

Ôn tập về hướng đối tượng

cuu duong than cong . com

cuu duong than cong . com



KHOA CÔNG NGHỆ THÔNG TIN
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN



Tham khảo

“Mastering Object-Oriented Analysis and Design with UML 2.0”

IBM Software Group

cuu duong than cong . com

cuu duong than cong . com

Nhắc lại về hướng đối tượng

Một số ký hiệu

Tên class

Tên class
(Các) thuộc tính
(Các) phương thức

cuu duong than cong . com

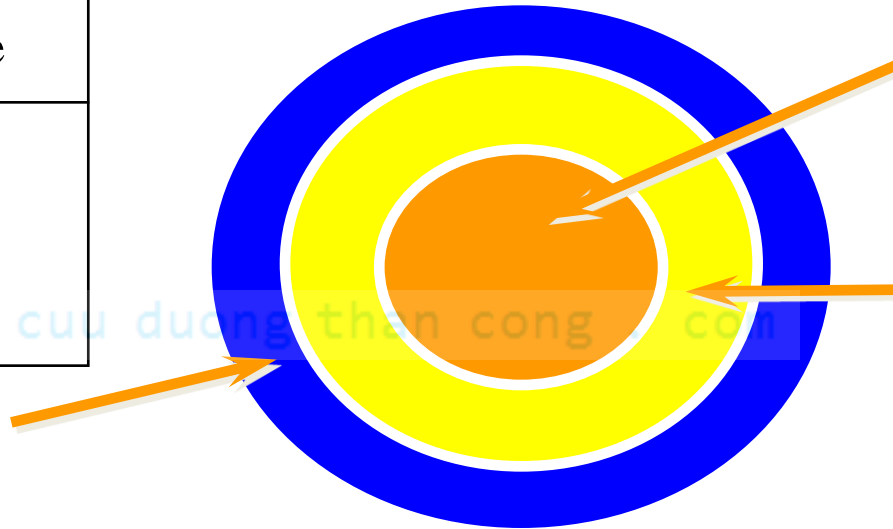
cuu duong than cong . com

Public/Protected/Private

- + Thuộc tính/Phương thức **public**
- # Thuộc tính/Phương thức **protected**
- Thuộc tính/Phương thức **private**

Class
- privateAttribute # protectedAttribute
+publicOp() # protectedOp() - privateOp()

Phương thức
Public



Phương thức
Private

Phương thức
Protected

Tầm vực

- Xác định số lượng thể hiện của thuộc tính / phương thức

Class
- <u>classifierScopeAttribute</u> - instanceScopeAttribute
<u>classifierScopeOperation()</u> instanceScopeOperation()

Ví dụ

Student
<ul style="list-style-type: none"> - name - address - studentID - <u>nextAvailID</u> : int
<ul style="list-style-type: none"> + addSchedule(theSchedule : Schedule, forSemester : Semester) + getSchedule(forSemester : Semester) : Schedule + hasPrerequisites(forCourseOffering : CourseOffering) : boolean # passed(theCourseOffering : CourseOffering) : boolean + <u>getNextAvailID()</u> : int

