

Mục tiêu

Kết thúc chương, học viên có thể:

- Định nghĩa Lập trình hướng Đối tượng (Object-oriented Programming).
- Nhận thức về Trừu tượng hóa Dữ liệu (Data Abstraction).
- Định nghĩa một Lớp (Class).
- Định nghĩa một Đối tượng (Object).
- Nhận thức được sự khác biệt giữa Lớp và Đối tượng.
- Nhận thức được sự cần thiết đối với Thiết lập (Construction) và Hủy (Destruction).
- Định nghĩa tính Bền vững (Persistence).
- Hiểu biết về tính Thừa kế (Inheritance).
- Định nghĩa tính Đa hình (Polymorphism).
- Liệt kê những thuận lợi của phương pháp hướng Đối tượng.

1.1 Giới thiệu về Lập trình hướng Đối tượng (Object-oriented Programming)

Lập trình hướng Đối tượng (OOP) là một phương pháp thiết kế và phát triển phần mềm. Những ngôn ngữ OOP không chỉ bao gồm cú pháp và một trình biên dịch (compiler) mà còn có một môi trường phát triển toàn diện. Môi trường này bao gồm một thư viện được thiết kế tốt, thuận lợi cho việc sử dụng các đối tượng.

Đối với một ngôn ngữ lập trình hỗ trợ OOP thì việc triển khai kỹ thuật lập trình hướng đối tượng sẽ dễ dàng hơn. Kỹ thuật lập trình hướng đối tượng cải tiến việc phát triển các hệ thống phần mềm. Kỹ thuật ấy đề cao nhân tố chức năng (functionality) và các mối quan hệ dữ liệu.

OOP là phương thức tư duy mới để giải quyết vấn đề bằng máy tính. Để đạt kết quả, lập trình viên phải nắn vấn đề thành một thực thể quen thuộc với máy tính. Cách tiếp cận hướng đối tượng cung cấp một giải pháp toàn vẹn để giải quyết vấn đề.

Hãy xem xét một tình huống cần được triển khai thành một hệ thống trên máy vi tính: việc mua bán xe hơi. Vấn đề vi tính hóa việc mua bán xe hơi bao gồm những gì?

Những yếu tố rõ ràng nhất liên quan đến việc mua bán xe hơi là:

- 1) Các kiểu xe hơi (model).
- 2) Nhân viên bán hàng.
- 3) Khách hàng.

Những hoạt động liên quan đến việc mua bán:

- 1) Nhân viên bán hàng đưa khách hàng tham quan phòng trưng bày.
- 2) Khách hàng chọn lựa một xe hơi.
- 3) Khách hàng đặt hóa đơn.

- 4) Khách hàng trả tiền.
- 5) Chiếc xe được trao cho khách hàng.

Mỗi vấn đề được chia ra thành nhiều yếu tố, được gọi là các Đối tượng (Objects) hoặc các Thực thể (Entities). Chẳng hạn như ở ví dụ trên, khách hàng, xe hơi và nhân viên bán hàng là những đối tượng hoặc thực thể.

Lập trình viên luôn luôn cố gắng tạo ra những kịch bản (scenarios) thật quen thuộc với những tình huống đời sống thực. Bước thứ nhất trong đường hướng này là làm cho máy tính liên kết với những đối tượng thế giới thực.

Tuy nhiên, máy tính chỉ là một cỗ máy. Nó chỉ thực hiện những công việc được lập trình mà thôi. Vì thế, trách nhiệm của lập trình viên là cung cấp cho máy tính những thông tin theo cách thức mà nó cũng nhận thức được cùng những thực thể như chúng ta nhận thức.

Đó chính là lãnh vực của kỹ thuật hướng đối tượng. Chúng ta sử dụng kỹ thuật hướng đối tượng để ánh xạ những thực thể chúng ta gặp phải trong đời sống thực thành những thực thể tương tự trong máy tính.

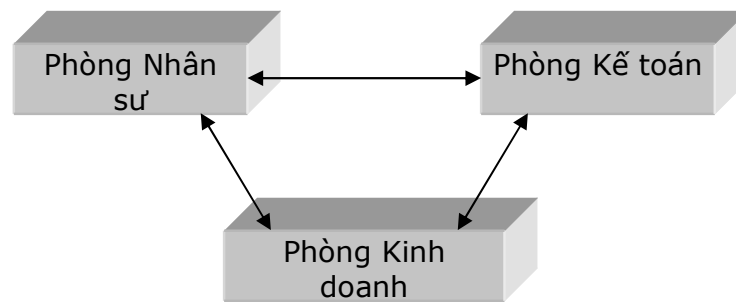
Phát triển phần mềm theo kỹ thuật lập trình hướng đối tượng có khả năng giảm thiểu sự lẫn lộn thường xảy ra giữa hệ thống và lãnh vực ứng dụng.

Lập trình hướng đối tượng đề cập đến dữ liệu và thủ tục xử lý dữ liệu theo quan điểm là một đối tượng duy nhất. Lập trình hướng đối tượng xem xét dữ liệu như là một thực thể hay là một đơn vị độc lập, với bản chất riêng và những đặc tính của thực thể ấy. Bây giờ chúng ta hãy khảo sát những hạn từ 'đối tượng' (object), 'dữ liệu' (data) và 'phương thức' (method).

Có nhiều loại đối tượng khác nhau. Chúng ta có thể xem các bộ phận khác nhau trong một cơ quan là các đối tượng. Điển hình là một cơ quan có những bộ phận liên quan đến việc quản trị, đến việc kinh doanh, đến việc kế toán, đến việc tiếp thị ... Mỗi bộ phận có nhân sự riêng, các nhân sự được trao cho những trách nhiệm rõ ràng. Mỗi bộ phận cũng có những dữ liệu riêng chẳng hạn như thông tin cá nhân, bảng kiểm kê, những thể thức kinh doanh, hoặc bất kỳ dữ liệu nào liên quan đến chức năng của bộ phận đó. Rõ ràng là một cơ quan được chia thành nhiều bộ phận thì việc quản trị nhân sự và những hoạt động doanh nghiệp dễ dàng hơn. Nhân sự của cơ quan điều khiển và xử lý dữ liệu liên quan đến bộ phận của mình.

