SV : Trần Trung Thăng

MSSV : 2024801030146

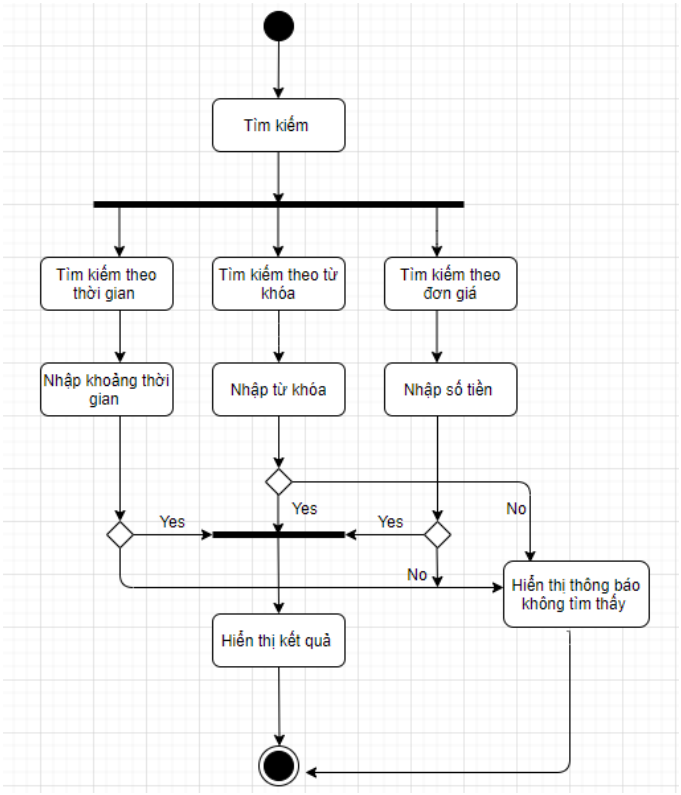
Tìm hiểu 3 biểu đồ UML

1.Sơ đồ hoạt động (Activity Diagram)

a) Giới thiệu sơ đồ hoạt động (Activity Diagram)

Activity diagram (biểu đồ hoạt động) là một mô hình logic được dùng để mô hình hoá cho các hoạt động trong một quy trình nghiệp vụ. Nó chỉ ra luồng đi từ hoạt động này sang hoạt động khác trong một hệ thống. Nó đặc biệt quan trọng trong việc xây dựng mô hình chức năng của hệ thống và nhấn mạnh tới việc chuyển đổi quyền kiểm soát giữa các đối tượng.

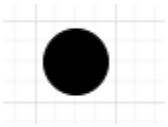
Ví dụ về Activity Diagram về hoạt động tìm kiếm :



b) Các thành phần của Activity Diagram

- Start

- Ký hiệu:



- Ý nghĩa: Khởi tạo hoạt động

- Transition

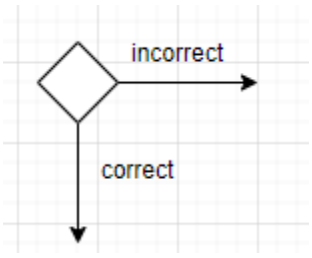
- Ký hiệu

- Ý nghĩa: Mô tả sự chuyển đổi trạng thái của các hoạt động

- Decision

- Ký hiệu:

- Ý nghĩa: **Tập các điều kiện kích hoạt việc chuyển trạng thái**
- Branch:



Mô tả điều kiện rẽ nhánh

- Chỉ một dòng điều khiển đi vào
- Hai hoặc nhiều dòng điều khiển ra

- Chỉ một dòng điều khiển ra dẫn đến kết quả
- Mỗi dòng chứa một điều kiện và loại trừ nhau

Có thể hiểu đây là ký hiệu biểu thị nút điều kiện chuyển hướng. Tùy theo trường hợp đúng hay sai của kết quả biểu thức logic bên trong ký hiệu mà có hướng di chuyển tiếp theo tương ứng.

