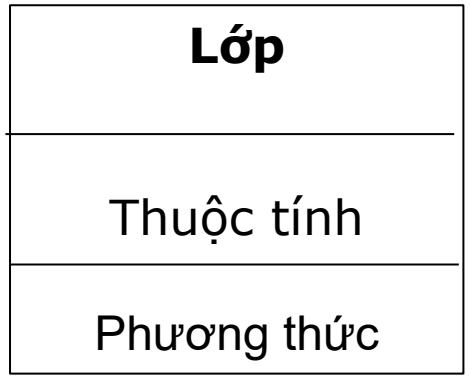
Lớp

- Ví dụ: Lớp / Đối tượng



Đóng gói

- Dữ liệu + Xử lý dữ liệu = Đối tượng
- Thuộc tính + Phương thức = Lớp



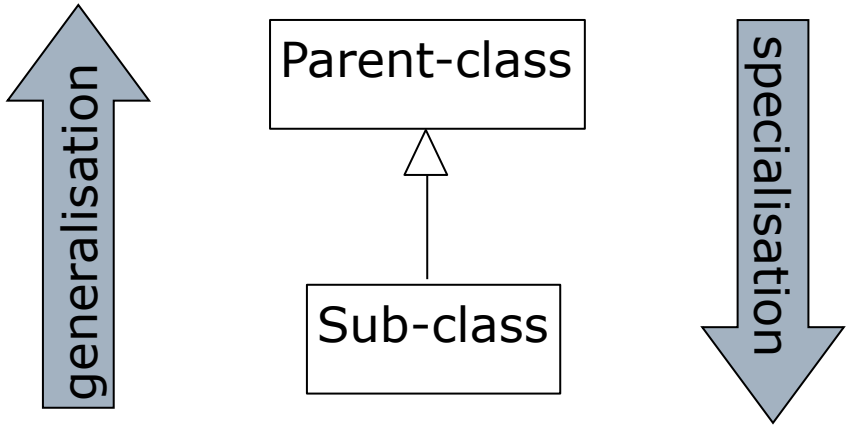
- Trạng thái của đối tượng được đóng gói bởi một tập các thuộc tính
- Hành vi được đóng gói bởi một tập các phương thức
  - Người dùng của một đối tượng biết các thông điệp mà đối tượng có thể nhận được (các phương thức công khai)
  - Việc triển khai các phương pháp bị ẩn vớii người dùng bên ngoài

Đóng gói

- Ưu điểm
  - Ẩn thông tin
  - Hạn chế truy cập thông tin từ bên ngoài
  - Tránh những thay đổi toàn cục trong toàn bộ hệ thống: triển khai nội bộ có thể được sửa đổi mà không ảnh hưởng đến người dùng bên ngoài
  - Tạo điều kiện thuận lợi cho mô-đun
  - Dễ dàng sử dụng lại
  - Dễ bảo trì