Chẳng hạn như bộ phận kế toán chịu trách nhiệm về lương bổng đối với cơ quan. Nếu một người ở đơn vị tiếp thị cần những chi tiết liên quan đến lương bổng của đơn vị mình, người ấy chỉ cần liên hệ với bộ phận kế toán. Một người có thẩm quyền trong bộ phận kế toán sẽ cung cấp thông tin cần biết, nếu như thông tin ấy có thể chia sẻ được. Một người không có thẩm quyền từ một bộ phận khác thì không thể truy cập dữ liệu, hoặc không thể thay đổi làm hư hỏng dữ liệu. Ví dụ này minh chứng các đối tượng là hữu dụng trong việc phân cấp và tổ chức dữ liệu.

Hình 1.1 Minh họa cấu trúc của một cơ quan điển hình.



Hình 1.1

Khái niệm về đối tượng có thể được mở rộng đến hầu hết các lãnh vực đời sống, và hơn nữa, đến thế giới lập trình. Bất kỳ ứng dụng nào đều có thể được định nghĩa theo hạn từ thực thể hoặc đối tượng để tạo ra tiến trình xử lý mô phỏng theo tiến trình xử lý mà con người nghĩ ra.

Phương pháp giải quyết 'top-down' (từ trên xuống) cũng còn được gọi là 'lập trình hướng cấu trúc' (structured programming). Nó xác định những chức năng chính của một chương trình và những chức năng này được phân thành những đơn vị nhỏ hơn cho đến mức độ thấp nhất. Bằng kỹ thuật này, các chương trình được cấu trúc theo hệ thống phân cấp các module. Mỗi một module có một đầu vào riêng và một đầu ra riêng. Trong mỗi module, sự điều khiển có chiều hướng đi xuống theo cấu trúc chứ không có chiều hướng đi lên.

Phương pháp OOP cố gắng quản lý việc thừa kế phức tạp trong những vấn đề đời thực. Để làm được việc này, phương thức OOP che giấu một vài thông tin bên trong các đối tượng. OOP tập trung trước hết trên dữ liệu. Rồi gắn kết các phương thức thao tác trên dữ liệu, việc này được xem như là phần thừa kế của việc định nghĩa dữ liệu. Bảng 1.1 minh họa sự khác biệt giữa hai phương pháp:

Phương pháp Top-Down	OOP
Chúng ta sẽ xây dựng một khách sạn.	Chúng ta sẽ xây dựng một tòa nhà 10 tầng với những dãy phòng trung bình, sang trọng, và một phòng họp lớn.
Chúng ta sẽ thiết kế các tầng lầu, các phòng và phòng họp.	Chúng ta sẽ xây dựng một khách sạn với những thành phần trên.

Bảng 1.1 Một ví dụ về hai phương pháp giải quyết OOP và Structured

1.2 Trừu tượng hóa dữ liệu (Data Abstraction)

Khi một lập trình viên phải phát triển một chương trình ứng dụng thì **không có nghĩa là người ấy lập tức viết mã cho ứng dụng ấy. Trước hết, người ấy phải nghiên cứu ứng dụng và xác định những thành phần tạo nên ứng dụng. Kế tiếp, người ấy phải xác định những thông tin cần thiết về mỗi thành phần.**

Hãy khảo sát chương trình ứng dụng cho việc mua bán xe hơi nói trên. Chương trình phải xuất hóa đơn cho những xe hơi đã bán cho khách hàng. Để xuất một hóa đơn, chúng ta cần những thông tin chi tiết về khách hàng. Vậy bước thứ nhất là xác định những đặc tính của khách hàng.

Một vài đặc tính gắn kết với khách hàng là:

- Tên.
- Địa chỉ.
- Tuổi.
- Chiều cao.
- Màu tóc.

Từ danh sách kể trên, chúng ta xác định những đặc tính thiết yếu đối với ứng dụng. Bởi vì chúng ta đang đề cập đến những khách hàng mua xe, vì thế những chi tiết thiết yếu là:

- Tên.
- Địa chỉ.

Còn những chi tiết khác (chiều cao, màu tóc ...) là không quan trọng đối với ứng dụng. Tuy nhiên, nếu chúng ta phát triển một ứng dụng hỗ trợ cho việc điều tra tội phạm thì những thông tin chẳng hạn như màu tóc là thiết yếu.

Bên cạnh những chi tiết về khách hàng, những thông tin sau cũng cần thiết:

- Kiểu xe được bán.
- Nhân viên nào bán xe.

Bên cạnh những đặc tính của khách hàng, xe hơi và nhân viên bán hàng, chúng ta cũng cần liệt kê những hành động được thực hiện.

Công việc xuất hóa đơn đòi hỏi những hành động sau:

- Nhập tên của khách hàng.
- Nhập địa chỉ của khách hàng.
- Nhập kiểu xe.
- Nhập tên của nhân viên bán xe.
- Xuất hóa đơn với định dạng đòi hỏi.

Khung thông tin bên dưới cho thấy những thuộc tính và những hành động liên quan đến một hóa đơn:

Các thuộc tính
Tên của khách hàng
Địa chỉ của khách hàng
Kiểu xe bán
Nhân viên bán xe

Các hành động
Nhập tên
Nhập địa chỉ
Nhập kiểu xe

Nhập tên nhân viên bán xe
Xuất hóa đơn

Định nghĩa

Sự trừu tượng hóa dữ liệu là tiến trình xác định và nhóm các thuộc tính và các hành động liên quan đến một thực thể đặc thù, xét trong mối tương quan với ứng dụng đang phát triển.

Tiếp theo, chúng ta muốn ứng dụng tính toán tiền hoa hồng cho nhân viên bán hàng.

Những thuộc tính liên kết với nhân viên bán hàng có tương quan với ứng dụng này là:

- Tên.
- Số lượng xe bán được.
- Tiền hoa hồng.

Những hành động đòi buộc đối với công việc này là:

- Nhập tên nhân viên bán xe.
- Nhập số lượng xe bán được.
- Tính tiền hoa hồng kiếm được.

Những thuộc tính
Tên
Số lượng xe bán được
Tiền hoa hồng

Những hành động
Nhập tên
Nhập số lượng xe bán được
Tính tiền hoa hồng

Như thế, việc trừu tượng hóa dữ liệu tra đặt ra câu hỏi 'Đâu là những thuộc tính và những hành động cần thiết cho một vấn đề đặt ra?'