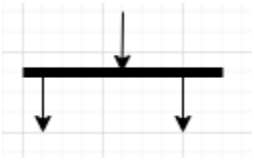
- Synchronization bar

- Ký hiệu

- Ý nghĩa: Mô tả điều kiện thực hiện song song

- Fork

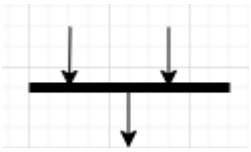
- Ký hiệu



- Ý nghĩa: thể hiện cho trường hợp thực hiện xong một hoạt động rồi sẽ rẽ nhánh thực hiện nhiều hoạt động tiếp theo.

- Join

- Ký hiệu



- Ý nghĩa:

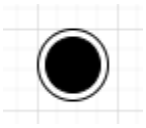
Cùng ký hiệu với Fork nhưng thể hiện trường hợp phải thực hiện hai hay nhiều hành động trước rồi mới thực hiện hành động tiếp theo.

Có thể hiểu đơn giản .Có các trường hợp hội tụ đủ nhiều luồng điều khiển một lúc để gộp thành một luồng xử lý thì cần Join .

Và đôi khi cần phải tách một luồng điều khiển ra thành hai hoặc nhiều luồng khác biệt nhau thì cần Fork. Và mỗi luồng của Fork hoàn toàn không lệ thuộc nhau.

- End

- Ký hiệu



- Ý nghĩa

- Mô tả trạng thái kết thúc quy trình

- Một activity diagram có một hoặc nhiều trạng thái kết thúc

c) Cách dùng sơ đồ hoạt động (Activity Diagram)

- Bước 1: Xác định các nghiệp vụ cần mô tả

Xem xét bản vẽ Use case để xác định nghiệp vụ nào bạn cần mô tả.

- Bước 2: Xác định trạng thái đầu tiên và trạng thái kết thúc

- Bước 3: Xác định các hoạt động tiếp theo

Xuất phát từ điểm bắt đầu, phân tích để xác định các hoạt động tiếp theo chi đến khi gặp thời điểm kết thúc để hoàn tất bản vẽ .

d) Ứng dụng

- Phân tích nghiệp vụ để hiểu rõ hệ thống

- Phân tích Use Case

- Cung cấp thông tin để thiết kế bản vẽ Sequence Diagram

2. Sơ đồ Use Case

a) Giới thiệu sơ đồ Use case

Use Case mô tả sự tương tác giữa người dùng và hệ thống ở trong một môi trường cụ thể, vì một mục đích cụ thể. Môi trường nằm trong một bối cảnh, phạm vi hoặc hệ thống phần mềm cụ thể. Mục đích cụ thể là diễn tả được yêu cầu theo góc nhìn từ phía người dùng.

Sự tương tác giữa người dùng và hệ thống có 2 cách thức phổ biến:

- Cách thức mà người dùng tương tác với hệ thống.
- Cách thức mà hệ thống tương tác với các hệ thống khác.

b) Các thành phần đặc tả Use case

Các thành phần đặc tả Use Case bao gồm: **Actor** (người sử dụng), **Use Case** (chức năng tương tác) **& Relationship** (các quan hệ trong Use Case).

❖ Actor (Người sử dụng)

Actor là thành phần chỉ người dùng hoặc một đối tượng nào đó bên ngoài tương tác với hệ thống. Để xác nhận đó có phải là Actor hay không thì cần xem xét dựa và những câu hỏi sau:

- Ai là người sử dụng chức năng chính của hệ thống (tác nhân chính)?
- Ai sẽ là admin của hệ thống – Người cài đặt, quản lý và bảo trì hệ thống (tác nhân phụ)?
- Ai sẽ cần hệ thống hỗ trợ để thực hiện các tác vụ hằng ngày?
- Hệ thống này có cần phải tương tác với các hệ thống nào khác không?
- Ai là người input dữ liệu vào hệ thống (trường hợp hệ thống lưu trữ dữ liệu)?
- Ai hay cái gì quan tâm đến giá trị mà hệ thống sẽ mang lại?

❖ Use Case (Chức năng tương tác)

Use Case là các chức năng mà các Actor sẽ sử dụng hay thể hiện sự tương tác giữa người dùng và hệ thống. Để tìm ra được các Use Case, ta cần trả lời những câu hỏi sau:

- Actor cần những chức năng nào của hệ thống?
- Actor có hành động chính là gì?

