Chương 10
KẾ THỪA - INHERITANCE

1. TS. Nguyễn Tấn Trần Minh Khang
2. ThS. Võ Duy Nguyên
3. ThS. Nguyễn Hoàng Ngân
4. Hồ Thái Ngọc – Source code.

1. MỤC TIÊU

1. Mục tiêu

- Hiểu được các loại quan hệ?
- Hiểu được kế thừa trong lập trình hướng đối tượng (object–oriented programming) là gì?
- Hiểu được khái niệm cây kế thừa.
- Hiểu được khái niệm sơ đồ lớp.

2. QUAN HỆ

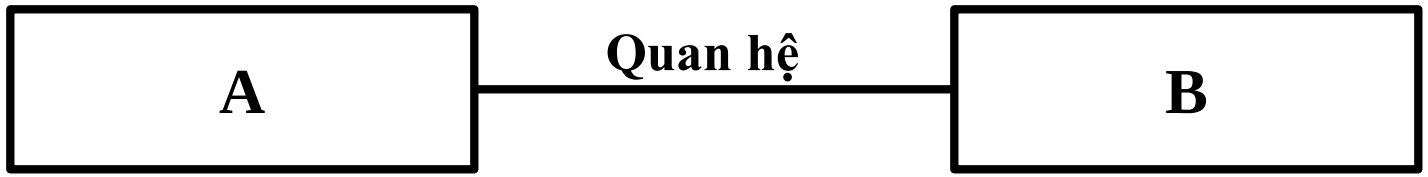
2. Quan hệ

Người ta chia các quan hệ thành những loại như sau:

- Quan hệ một một (1–1).
- Quan hệ một nhiều (1–n).
- Quan hệ nhiều nhiều (m–n).
- Quan hệ đặt biệt hóa, tổng quát hóa.

Quan hệ một một (1-1)

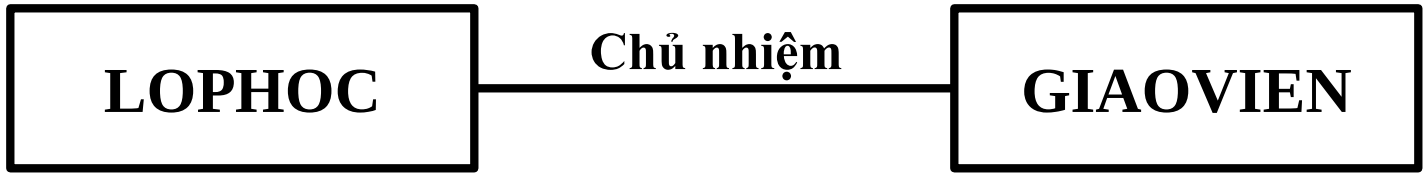
- Khái niệm: Hai lớp đối tượng được gọi là quan hệ một-một với nhau khi một đối tượng thuộc lớp này quan hệ với một đối tượng thuộc lớp kia và một đối tượng thuộc lớp kia quan hệ duy nhất với một đối tượng thuộc lớp này.
- Hình vẽ:



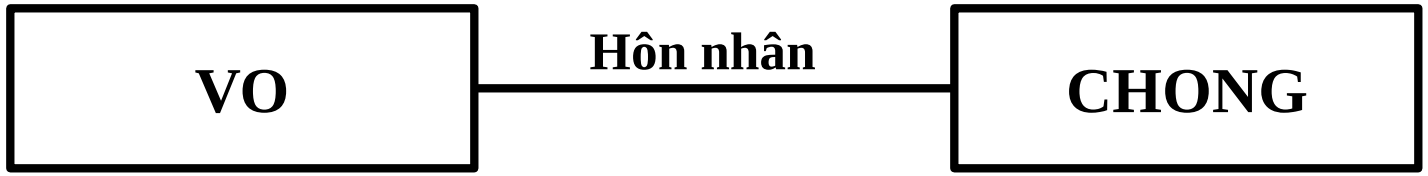
- Trong hình vẽ trên ta nói: một đối tượng thuộc lớp A quan hệ với một đối tượng thuộc lớp B và một đối tượng lớp B quan hệ duy nhất với một đối tượng thuộc lớp A.