1.2.1 Những thuận lợi của việc Trừu tượng hóa

Những thuận lợi của việc Trừu tượng hóa là:

- Tập trung vào vấn đề.
- Xác định những đặc tính thiết yếu và những hành động đòi hỏi.
- Giảm thiểu những chi tiết không cần thiết.

Việc trừu tượng hóa dữ liệu là cần thiết, bởi vì không thể mô phỏng tất cả các hành động và các thuộc tính của một thực thể. Vấn đề mấu chốt là tập trung đến những hành vi cốt

yếu và áp dụng chúng trong ứng dụng.

Chẳng hạn như khách hàng hoặc nhân viên bán hàng cũng có thể thực hiện những hành động sau:

- Người ấy đi lại.
- Người ấy nói chuyện.

Tuy nhiên, những hành động này không liên quan đến ứng dụng. Việc trừu tượng hóa dữ liệu sẽ loại bỏ chúng.

1.3 Lớp (Class)

Trong ứng dụng mua bán xe, chúng ta đã xác định các thuộc tính và các hành động cần có để xuất một hóa đơn cho một khách hàng.

Các hành động và các thuộc tính này là chung cho mọi khách hàng mua xe. Ví thể, chúng có thể được nhóm lại trong một thực thể đơn nhất gọi là một 'lớp'.

Hãy khảo sát lớp có tên là 'khách hàng' dưới đây. Lớp này bao gồm mọi thuộc tính và hành động đòi hỏi đối với một khách hàng.

Lớp Khách hàng
Tên khách hàng
Địa chỉ khách hàng
Kiểu xe được bán
Nhân viên bán xe
Nhập tên
Nhập địa chỉ
Nhập kiểu xe được bán
Nhập tên nhân viên bán xe
Xuất hóa đơn

Định nghĩa

Một **lớp** định nghĩa một thực thể theo những thuộc tính và những hành động chung. *Hoặc* Những thuộc tính và những hành động chung của một thực thể được nhóm lại để tạo nên một đơn vị duy nhất gọi là một **lớp**. *Hoặc* Một **lớp** là một sự xác định cấp chung loại của các thực thể giống nhau.

Một lớp là một mô hình khái niệm về một thực thể. Nó mang tính cách tổng quát chứ không mang tính cách đặc thù.

Khi định nghĩa một lớp, chúng ta muốn phát biểu rằng một lớp sẽ phải có một tập hợp các thuộc tính và các hành động riêng. Chẳng hạn như một định nghĩa lớp dưới đây:

Lớp Con người

Tên
Chiều cao
Màu tóc
Viết
Nói

Lớp này định nghĩa thực thể 'Con người'. Mọi thực thể thuộc kiểu 'Con người' sẽ đều có những đặc tính và những hành động như đã được định nghĩa.

Một khi một lớp đã được định nghĩa, chúng ta biết được những thuộc tính và những hành động của những thực thể 'trông giống' như lớp này. Vì thế, tự bản chất một lớp là một nguyên mẫu (prototype).

Một ví dụ khác về một lớp liên quan đến việc mua bán xe hơi như sau:

Lớp Nhân viên bán hàng
Tên
Số lượng xe bán được
Tiền hoa hồng
Nhập tên
Nhập số lượng xe bán được
Tính tiền hoa hồng

Lớp trên định nghĩa các thuộc tính và các hành động đặc trưng cho mọi nhân viên bán xe hơi.

1.4 Đối tượng (Object)

Một lớp là một nguyên mẫu phác họa những thuộc tính và những hành động khả thể của một thực thể. Để có thể sử dụng thực thể mà lớp định nghĩa, chúng ta phải tạo một 'đối tượng' từ lớp đó.

Lớp là một khái niệm, còn đối tượng là một mẫu thực được định nghĩa bởi lớp.

Hãy khảo sát lớp 'Khách hàng' được định nghĩa trên. Lớp này định nghĩa mọi thuộc tính và hành động gắn liền với một khách hàng.

Khi một người mua một xe hơi ở một cửa hàng, cửa hàng ấy có một khách hàng mới. Vào thời điểm ấy, một đối tượng giống như lớp 'Khách hàng' được tạo ra. Đối tượng này sẽ phải có những giá trị thực đối với các thuộc tính 'Tên', 'Địa chỉ', 'Kiểu xe' ...

Chẳng hạn như một khách hàng có tên là 'Mark', sống ở 'London' đã mua một xe kiểu 'Honda Civic' từ nhân viên bán hàng tên là 'Tom'. Như thế, 'Mark' là một đối tượng của kiểu 'Khách hàng'.

Định nghĩa: Một đối tượng là một trường hợp của một lớp.

Một đối tượng là một thực thể cụ thể (thông thường bạn có thể sờ chạm, xem thấy và cảm nhận).

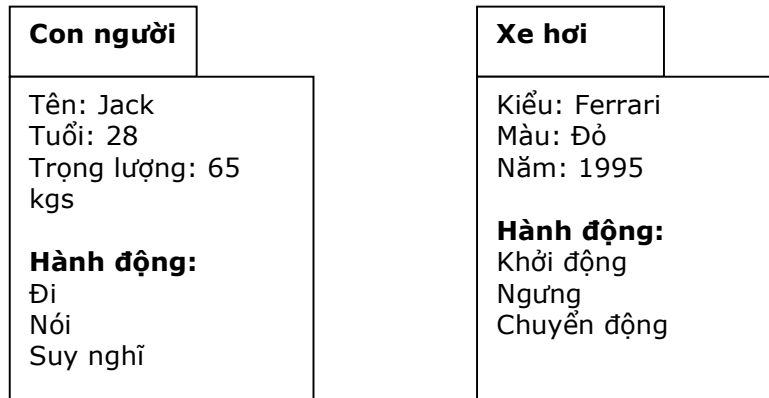
Kể từ lúc một đối tượng hiện hữu, những thuộc tính của nó là những giá trị xác định, và

những hành động được định nghĩa cho đối tượng này được thực thi.

Trong mỗi một đối tượng, các khía cạnh sau đây được xác định rõ:

- Tình trạng (state).
- Thái độ (behavior).
- Chân tính (identity).

Hình 1.2 trình bày hai đối tượng đời thực.



Hình 1.2: Một đối tượng Con người và một đối tượng Xe hơi

Mỗi đối tượng có những đặc tính riêng mô tả đối tượng ấy là gì, hoặc hành động ra sao.

Chẳng hạn như những thuộc tính của một đối tượng 'Con người' sẽ là:

- Tên.
- Tuổi.
- Trọng lượng.