- Actor có cần đọc, thêm mới, hủy bỏ, chỉnh sửa hay lưu trữ loại thông tin nào trong hệ thống không?
- Hệ thống có cần thông báo những thay đổi bất ngờ trong nội bộ cho Actor không?
- Công việc hàng ngày của Actor có thể được đơn giản hóa hoặc hữu hiệu hóa qua các chức năng của hệ thống?
- Use Case có thể được tạo ra bởi sự kiện nào khác không?
- Hệ thống cần những thông tin đầu vào/đầu ra nào? Những thông tin đó sẽ đi từ đâu đến đâu?
- Những khó khăn và thiếu hụt của hệ thống hiện tại nằm ở đâu?

❖ **Relationship (Các quan hệ trong Use Case)**

Các quan hệ trong Use Case gồm 3 loại: **Include, Extend & Generalization**.

Mối quan hệ Include trong Use Case là gì?

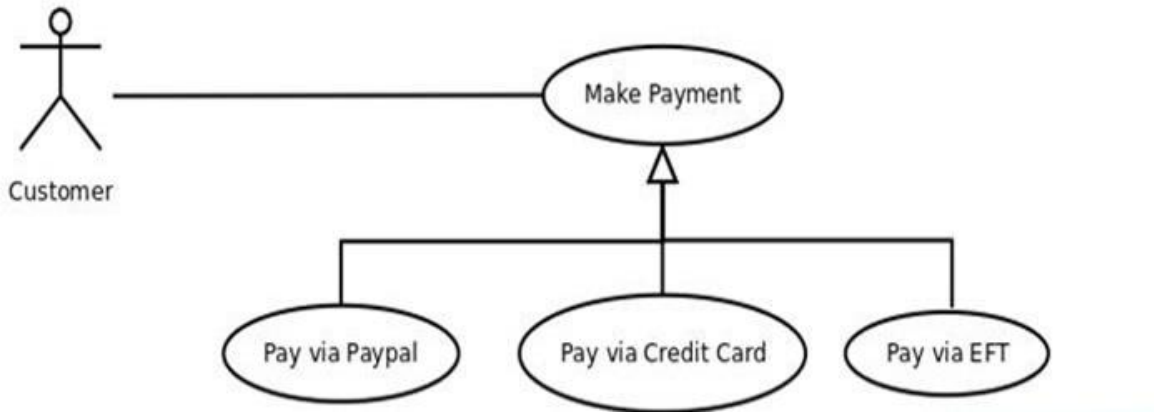
Include là mối quan hệ bao gồm hoặc bắt buộc phải có giữa các Use Case với nhau. Hiểu đơn giản hơn: Để Use Case A xảy ra thì phải đạt được Use Case B.

Ví dụ: Use Case A rút tiền xảy ra thì Use Case B xác thực tài khoản phải hoàn thành.

Mối quan hệ Extend

Extend biểu diễn mối quan hệ mở rộng, không bắt buộc, có thể có hoặc không giữa các Use Case với nhau.

Ví dụ: Use Case B quên mật khẩu có thể xảy ra hoặc không và nó có liên quan đến Use Case A đăng nhập hệ thống chứ không phải bất kỳ một Use Case nào khác.



Mối quan hệ extend thể hiện mối quan hệ không bắt buộc giữa các Use Case với nhau

Mối quan hệ Generalization

Generalization là mối quan hệ cha con giữa các Use Case với nhau. Generalization còn thể hiện khả năng thể hiện mối quan hệ giữa các Actor với nhau.

Ví dụ: Mối quan hệ cha – con giữa các Use Case:

- Đăng nhập (cha): Có thể thông qua số điện thoại (con) hoặc Email (con).
- Đặt hàng (cha): Có đặt hàng qua số điện thoại (con) hoặc website (con).

Ví dụ: Mối quan hệ cha – con giữa các Actor:

- Khách hàng (cha): Gồm khách hàng cũ (con) và khách hàng mới (con).

c) Quy trình vẽ Use case diagram

Xây dựng được một sơ đồ Use Case hoàn chỉnh cần trải qua 3 giai đoạn: **Giai đoạn mô hình hóa, giai đoạn cấu trúc & giai đoạn review**.