Quan hệ một một (1-1)

- Ví dụ 01: Xét ngữ cảnh trường PTTH Bạch Thái Bưởi thuộc quận Hồ Hoàn Kiếm thành phố Hà Nội trong niên khóa 2018 – 2019:

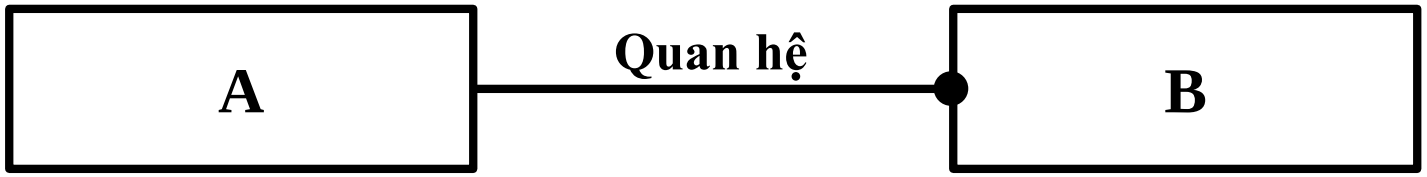


- Ví dụ 02: Xét ngữ cảnh nước Cộng Hòa Xã Hội Chủ Nghĩa Việt Nam, theo hiến pháp và pháp luật tại thời điểm 2019:



Quan hệ một nhiều (1-n)

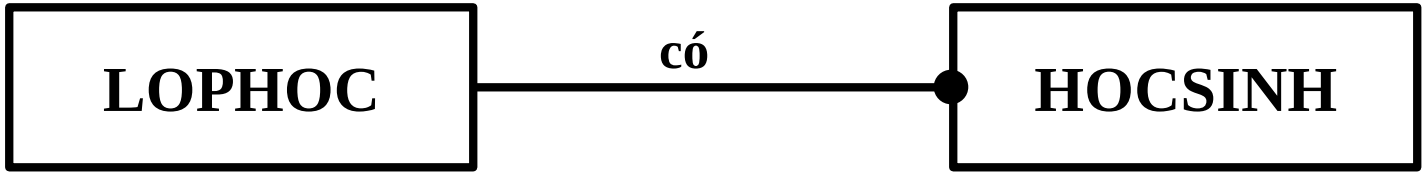
- Khái niệm: Hai lớp đối tượng được gọi là quan hệ một nhiều với nhau khi một đối tượng thuộc lớp này quan hệ với nhiều đối tượng thuộc lớp kia và một đối tượng lớp kia quan hệ duy nhất với một đối tượng thuộc lớp này.
- Hình vẽ:



- Trong hình vẽ trên ta nói: một đối tượng thuộc lớp A quan hệ với nhiều đối tượng thuộc lớp B và một đối tượng lớp B quan hệ duy nhất với một đối tượng thuộc lớp A.

Quan hệ một nhiều (1-n)

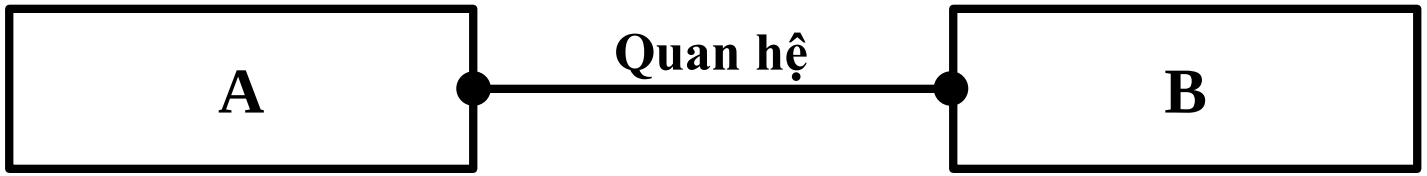
– Ví dụ 01: Xét ngữ cảnh trường PTTH Bạch Thái Bưởi trong niên khóa 2018 – 2019.



– Ví dụ 02: Xét ngữ cảnh quả đất.

Quan hệ nhiều nhiều (m-n)

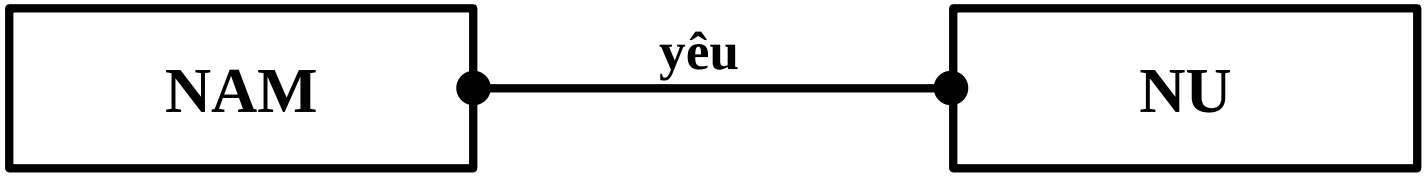
- Khái niệm: hai lớp đối tượng được gọi là quan hệ nhiều -nhiều với nhau khi một đối tượng thuộc lớp này quan hệ với nhiều đối tượng thuộc lớp kia và một đối tượng lớp kia cũng có quan hệ với nhiều đối tượng thuộc lớp này.
- Hình vẽ:



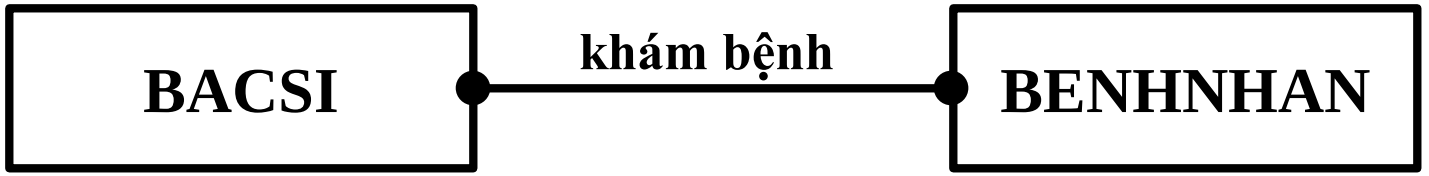
- Trong hình vẽ trên ta nói: một đối tượng thuộc lớp A quan hệ với nhiều đối tượng thuộc lớp B và một đối tượng lớp B cũng có quan hệ với nhiều đối tượng thuộc lớp A.

Quan hệ nhiều nhiều (m-n)

- Ví dụ 01: Xét ngữ cảnh quả đất từ hồi nẩm đến bi chừ.



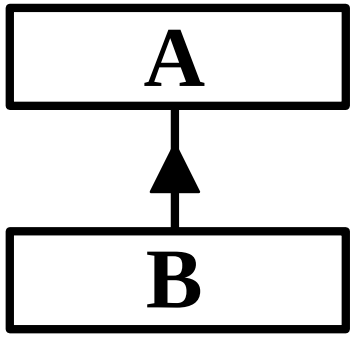
- Ví dụ 02: Xét ngữ cảnh quả đất từ hồi ấy đến hồi này.



Quan hệ đặc biệt hóa – tổng quát hóa

- **Khái niệm:** hai lớp đối tượng được gọi là quan hệ đặc biệt hóa – tổng quát hóa với nhau khi, lớp đối tượng này là trường hợp đặc biệt của lớp đối tượng kia và lớp đối tượng kia là trường hợp tổng quát của lớp đối tượng này.

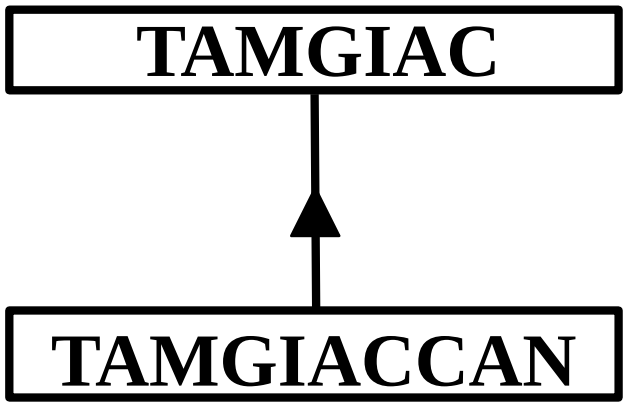
- Hình vẽ



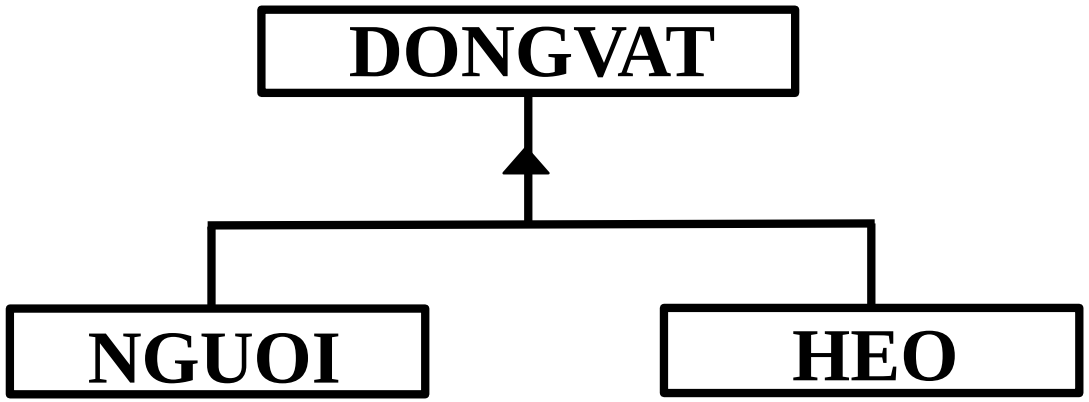
- Trong hình vẽ trên ta nói: lớp đối tượng B là trường hợp đặc biệt của lớp đối tượng A và lớp đối tượng A là trường hợp tổng quát của lớp đối tượng B.