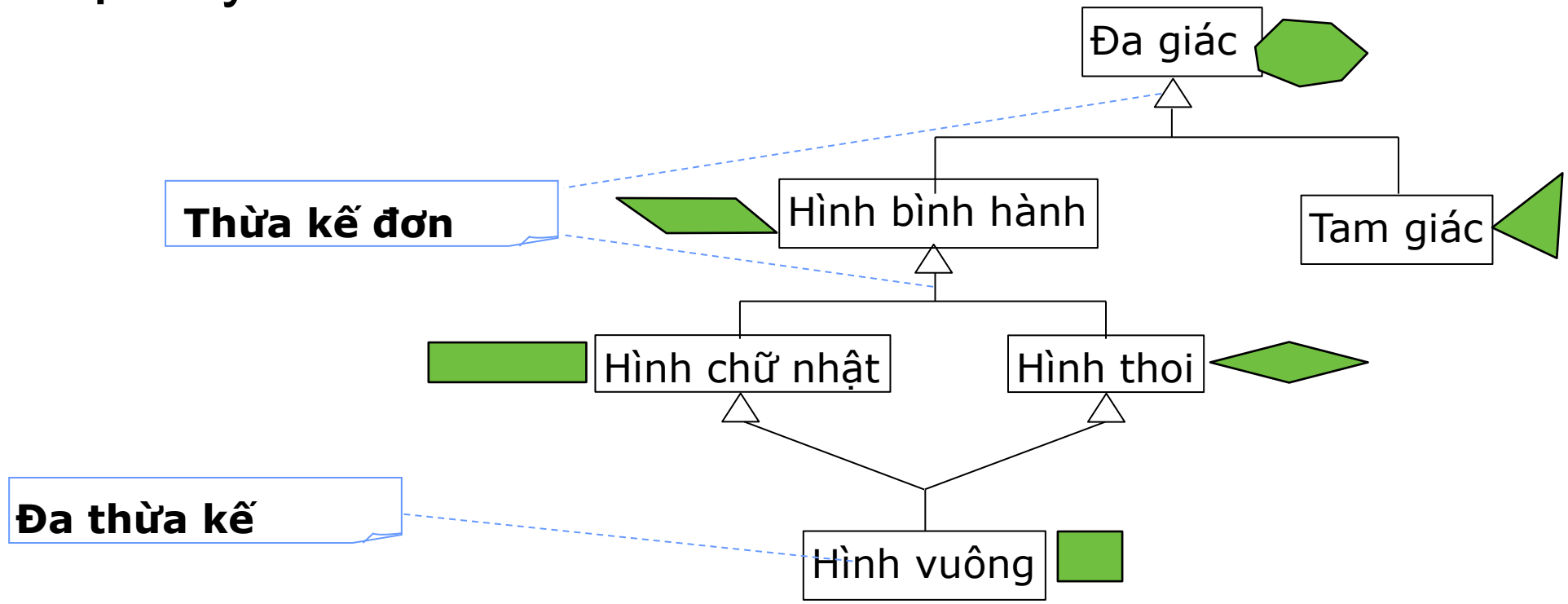
Thừa kế

- Kế thừa cho phép các lớp khác sử dụng lại trạng thái và hành vi của một lớp
- Một lớp được dẫn xuất từ một hoặc nhiều lớp bằng cách chia sẻ các thuộc tính và phương thức
- Lớp con kế thừa các thuộc tính và phương thức của lớp cha
- Tổng quát hóa / Chuyên môn hóa