Nhận xét

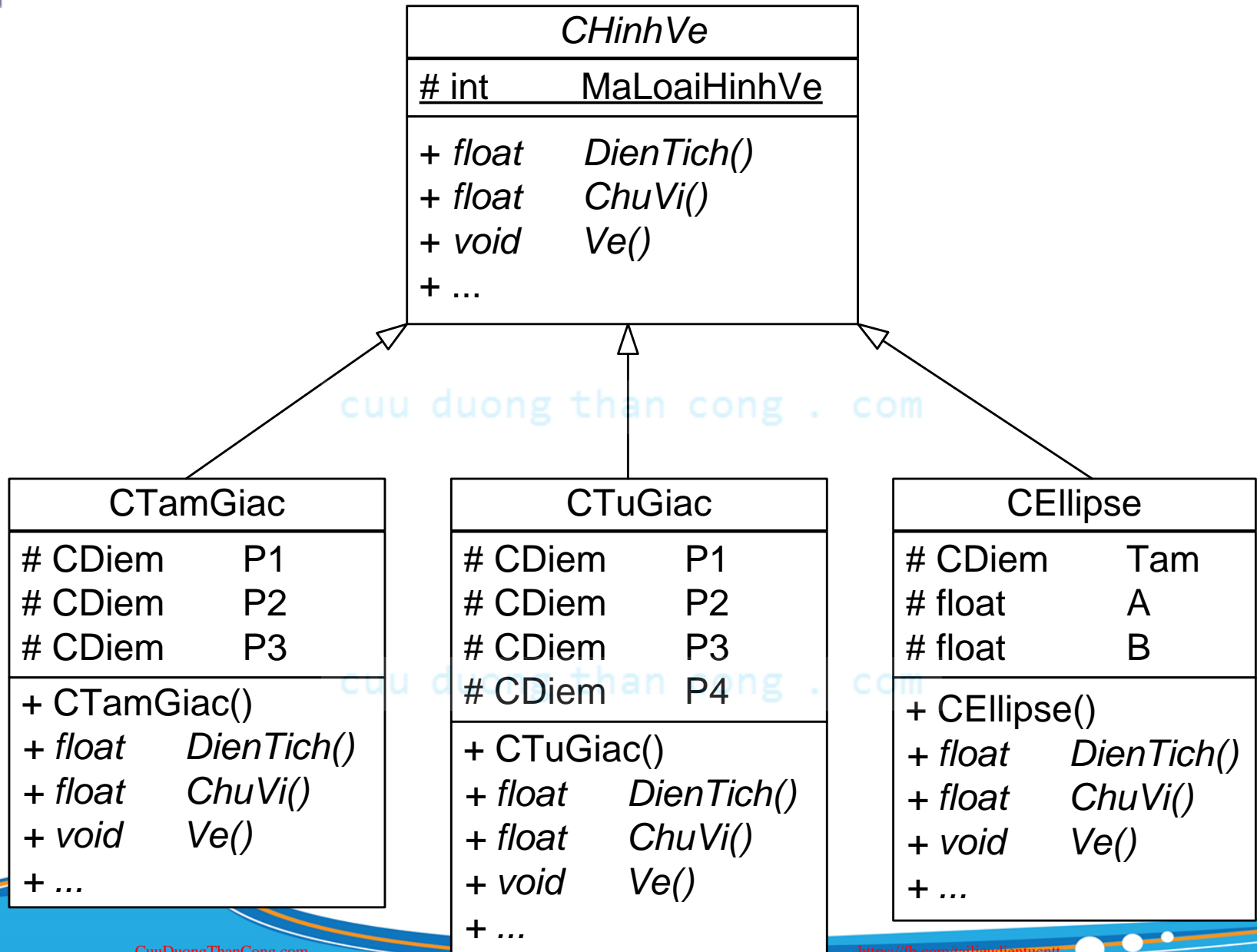
Tên class
(Các) thuộc tính
(Các) phương thức

Bình thường: Class bình thường/Interface
In nghiêng: Class thuần ảo
Gạch dưới: Object (không phải class)

Bình thường: Thuộc tính bình thường
In nghiêng: không sử dụng
Gạch dưới: Thuộc tính static

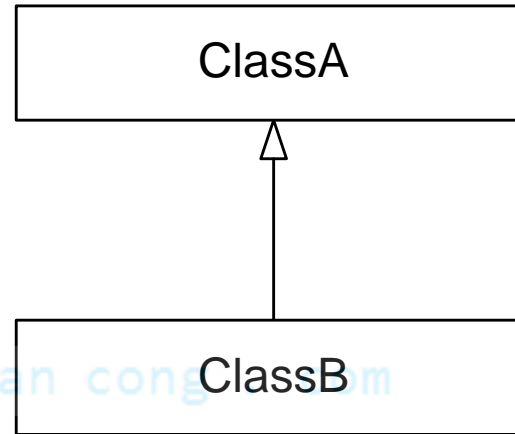
Bình thường: Phương thức bình thường
In nghiêng: Phương thức virtual
Gạch dưới: Phương thức static

Ví dụ



Quan hệ giữa các lớp đối tượng

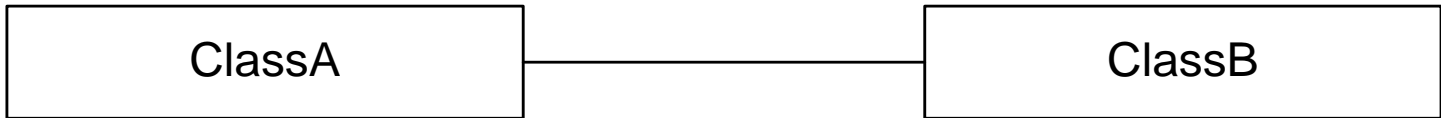
□ Quan hệ kế thừa



- ClassB kế thừa từ ClassA
- ClassB là một trường hợp đặc biệt của ClassA
- ClassA là trường hợp tổng quát của ClassB

Quan hệ giữa các lớp đối tượng

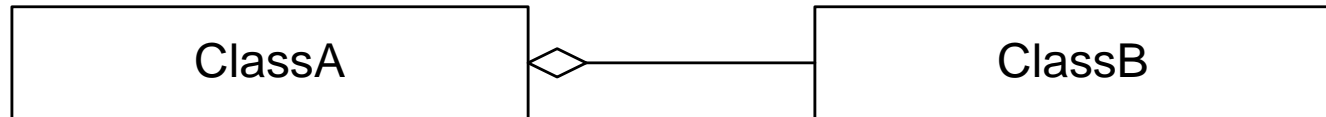
□ Quan hệ Association



- Hoặc
 - Trong **ClassA** có thuộc tính có kiểu là **ClassB**
- Hoặc
 - Trong **ClassB** có thuộc tính có kiểu là **ClassA**
- Nhận xét: Về mặt lập trình, thuộc tính có thể được lưu trữ dạng **biến đơn**, **biến mảng**, hay **biến con trỏ**
- Ví dụ:?