Những thuộc tính của một đối tượng 'Xe hơi' sẽ là:

- Màu sắc.
- Kiểu xe.
- Năm.

Một đối tượng cũng thực hiện một số hành động. Một xe hơi có khả năng thực hiện những hành động sau:

- Khởi động.
- Ngừng.
- Chuyển động.

Để chuyển đổi giữa các đối tượng lập trình và các đối tượng đời thực, cần phải kết hợp các thuộc tính và các hành động của một đối tượng.

1.4.1 Thuộc tính

Chúng ta xác định các thuộc tính và các hành động để định nghĩa một lớp.

Một khi các thuộc tính được gán cho các giá trị, chúng mô tả một đối tượng. Hãy khảo sát lớp sau:

Các thuộc tính của lớp Khách hàng
Tên của khách hàng
Địa chỉ của khách hàng
Kiểu xe được bán
Nhân viên đã bán xe

Khi thuộc tính 'Tên' được gán cho giá trị 'Mark' thì nó mô tả một đối tượng xác định được tạo từ lớp 'Khách hàng'.

Định nghĩa

Một **thuộc tính** là một đặc tính mô tả một đối tượng.

Như thế, các thuộc tính nắm giữ các giá trị dữ liệu trong một đối tượng, chúng định nghĩa một đối tượng đặc thù.

Bởi vì một lớp là một nguyên mẫu cho nên các thuộc tính trong một lớp không thể nắm giữ các giá trị. Một thuộc tính có thể được gán một giá trị chỉ sau khi một đối tượng dựa trên lớp ấy được tạo ra.

Để có thể lưu giữ những chi tiết của một khách hàng, một trường hợp (đối tượng) của lớp 'Khách hàng' phải được tạo ra. Các thuộc tính của một đối tượng hiện hữu chỉ khi đối tượng ấy được tạo ra.

Mọi đối tượng của một lớp phải có cùng các thuộc tính.

Khảo sát ví dụ sau:

Các thuộc tính của lớp Con người		Đối tượng được tạo từ lớp Con người
Tên	=	Mark
Chiều cao	=	6 ft. 1 in.
Màu tóc	=	Black

1.4.2 Hoạt động (Operation)

Các hành động khả thi, như được định nghĩa trong một lớp, được gọi là 'các hoạt động'.

Định nghĩa

Một **hoạt động** là một dịch vụ được đòi hỏi của một đối tượng.

Các hoạt động xác định các hành động được đòi hỏi của một đối tượng được tạo ra từ một lớp. Chẳng hạn như chúng ta không thể đòi hỏi một hoạt động 'Mua một xe hơi khác' của một đối tượng được tạo ra từ lớp 'Khách hàng'.

Một lớp chỉ là một nguyên mẫu. Vì thế, trong một lớp một hoạt động chỉ được định nghĩa. Còn việc áp dụng hoạt động ấy chỉ xảy ra nơi các đối tượng riêng rẽ. Chẳng hạn như hoạt động 'Nhập Tên' mà lớp "Khách hàng" định nghĩa có thể được thực hiện nơi một đối tượng nào đó.

Tập hợp các hoạt động được yêu cầu cho tất cả các đối tượng trong một lớp.

1.4.3 Phương thức (Method)

Các hoạt động định nghĩa các hành động khả thi có thể được yêu cầu của một đối tượng. Một phương thức là sự thực thi thực tế của một hoạt động.

Định nghĩa

Phương thức là một sự xác định về cách thức một hoạt động được yêu cầu được thực thi.

Các phương thức xác định cách thức thao tác trên các dữ liệu của một đối tượng. Bởi vì phương thức là sự thực thi thực tế một hoạt động, cho nên nó có thể được áp dụng cho một đối tượng. Một phương thức là một thuật toán từng bước (step-by-step) xác định điều gì được thực hiện khi hoạt động ấy được yêu cầu.

Hãy khảo sát những hoạt động chung của một thực thể thuộc loại 'Con người': Đi, Nói. Chỉ khi một đối tượng cụ thể của loại 'Con người' được tạo ra thì các hành động 'Đi', 'Nói' mới được thực thi.

1.4.4 Thông điệp (Message)

Để yêu cầu một hoạt động cụ thể nào đó được thực hiện, một thông điệp được gửi tới đối tượng, thông điệp này định nghĩa hoạt động.

Định nghĩa

Một **thông điệp** là một lời yêu cầu một hoạt động.

Khi một đối tượng nhận được một thông điệp, nó thực hiện một phương thức tương ứng.

Chẳng hạn, một đối tượng được tạo ra từ lớp 'Khách hàng' để nhập tên của người sử dụng. Khi đối tượng nhận được thông điệp, nó tìm và thực thi phương thức 'Nhập tên'.

Trong trường hợp một công ty, mỗi bộ phận được coi là một đối tượng. Những thông tin được chuyển tới và được đón nhận từ mỗi bộ phận (hoặc qua thông báo liên bộ phận, hoặc qua những chỉ thị miệng) tạo nên những thông điệp giữa các đối tượng. Những chỉ thị này có thể được chuyển dịch thành những lời gọi hàm trong một chương trình.



Hình 1.3 Các đối tượng gửi thông điệp cho nhau

Trong hình 1.3, 'Kinh doanh' và 'Kế toán' là hai bộ phận khác nhau trong một công ty. Hai bộ phận này được coi là hai đối tượng khác nhau. Thông tin được truyền đi và được đón nhận giữa các bộ phận tạo nên các thông điệp giữa các đối tượng.

1.4.5 Sự kiện (Event)

Một sự kiện là một sự việc xảy ra cho một đối tượng tại một thời điểm. Để đáp ứng lại sự kiện ấy, đối tượng sẽ thực hiện một hoặc nhiều phương thức.

Nói cách khác, một sự kiện là một tác nhân mà đối tượng này gây ra cho một đối tượng khác. Chẳng hạn như click chuột trái trên một nút.

Để hiểu rõ hơn các sự kiện, hãy khảo sát ví dụ sau từ đời thực:

'Một người sẽ **thét lên** khi **bị thọc** bằng một vật nhọn'.

'**Thọc**' là sự kiện gây ra sự phản ứng là '**thét lên**'.

Trong máy tính, một người sử dụng nhấn một nút trên bàn phím là một sự kiện chung. Sự phản hồi đối với sự kiện này là việc hiển thị ký tự tương ứng trên màn hình.