  - Tổng quát hóa: các thuộc tính chung của các lớp con được sử dụng để xây dựng lớp cha
  - Chuyên môn hóa: các lớp con được xây dựng từ lớp cha bằng cách thêm vào chúng các thuộc tính khác duy nhất



Thừa kế

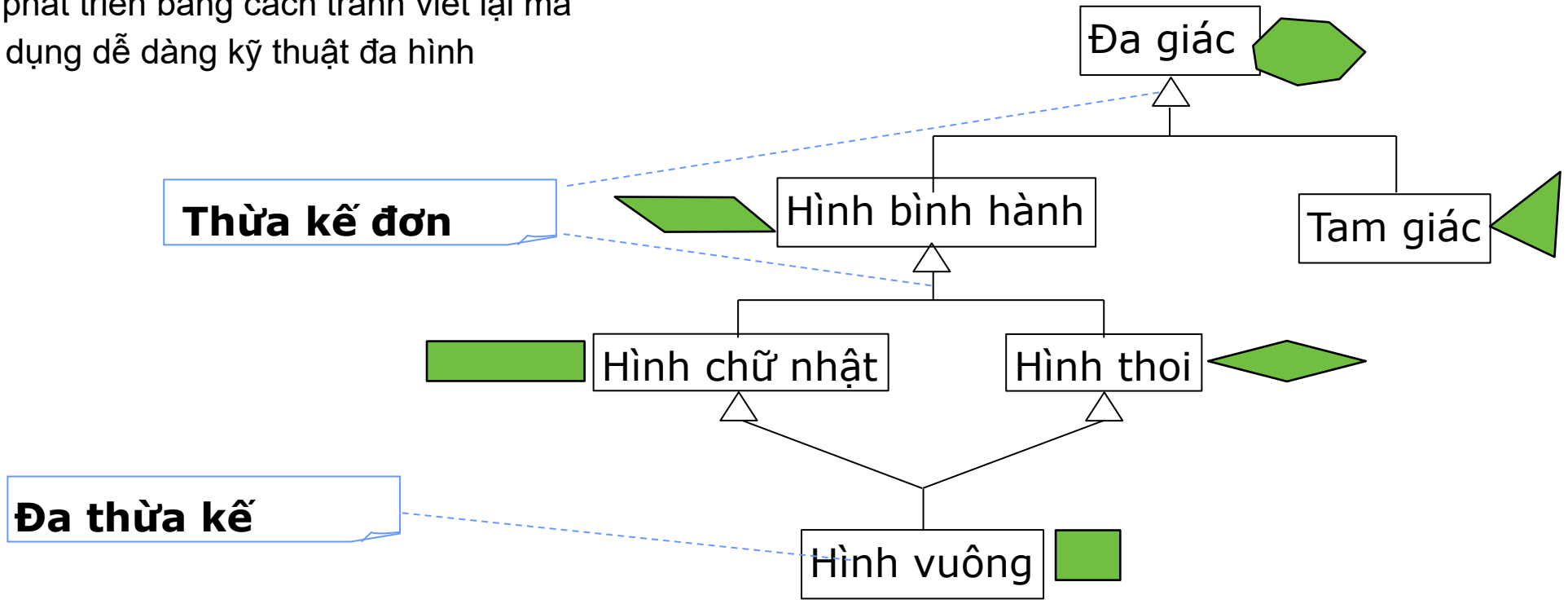
- **Thừa kế đơn:** một lớp con kế thừa chỉ từ một lớp cha
- **Đa thừa kế:** một lớp con kế thừa từ nhiều lớp cha
- Ví dụ: cây thừa kế