Quan hệ giữa các lớp đối tượng

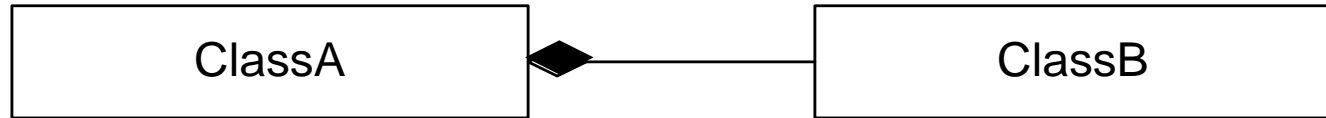
□ Quan hệ Aggregation



- Đã xác định được **ClassA** và **ClassB** có quan hệ Association với nhau
- Xác định rõ hơn:
 - Trong object của **ClassA** có chứa (trong phần thuộc tính) object của **ClassB**
 - **ObjectX** của **ClassA** bị hủy thì **ObjectY** của **ClassB** (bên trong **ObjectX**) vẫn có thể còn tồn tại
- Ví dụ:?

Quan hệ giữa các lớp đối tượng

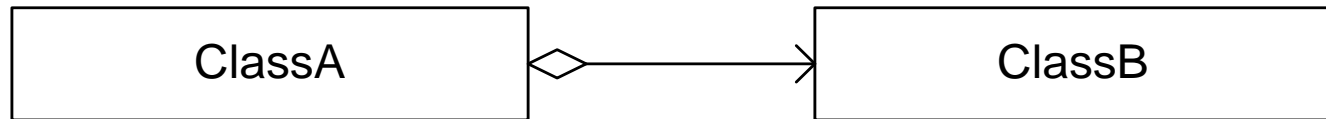
□ Quan hệ Composition



- Đã xác định được **ClassA** và **ClassB** có quan hệ Association với nhau
- Xác định rõ hơn:
 - Trong object của **ClassA** có chứa (trong phần thuộc tính) object của **ClassB**
 - **ObjectX** của **ClassA** bị hủy thì **ObjectY** của **ClassB** (bên trong **ObjectX**) không thể còn tồn tại
- Ví dụ:?

Quan hệ giữa các lớp đối tượng

- Chiều của quan hệ (Association, Aggregation, Composition)

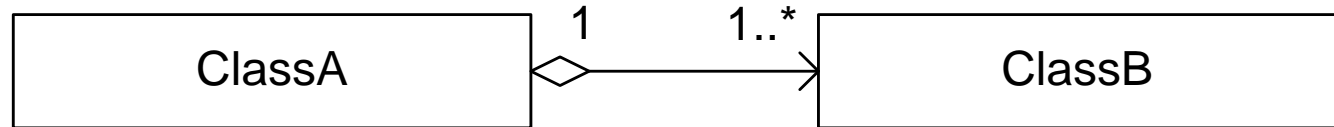


- Nếu quan hệ là 1 chiều: đa số các lời gọi hàm được gọi theo đúng chiều của quan hệ
- Nếu quan hệ là 2 chiều: không vẽ mũi tên

cuu duong than cong . com

Quan hệ giữa các lớp đối tượng

□ Bản số - Multiplicity (Association, Aggregation, Composition)



□ Ý nghĩa

□ Ví dụ:

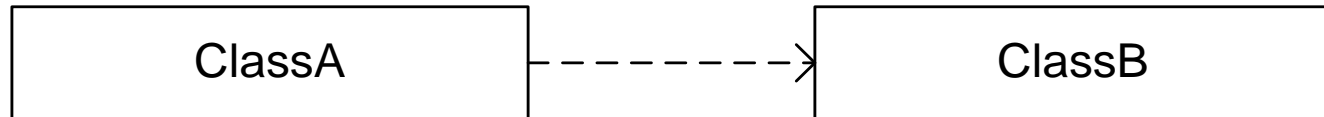
- 1
- 2
- 1..*
- 0..*
- *
- 1, 3, 5..9

cuu duong than cong . com

cuu duong than cong . com

Quan hệ giữa các lớp đối tượng

□ Quan hệ Dependency



- ClassA và ClassB không có quan hệ Association
- ClassA “phụ thuộc” vào ClassB

Tham số truyền vào

```

class A
{
    void F(B x)
    {
        ...
    }
};
  
```

Kết quả trả ra

```

class A
{
    B F()
    {
        ...
    }
};
  
```

Biến cục bộ

```

class A
{
    void F()
    {
        B x;
    }
};
  
```

Trong ClassA có sử dụng biến toàn cục (kiểu B), hoặc sử dụng phương thức/thuộc tính static của ClassB