1.5 Lớp và Đối tượng

Có một sự khác biệt thực sự giữa một lớp và một đối tượng. Cần nhận thức rõ sự khác biệt này.

Một lớp định nghĩa một thực thể, trong khi đó một đối tượng là một trường hợp của thực thể ấy.

Đối tượng là một mô hình thực, trong khi lớp là một mô hình khái niệm - định nghĩa tất cả các thuộc tính và các phương thức được đòi hỏi của một đối tượng.

Tất cả các đối tượng thuộc về cùng một lớp có cùng các thuộc tính và các phương thức.

Một lớp là một nguyên mẫu của một đối tượng. Nó xác định các hành động khả thi và các thuộc tính cần thiết cho một nhóm các đối tượng đặc thù.

1.6 Thiết lập (Construction) và Hủy (Destruction)

1.6.1 Thiết lập

Một lớp chỉ cung cấp những định nghĩa về các thuộc tính và các phương thức khả thi. Các thuộc tính và các phương thức có thể được truy cập chỉ khi một đối tượng dựa trên một lớp được tạo ra.

Khi một đối tượng mới được tạo, các thuộc tính của nó trở nên hiện thực và có thể được gán giá trị. Tương tự, các phương thức được định nghĩa cũng được áp dụng.

Định nghĩa

Thiết lập là một tiến trình hiện thực hóa một đối tượng.

Hàm **thiết lập** là một phương thức đặc biệt phải được gọi trước khi sử dụng bất kỳ phương thức nào trong một lớp. Hàm Thiết lập khởi tạo các thuộc tính, và cấp phát bộ nhớ trong máy tính khi cần thiết.

Mỗi một lớp có một hàm thiết lập.

Khảo sát lại trường hợp cửa hàng bán xe hơi. Ngay từ lúc đầu chỉ định nghĩa các lớp. Cho đến khi một khách hàng mua một xe hơi tại cửa hàng thì một đối tượng mới giống như lớp 'Khách hàng' mới được tạo.

Khi đối tượng này được tạo, một số khoảng trống bộ nhớ được cấp phát cho những thuộc tính của nó để lưu trữ các giá trị được gán cho các thuộc tính ấy ('Tên', 'Địa chỉ' ...). **Hàm thiết lập thực hiện việc cấp phát này.** Vào lúc này, mọi thuộc tính và phương thức của đối tượng sẵn sàng để sử dụng.

Tương tự như trường hợp một học sinh nhập học tại một trường học. Khi một học sinh nhập học, một vài hành động được thực hiện để nhận học sinh ấy vào trường. Đó là:

- Xếp lớp cho học sinh ấy.
- Ghi tên học sinh ấy vào danh sách.
- Xếp chỗ ngồi.

Đây là những hành động đồng loạt được thực hiện ngay lúc bắt nhập học. Chúng tương tự với những hành động mà hàm thiết lập của một đối tượng thực hiện.

1.6.2 Hủy

Khi một đối tượng không còn cần thiết nữa thì nó sẽ bị hủy bỏ.

Sẽ lãng phí tài nguyên, chẳng hạn như bộ nhớ, nếu như tiếp tục để cho một đối tượng tồn tại một khi nó không còn cần thiết.

Định nghĩa

Hàm **Hủy** là một phương thức đặc biệt được dùng để hủy bỏ một đối tượng.

Tiến trình **Hủy** tiêu hủy một đối tượng và giải phóng khoảng trống bộ nhớ mà hàm thiết lập đã cấp phát cho nó. Hàm **Hủy** cũng triệt tiêu khả năng truy cập đến đối tượng ấy.

Một khi một đối tượng bị hủy thì các thuộc tính của nó không thể được truy cập, cũng như không một phương thức nào có thể được thực thi.

Chẳng hạn, trong trường hợp bán xe hơi, một khi nhân viên bán hàng bỏ nghề, những chi tiết của người ấy không còn liên hệ. Vì thế, đối tượng tương ứng sẽ bị hủy. Điều này giải phóng bộ nhớ đã cấp phát cho nhân viên bán hàng ấy. Khoảng trống này giờ đây có thể được tái sử dụng.

Hãy xem xét ví dụ về trường học trên đây. Khi một học sinh thôi học, tên của học sinh ấy bị loại ra khỏi danh sách, và khoảng trống được giải phóng có thể được tái cấp phát.

Các hành động đồng loạt này tương tự với công việc của hàm hủy đối với một đối tượng.

1.7 Tính Bền vững (Persistence)

Hãy khảo sát trường hợp bán xe hơi. Những chi tiết của khách hàng được lưu trữ ngay khi xe hơi đã được phân phối. Việc duy trì dữ liệu vẫn cần thiết cho đến khi dữ liệu được chỉnh sửa hoặc hủy bỏ chính thức.

Định nghĩa

Tính Bền vững là khả năng lưu trữ dữ liệu của một đối tượng ngay cả khi đối tượng ấy không còn tồn tại.

Cửa hàng bán xe lưu trữ chi tiết khách hàng vào một file. Những chi tiết này sẽ tồn tại trong file cho đến khi chúng bị hủy, hoặc bản thân file bị hủy.

Chúng ta đừng chạm tính bền vững mỗi ngày. Hãy xem việc sáng tác một bài thơ. Bài thơ là dữ liệu tồn tại trong tâm trí của nhà thơ. Bao lâu nhà thơ còn tồn tại thì bấy lâu bài thơ còn tồn tại. Nếu bài thơ muốn tồn tại ngay cả sau khi nhà thơ qua đời thì nó phải được viết ra giấy.

Bài thơ được viết ra giấy tạo nên sự bền vững. Bài thơ sẽ tồn tại bao lâu văn bản ấy còn được duy trì. Bài thơ ấy không còn tồn tại khi tờ giấy ấy bị xé rách, hoặc chữ nghĩa bị xóa đi.

1.8 Tính Đóng gói dữ liệu

Tiến trình trừu tượng hóa dữ liệu hỗ trợ cho việc xác định những thuộc tính và những phương thức thiết yếu.

Thông thường, các đối tượng sử dụng những thuộc tính và những phương thức không được đòi hỏi bởi người sử dụng đối tượng.