- Khó khăn của đa thừa kế là gì?

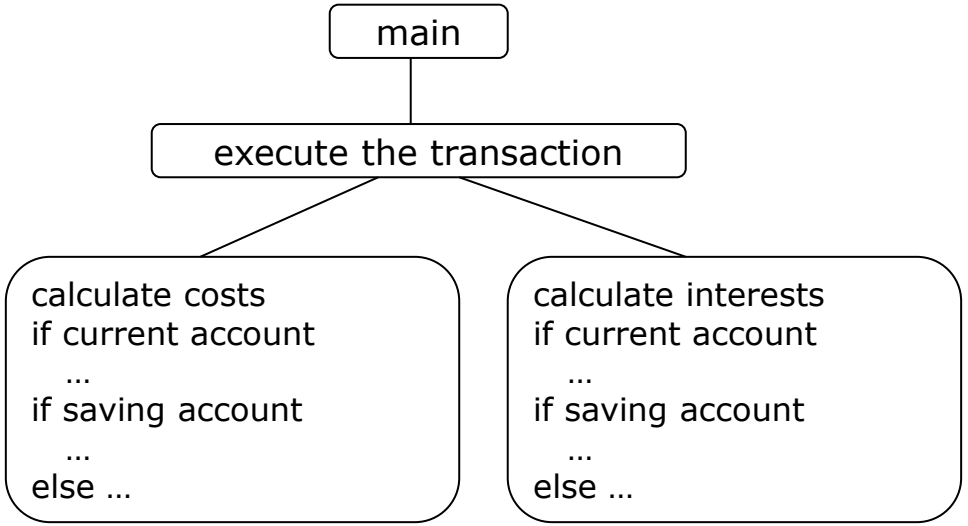
Thừa kế

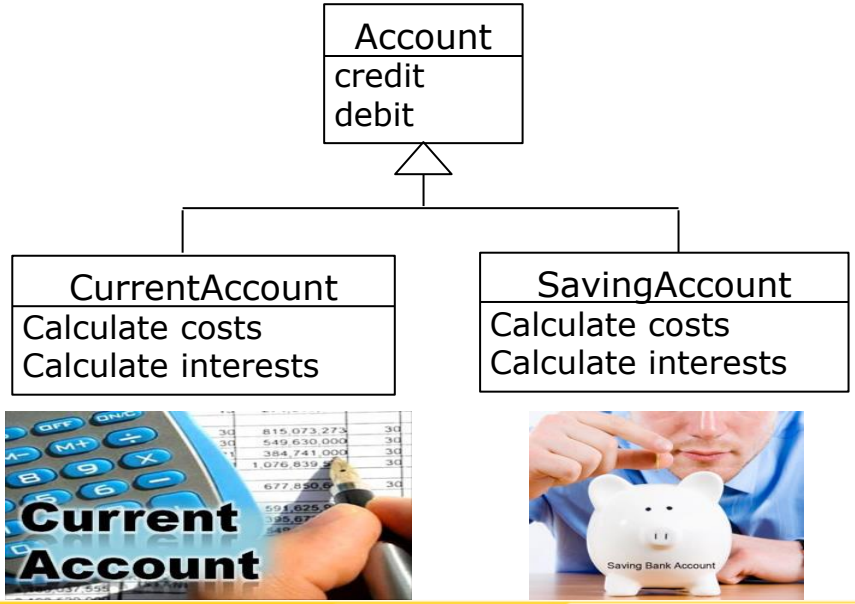
- Ưu điểm
  - Tổ chức các lớp
    - các lớp được tổ chức theo thứ bậc
    - tạo thuận lợi cho việc quản lý các lớp
  - Xây dựng các lớp
    - các lớp con được xây dựng từ các lớp cha
  - Giảm chi phí phát triển bằng cách tránh viết lại mã
  - Cho phép áp dụng dễ dàng kỹ thuật đa hình



Đa hình

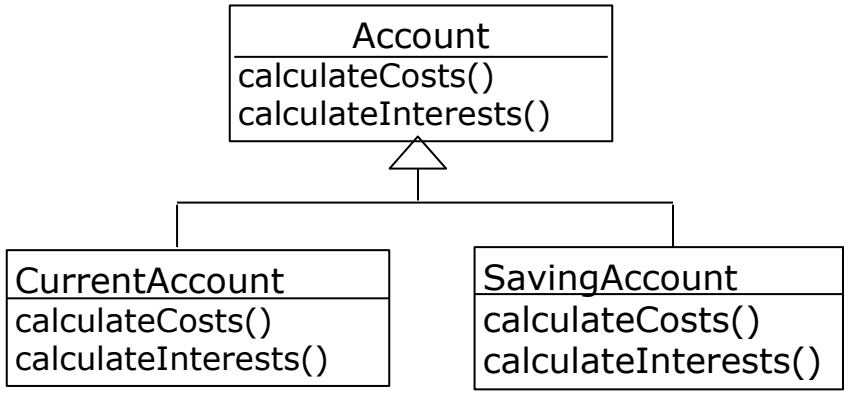
- Tính đa hình của các phương thức
  - Các phương pháp khác nhau có khả năng trả lời một yêu cầu
  - Các phương thức có cùng tên được định nghĩa khác nhau (các hành vi khác nhau) trong các lớp khác nhau
  - Các lớp con kế thừa đặc tả các phương thức từ lớp cha và các phương thức này có thể được định nghĩa lại một cách thích hợp
  - Giảm việc sử dụng các câu lệnh điều kiện (ví dụ: if-else, switch)
- Tiếp cận thủ tục so với tiếp cận hướng đối tượng





Đa hình: liên kết động

- Phương thức được thực thi bởi một đối tượng phụ thuộc vào lớp của đối tượng: liên kết động
- Liên kết động là cần thiết khi
  - Một biến đề cập đến một đối tượng có lớp thành viên là một phần của cây kế thừa
  - Một số phương thức tồn tại cho cùng một thông điệp (tên) trong cây kế thừa (đa hình)