Giai đoạn mô hình hóa:

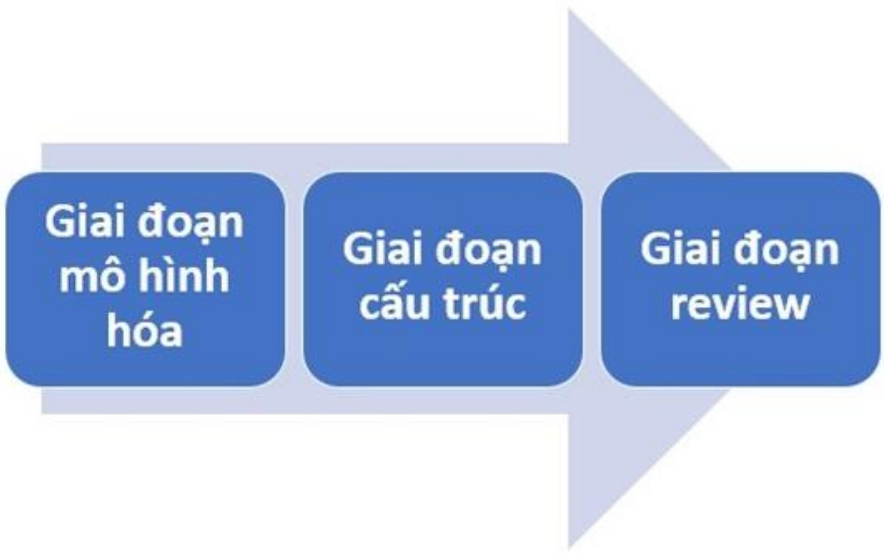
- Bước 1: Thực hiện thiết lập ngữ cảnh của hệ thống.
- Bước 2: Xác định các Actor.
- Bước 3: Xác định các Use Case.
- Bước 4: Định nghĩa các mối quan hệ giữa Actor và Use Case.
- Bước 5: Đánh giá các mối quan hệ đó để tìm cách chi tiết hóa.

Giai đoạn cấu trúc:

- Bước 6: Đánh giá các Use Case cho quan hệ Include.
- Bước 7: Đánh giá các Use Case cho quan hệ Extend.
- Bước 8: Đánh giá các Use Case cho quan hệ Generalization .

Giai đoạn review:

- Bước 9: Kiểm tra (verification): Đảm bảo hệ thống đúng với tài liệu đặc tả.
- Bước 10: Thẩm định (validation): Đảm bảo hệ thống sẽ được phát triển là thứ mà khách hàng cuối thực sự cần thiết.



Vẽ sơ đồ Use Case ***gồm 3 giai đoạn: mô hình hóa, cấu trúc & giai đoạn review***

3) Sơ đồ lớp (Class Diagram)

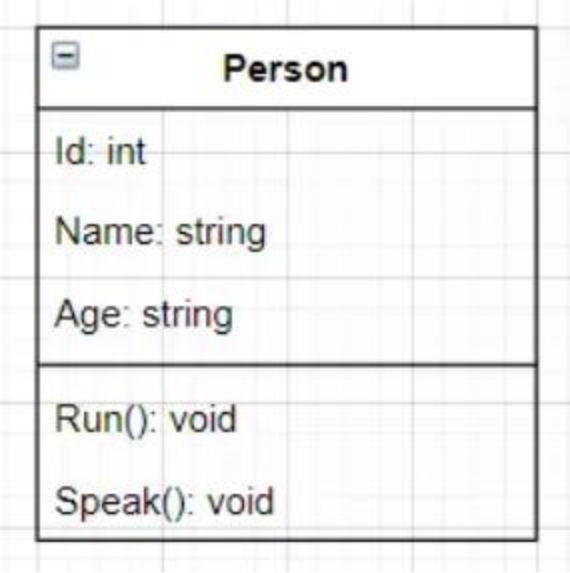
a) Giới thiệu sơ đồ lớp

- Class diagram mô tả kiểu của các đối tượng trong hệ thống và các loại quan hệ khác nhau tồn tại giữa chúng.
- Là một kỹ thuật mô hình hóa tồn tại ở tất cả các phương pháp phát triển hướng đối tượng.
- Biểu đồ hay dùng nhất trong UML và gần gũi nhất với các lập trình viên.
- Giúp các lập trình viên trao đổi với nhau và hiểu rõ ý tưởng của nhau.