Chẳng hạn như trong trường hợp lớp 'Khách hàng'. Lớp ấy có một phương thức xuất hóa đơn. Giả sử rằng khi hóa đơn được xuất, một trong những chi tiết được in ra trên hóa đơn là ngày phân phối. Tuy nhiên chúng ta không biết thuộc tính nào qua đó chúng ta có thể xác định thông tin này.

Ngày phân phối được phát sinh bên trong đối tượng, và được hiển thị trên hóa đơn. Như thế người sử dụng không nhận thức về cách thức mà ngày phân phối được hiển thị.

Ngày phân phối có thể được xử lý theo một trong những cách sau:

- Đó là một giá trị được tính toán - Chẳng hạn, 15 ngày kể từ ngày đặt hàng.
- Đó là một giá trị cố định - Xe hơi được phân phối vào ngày mùng 2 mỗi tháng.

Đối tượng sử dụng những thuộc tính và những phương thức mang tính nội bộ. Bởi vì những thuộc tính và những phương thức có thể được che khuất khỏi tầm nhìn. Các đối tượng khác và những người sử dụng không nhận thức được các thuộc tính và / hoặc các phương thức như thế có tồn tại hay không. **Tiến trình che giấu các thuộc tính, các phương thức, hoặc các chi tiết của việc thi hành được gọi là 'đóng gói' (encapsulation).**

Định nghĩa

Đóng gói là tiến trình che giấu việc thực thi những chi tiết của một đối tượng đối với người sử dụng đối tượng ấy.

Việc đóng gói phân tách những khía cạnh có thể truy cập từ bên ngoài với những khía cạnh chỉ được sử dụng trong nội bộ của đối tượng.

Điểm thuận lợi của việc đóng gói là có thể tạo ra bất kỳ thuộc tính hay phương thức cần thiết để đáp ứng đòi hỏi công việc khi xây dựng một lớp. Mặt khác, chỉ những thuộc tính và / hoặc những phương thức có thể được truy cập từ bên ngoài lớp là trông thấy.

Một ví dụ khác về việc đóng gói là lớp 'Nhân viên bán hàng' đã được định nghĩa ở trên. Khi phương thức tính tiền hoa hồng được thực thi, người sử dụng không biết chi tiết của việc tính toán. Tất cả những gì họ biết chỉ là tổng số tiền hoa hồng mà họ phải trả cho nhân viên bán hàng.

Một trường hợp về đóng gói mà chúng ta gặp trong đời sống hằng ngày là việc giao dịch kinh doanh ở một cửa hàng. Khách hàng yêu cầu sản phẩm X. Họ được trao cho sản phẩm X, và họ phải trả tiền cho sản phẩm ấy. Sau khi khách hàng yêu cầu sản phẩm, người bán hàng thực hiện những hành động sau:

- Kiểm tra mặt hàng trên kệ hàng.

- Giảm số lượng mặt hàng trong bảng kiểm kê sau khi bán.

Tuy nhiên, khách hàng không được biết những chi tiết này.

1.9 Tính thừa kế

Hãy khảo sát các lớp sau:

Lớp Sinh viên	Lớp Nhân viên	Lớp Khách hàng
Tên	Tên	Tên
Địa chỉ	Địa chỉ	Địa chỉ
Điểm môn 1	Lương	Kiểu xe đã bán
Điểm môn 2	Chức vụ	Nhập tên
Nhập tên	Nhập tên	Nhập địa chỉ
Nhập địa chỉ	Nhập địa chỉ	Nhập kiểu xe
Nhập điểm	Nhập chức vụ	Xuất hóa đơn
Tính tổng điểm	Tính lương	

Trong tất cả ba lớp, chúng ta thấy có một vài thuộc tính và hoạt động chung. Chúng ta muốn nhóm những thuộc tính và những hoạt động ấy lại, và định nghĩa chúng trong một lớp 'Người'.

Lớp Người
Tên
Địa chỉ
Nhập tên
Nhập địa chỉ

Ba lớp 'Sinh viên', 'Nhân viên' và 'Khách hàng' tạo nên lớp 'Người'. Nói cách khác, ba lớp ấy có tất cả các thuộc tính và các phương thức của lớp 'Người', ngoài ra chúng còn có những thuộc tính và những phương thức riêng.

Chúng ta cần phải định nghĩa lớp 'Người' và sử dụng nó trong khi định nghĩa các lớp 'Sinh viên', 'Nhân viên' và 'Khách hàng'.

Chúng ta xây dựng một lớp 'Người' với những thuộc tính và những hoạt động như đã trình bày ở hình trên. Kế tiếp, chúng ta xây dựng lớp 'Khách hàng' bao gồm lớp 'Người' cộng với những thuộc tính và những phương thức riêng.

Chúng ta có thể định nghĩa các lớp 'Sinh viên' và 'Nhân viên' theo cùng cách thức trên. Như thế, cả ba lớp 'Khách hàng', 'Sinh viên' và 'Nhân viên' đều chia sẻ những thuộc tính và những phương thức mà lớp 'Người' cung cấp.

Lớp Sinh viên	Lớp Nhân viên	Lớp Khách hàng
Điểm môn 1	Lương	Kiểu xe bán được
Điểm môn 2	Chức vụ	Nhập kiểu xe
Nhập điểm	Nhập chức vụ	Xuất hóa đơn

tính tổng điểm	Tính lương	
----------------	------------	--

Theo ngôn ngữ hướng đối tượng, lớp 'Khách hàng' được gọi là thừa kế lớp 'Người'.

Định nghĩa: Tính thừa kế cho phép một lớp chia sẻ các thuộc tính và các phương thức được định nghĩa trong một hoặc nhiều lớp khác.

Có hai khái niệm quan trọng khác liên kết với tính thừa kế. Lớp 'Khách hàng' là lớp 'Người' cộng thêm cái khác. Như thế, lớp 'Khách hàng' có tất cả các thuộc tính và các phương thức được định nghĩa trong lớp 'Người' cộng với các thuộc tính và các hoạt động của riêng nó.

Trong ví dụ này, lớp 'Khách hàng' được gọi là 'lớp con' (subclass).

Định nghĩa: Lớp thừa hưởng từ một lớp khác được gọi là Subclass.