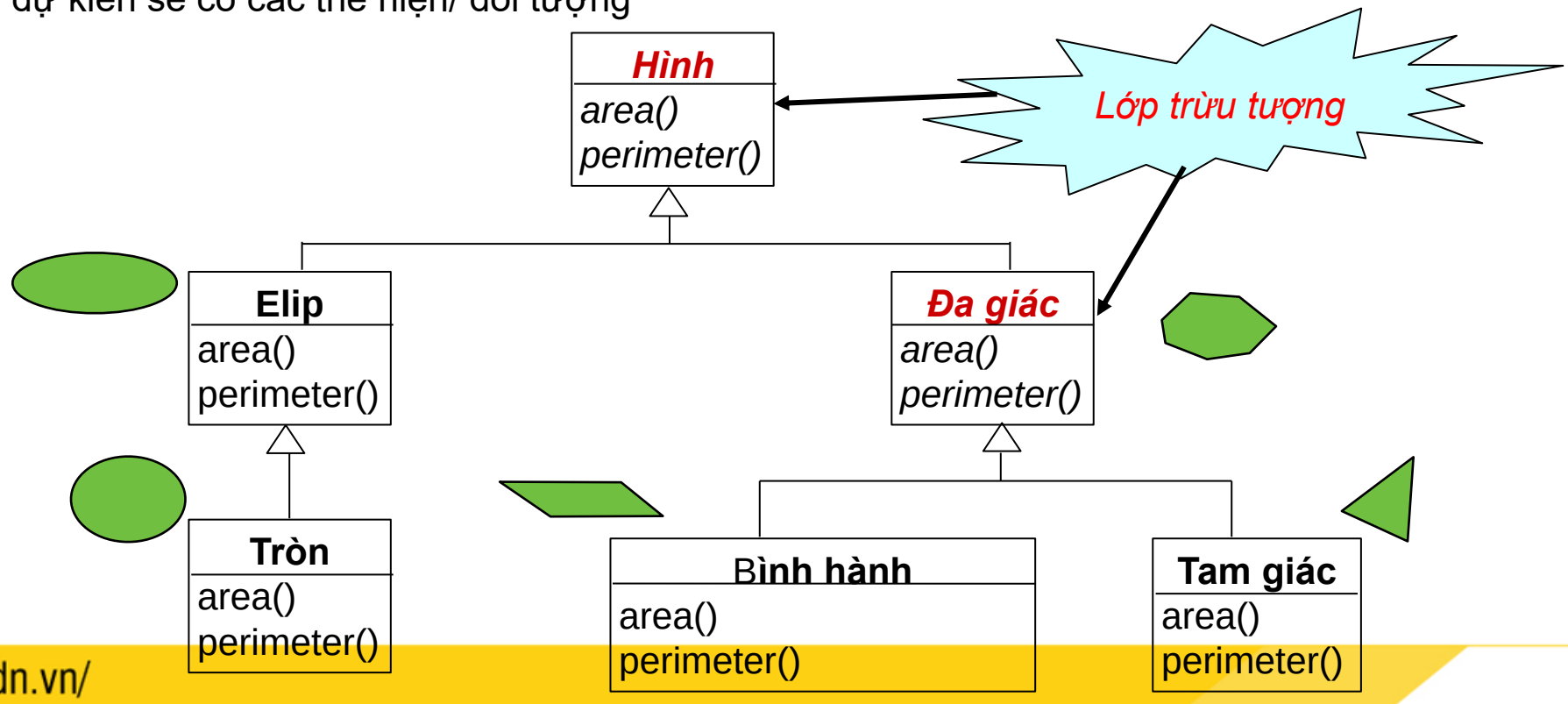



```
int calculateCost(Account accounts)
{
    int s = 0;
    for (int i = 0; i < accounts.length; i++)
            s = s + accounts[i]->calculateCosts();
    return s;
}

void main()
{
    Account accounts = new Account[2];
    accounts[0] = new CurrentAccount();
    accounts[1] = new SavingAccount();
    int s = calculateCost(accounts);
    …
}
```

Trừu tượng: lớp trừu tượng

- Lớp trừu tượng
  - chỉ ra đặc điểm chung của các lớp con
  - không thể có thể hiện/ đối tượng
- Lớp cụ thể
  - chứa một mô tả đầy đủ các đối tượng trong thế giới thực
  - dự kiến sẽ có các thể hiện/ đối tượng



Trừu tượng: phương thức trừu tượng

- Một phương thức nên được xác định ở mức trừu tượng cao nhất có thể
  - Ở mức này, phương thức có thể là trừu tượng (tức là không có triển khai)
  - Trong trường hợp này, lớp cũng trừu tượng
  - Nếu một lớp có một phương thức trừu tượng, ít nhất một trong các lớp con của nó phải triển khai phương thức này
  - Tất cả các phương thức của một lớp ở cuối cây kế thừa phải là cụ thể