b) Các tính chất cơ bản

- Tên class
- Attribute (field, property)
- Operation (method, function)

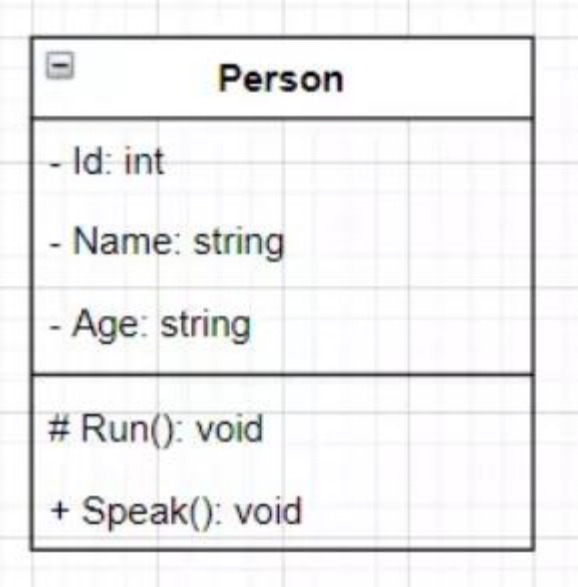
Ví dụ khai báo tên, attribute, operation kèm theo kiểu trả về của 1 class:



❖ Access Modifier trong class diagram

- Sử dụng để đặc tả phạm vi truy cập cho các Attribute và Operation của 1 class (Cấp quyền cho các class khác sử dụng Attribute và Operation của class này).

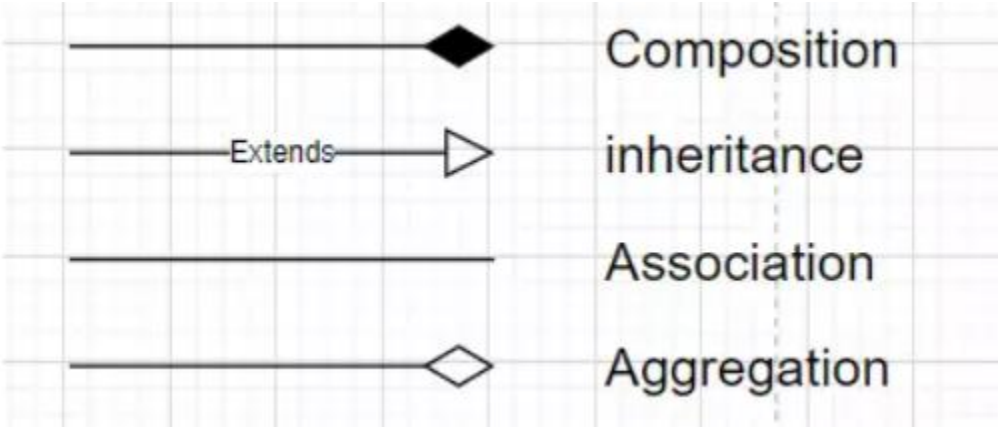
- 4 lựa chọn phạm vi truy cập
    - Private (-): Chỉnh mình các đối tượng được tạo từ class này có thể sử dụng.
    - Public (+): Mọi đối tượng đều có thể sử dụng.
    - Protected (#): Chỉ các đối tượng được tạo từ class này và class kế thừa từ class này có thể sử dụng.
    - Package/Default: Các đối tượng được tạo từ class trong lớp cùng gói có thể sử dụng.