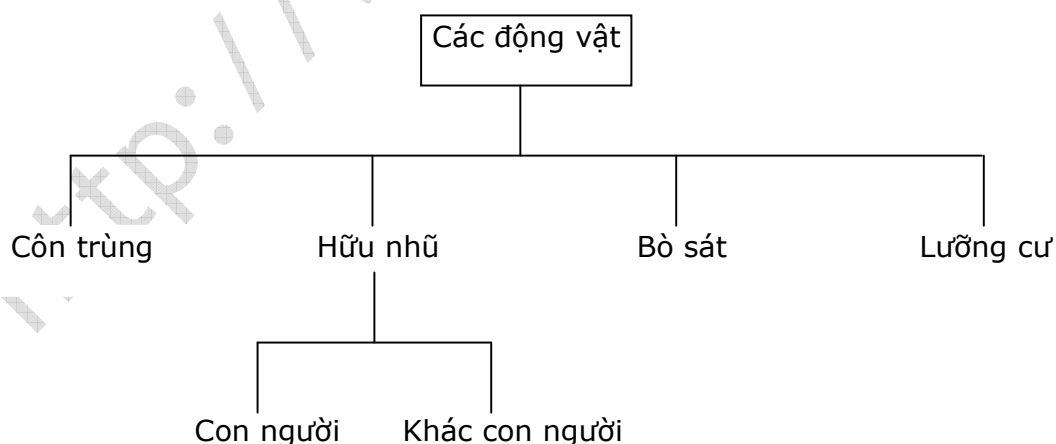
Trong ví dụ trên, lớp 'Người' được coi là 'lớp trên' (superclass).

Định nghĩa: Một Superclass là một lớp mà các đặc tính của nó được một lớp khác thừa hưởng.

Hãy xem xét ví dụ về lớp 'Các động vật' ở hình 1.4. 'Các động vật' là lớp trên cùng mà các lớp khác kế thừa. Chúng ta có một dãy các lớp trung gian - 'Côn trùng', 'Hữu nhũ', 'Bò sát', 'Lưỡng cư' - mà dãy các lớp dưới kế thừa.

Các lớp 'Côn trùng', 'Hữu nhũ', 'Bò sát', 'Lưỡng cư' là những lớp con của lớp trên 'Các động vật'. Như thế, những lớp này có tất cả những thuộc tính và các hoạt động của lớp 'Các động vật', cộng thêm những thuộc tính và những phương thức của riêng chúng.

Lớp 'Hữu nhũ' là lớp mà các lớp 'Con người' và 'Khác con người' thừa kế. Như thế, các lớp 'Con người' và 'Khác con người' là các lớp con của lớp trên 'Hữu nhũ'.



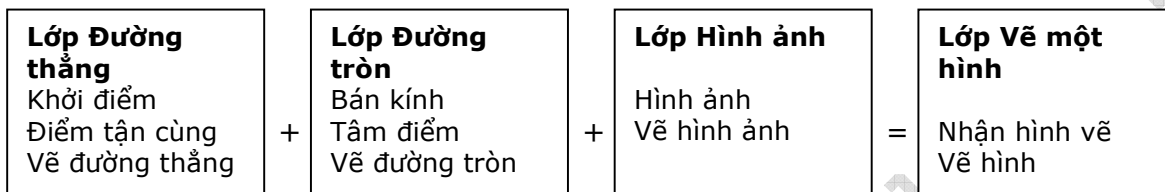
Hình 1.4 Tính thừa kế

1.9.1 Tính Đa Thừa kế

Trong tất cả các ví dụ trên, một lớp thừa kế chỉ từ một lớp. Ngay cả trong ví dụ thừa kế về các loại phương tiện di chuyển, mỗi lớp con chỉ có một lớp cha. Trường hợp như thế gọi là 'thừa kế đơn' (single inheritance).

Trong 'đa thừa kế', một lớp con thừa kế từ hai hay nhiều lớp cha.

Hãy khảo sát ví dụ sau:



Trong hình trên, chúng ta đã xây dựng một lớp 'Vẽ một hình', lớp này thừa hưởng ba lớp: 'Đường thẳng', 'Đường tròn', 'Hình ảnh'. Như thế lớp 'Vẽ một hình' kết hợp chức năng của ba lớp trên thêm vào chức năng được định nghĩa bên trong nó.

Lớp 'Vẽ một hình' là một ví dụ về tính đa thừa kế.

Có thể sử dụng tính đa thừa kế để xây dựng một lớp mới, lớp này dẫn xuất chức năng của nó từ một vài lớp khác. Như thế, xét theo góc cạnh của người sử dụng lớp mới này, chỉ cần một lớp mà cung cấp tất cả các chức năng. Như vậy, họ không cần phải sử dụng nhiều đối tượng khác nhau.

Sự thuận lợi quan trọng nhất của tính thừa kế là nó thúc đẩy việc tái sử dụng mã chương trình.

Trong ví dụ trên, chúng ta có ba lớp 'Đường thẳng', 'Đường tròn' và 'Hình ảnh'. Giả thiết rằng ba người khác nhau xây dựng ba lớp này riêng biệt. Bây giờ, người sử dụng cần xây dựng một lớp để vẽ đường thẳng, vẽ đường tròn cũng như hiển thị hình ảnh. Vì thế họ tìm kiếm xem có lớp nào đáp ứng một hoặc tất cả các yêu cầu đó. Nếu có những lớp cung cấp chức năng thỏa yêu cầu thì người sử dụng sẽ thừa kế những lớp đó để tạo một lớp mới.

Giờ đây người sử dụng chỉ còn phải viết mã chương trình cho những đặc tính chưa có sau tiến trình thừa kế. Người sử dụng có thể sử dụng chính ba lớp trên. Tuy nhiên, sự thừa kế cung cấp một bó những chức năng hỗn độn trong một lớp.

1.10 Tính Đa hình

Trong một chương trình có cấu trúc (a structured program), một phương thức chỉ ứng dụng cho một đối tượng. Chẳng hạn xét toán tử 'Cộng'. Toán tử này chỉ tính tổng của hai số nguyên. Khi truyền hai giá trị 2 và 3 thì nó hiển thị 5. Chúng ta không thể có một loại toán tử 'Cộng' để tính tổng của hai giá trị văn bản (text) 'Hello!' và 'How are you?' để có

được chuỗi văn bản kết quả 'Hello! How are you?'

Trong hệ thống hướng đối tượng thì tình huống mô tả trên là khả thể.

Định nghĩa

Tính đa hình cho phép một phương thức có các tác động khác nhau trên nhiều loại đối tượng khác nhau.