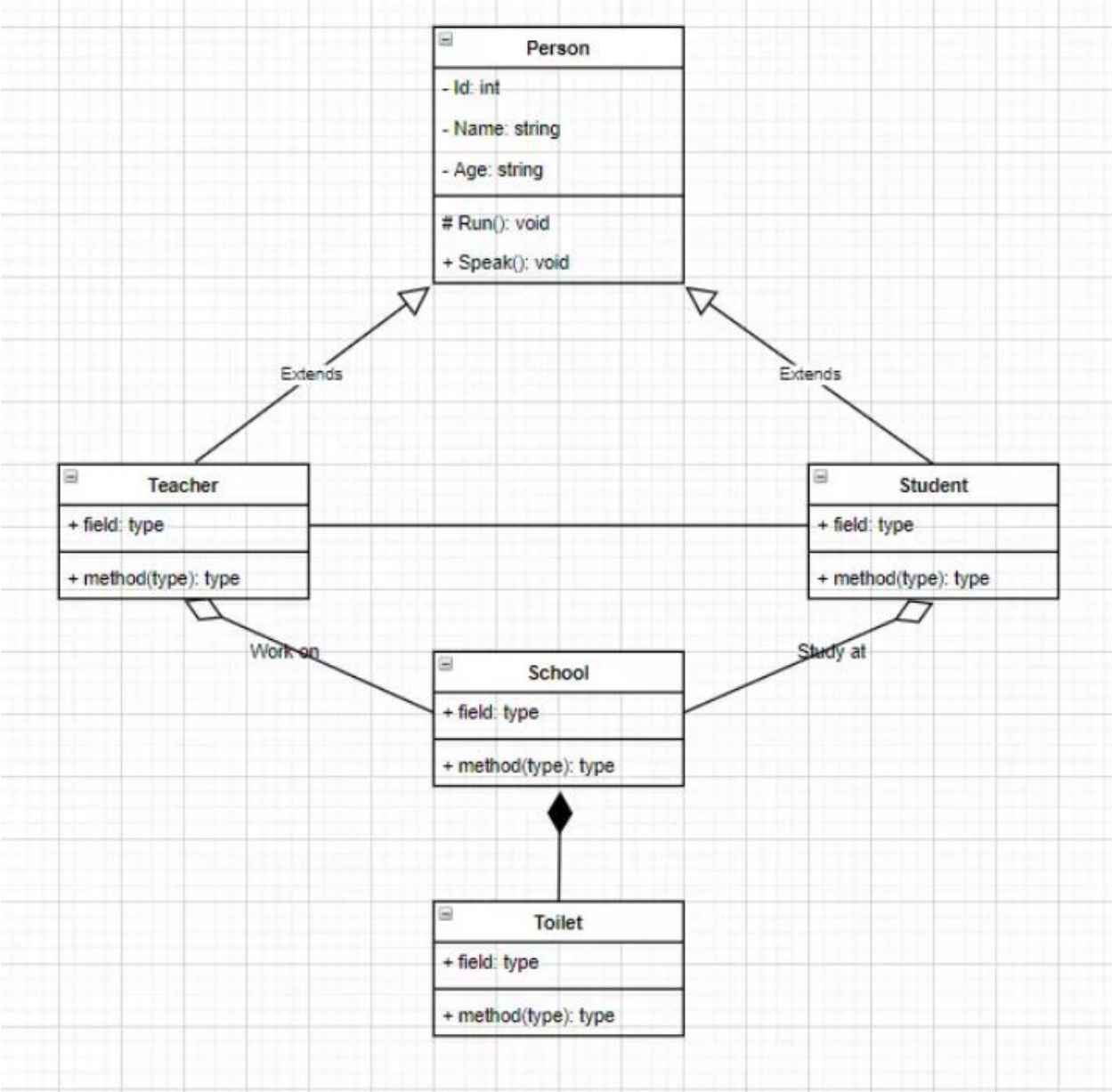
❖ Relationship trong class diagram

- Sử dụng để thể hiện mỗi quan hệ giữa đối tượng được tạo từ 1 class với các đối tượng được tạo từ class khác trong class diagram.

- 4 loại Relationship:



- Inheritance: 1 class kế thừa từ 1 class khác.
- Association: 2 class có liên hệ với nhau nhưng không chỉ rõ mối liên hệ.
- Composition: Đối tượng tạo từ class A mất thì đối tượng tạo từ class B sẽ mất.
- Agreegation: Đối tượng tạo từ class A mất thì đối tượng tạo từ class B vẫn tồn tại độc lập.



❖ Multiplicity trong class diagram

- Sử dụng để thể hiện quan hệ về số lượng giữa các đối tượng được tạo từ các class trong class diagram
  - 0...1: 0 hoặc 1
  - n : Bắt buộc có n
  - 0...* : 0 hoặc nhiều
  - 1...* : 1 hoặc nhiều

- m...n: có tối thiểu là m và tối đa là n