Với tính đa hình, nếu cùng một phương thức ứng dụng cho các đối tượng thuộc các lớp khác nhau thì nó đưa đến những kết quả khác nhau. Bản chất của sự việc chính là phương thức này bao gồm cùng một số lượng các tham số.

Tính đa hình là một trong những đặc tính quan trọng nhất của hệ thống hướng đối tượng.

Một ví dụ khác là phương thức hiển thị. Tùy thuộc vào đối tượng tác động, phương thức ấy có thể hiển thị một chuỗi, hoặc vẽ một đường thẳng, hoặc hiển thị một hình ảnh. Hãy khảo sát hình sau:

Lớp: Hình thể

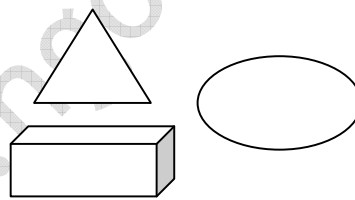
Các lớp con

Các phương thức:

Vẽ

Di chuyển

Khởi tạo



Hình 1.5: Lớp 'Hình thể' và các lớp con

Hình trên cho thấy rằng 'Vẽ' là một phương thức được chia sẻ giữa các lớp con của lớp 'Hình thể'. Tuy nhiên, phương thức 'Vẽ' được ứng dụng cho hình hộp sẽ khác với hình elip.

Tính đa hình hỗ trợ tính đóng gói.

Xét trên mức độ người sử dụng, họ chỉ cần một phương thức 'Vẽ' của lớp 'Hình thể'. Còn cách thức mà phương thức 'Vẽ' được thực thi cho các trường hợp khác nhau thì họ không cần biết.

1.11 Những thuận lợi của Phương pháp hướng Đối tượng

Lập trình hướng đối tượng đòi hỏi một sự chuyển hướng quan trọng trong tư duy của các lập trình viên. Phương pháp này làm cho tốc độ phát triển các chương trình mới nhanh hơn, và nếu được sử dụng cách đúng đắn phương pháp này sẽ cải tiến việc duy trì, việc tái sử dụng và việc đánh giá phần mềm.

Những điểm thuận lợi của phương pháp hướng đối tượng là:

- Phương pháp này tiến hành tiến trình phân tích, thiết kế và phát triển một vấn đề trong khuôn khổ những khái niệm và thuật ngữ thuộc lãnh vực ứng dụng. Vì thế, có một sự tương hợp cao nhất giữa việc phát triển ứng dụng và vấn đề thực tế.
- Chẳng hạn như trong trường hợp bán xe hơi, ở mọi giai đoạn của việc phân tích, thiết kế và phát triển ứng dụng, luôn luôn có tiếng nói của khách hàng, của nhân viên bán hàng ...
- Phương pháp này hỗ trợ việc chia sẻ bên trong một ứng dụng.
- Phương pháp này hỗ trợ việc tái sử dụng các đối tượng khi các ứng dụng mới được phát triển. Đây là sự thuận lợi rất quan trọng xét trong khía cạnh giảm thiểu chi phí có ý nghĩa lâu dài.
- Chẳng hạn như hành vi của khách hàng một khi được mô hình hóa trong một ứng dụng thì có thể được sử dụng lại cho những ứng dụng liên hệ có bao gồm mô hình khách hàng.
- Phương pháp này giảm thiểu các lỗi và những vấn đề liên quan đến việc bảo trì ứng dụng do khả năng tái sử dụng các đối tượng.
- Phương pháp này tăng tốc tiến trình thiết kế và phát triển, một lần nữa đây là kết quả của việc tái sử dụng các đối tượng.

Tóm tắt bài học

- Lập trình hướng Đối tượng là một cách tư duy mới để giải quyết vấn đề với máy vi tính. Thay vì nỗ lực đưa vấn đề vào trong khuôn khổ quen thuộc với máy vi tính, phương pháp hướng đối tượng tìm kiếm một giải pháp toàn vẹn cho một vấn đề.
- Sự trừu tượng hóa dữ liệu là tiến trình xác định và nhóm các thuộc tính và các phương thức liên quan đến một thực thể đặc thù, trong tương quan với một ứng dụng.
- Một lớp định nghĩa một thực thể theo những thuộc tính và những phương thức chung.
- Một đối tượng là một trường hợp của một lớp.
- Một lớp định nghĩa một thực thể, còn đối tượng là thực thể hiện thực.
- Tiến trình hiện thực hóa một đối tượng được gọi là Thiết lập (Construction).
- Tiến trình hủy bỏ một đối tượng được gọi là Hủy (Destruction).
- Tính bền vững là khả năng lưu trữ dữ liệu của một đối tượng vượt quá thời gian tồn tại của đối tượng đó.
- Việc đóng gói là tiến trình che giấu việc thực thi chi tiết của một đối tượng đối với người sử dụng đối tượng ấy.
- Tính thừa kế là cơ chế cho phép một lớp chi sẻ các thuộc tính và các phương thức được định nghĩa trong một hoặc nhiều lớp khác.
- Tính đa hình là một thuộc tính cho phép một phương thức có các tác động khác nhau trên nhiều đối tượng khác nhau.
- Phương pháp hướng đối tượng đưa ra tiến trình phân tích, thiết kế và phát triển ứng dụng trong khuôn khổ các khái niệm và các thuật ngữ thuộc lãnh vực ứng dụng.

Kiểm tra sự tiên bộ

1. Sự trừu tượng hóa dữ liệu đồng nghĩa với sự che giấu dữ liệu. **Đúng/Sai**
2. Định nghĩa sự Trừu tượng hóa dữ liệu.
3. Việc đóng gói dữ liệu che giấu những chi tiết thực thi đối với những đối tượng khác. **Đúng/Sai**
4. Tính đa hình cho phép chúng ta tạo những đối tượng khác với cùng một tên. **Đúng/Sai**
5. Mỗi một đối tượng có một _____ và _____ được xác định rõ.
6. Tất cả các đối tượng của một lớp đều có cùng một tập hợp các thuộc tính. **Đúng/Sai**
7. Một đối tượng định nghĩa một thực thể, trong khi một lớp là thực thể cụ thể. **Đúng/Sai**
8. Định nghĩa tính Đa hình.

Bài tập

1. Thiết kế các thành phần và các hành động khi một khách hàng thực hiện một giao dịch ATM (Automatic Teller Machine).
2. Liệt kê những thuộc tính và những phương thức cần có để vẽ một hình đa giác.