Quan hệ đặc biệt hóa – tổng quát hóa

– Ví dụ 01:



– Ví dụ 02:



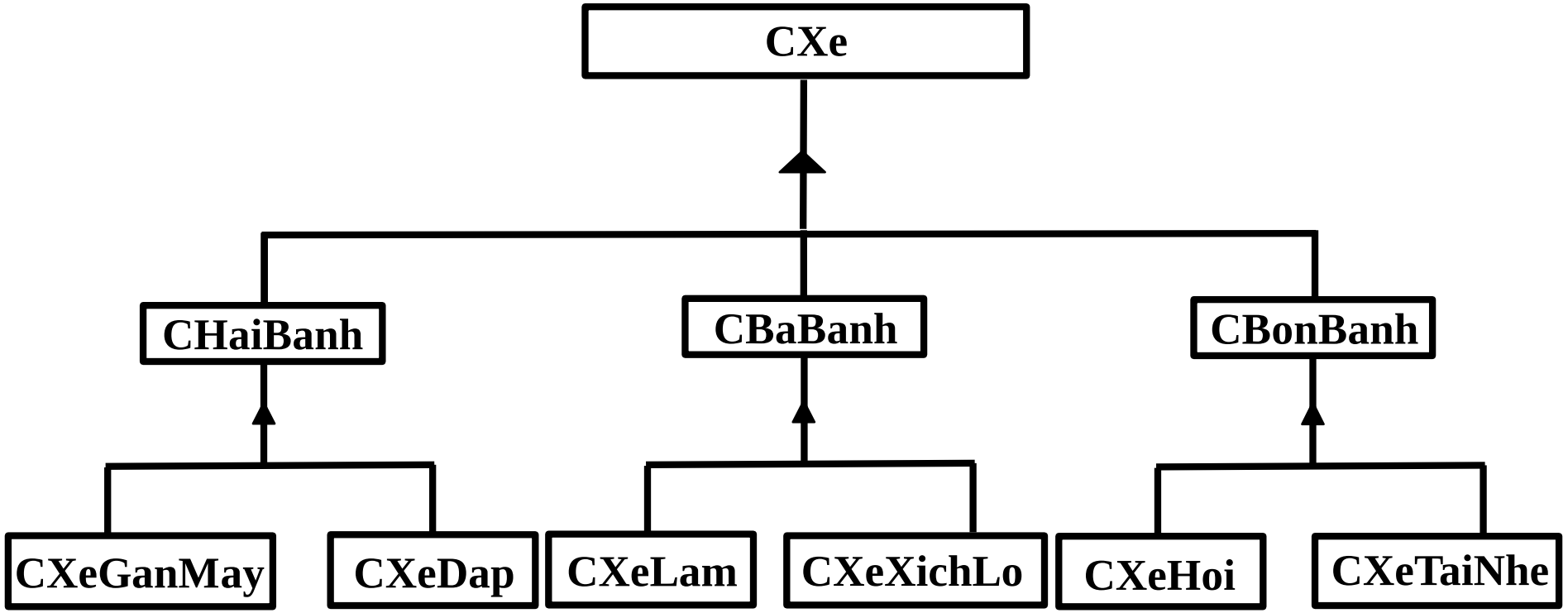
3. CÂY KẾ THỪA

3. Cây kế thừa

- **Khái niệm:** Cây kế thừa là một cây đa nhánh thể hiện mối quan hệ đặc biệt hóa-tổng quát hóa giữa các lớp trong hệ thống, chương trình.
- Ví dụ: Hãy vẽ cây kế thừa cho các lớp đối tượng sau:

+ Lớp CXeDap
+ Lớp CXeGanMay
+ Lớp CXeHoi
+ Lớp CXeHaiBanh
+ Lớp CXeTaiNhe
+ Lớp CXeLam
+ Lớp CXe
+ Lớp CXeBaBanh
+ Lớp CXeBonBanh
+ Lớp CXeXichLo

3. Cây kế thừa

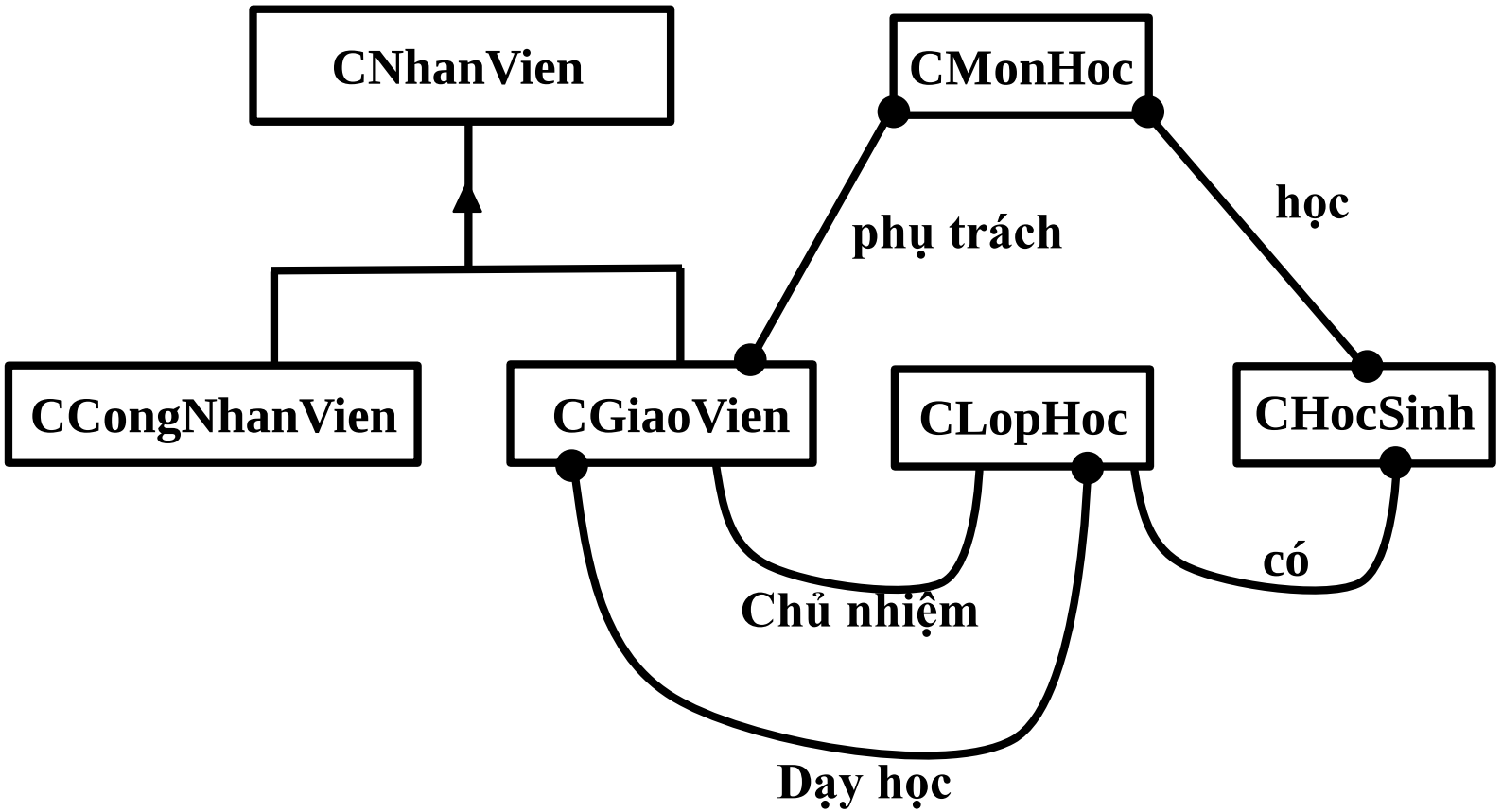

CXe
CHaiBanh
CBaBanh
CBonBanh
CXeGanMay
CXeDap
CXeLam
CXeXichLo
CXeHoi
CXeTaiNhe

4. SƠ ĐỒ LỚP

Sơ đồ lớp

- **Khái niệm:** Sơ đồ lớp là sơ đồ thể hiện tất cả các mối quan hệ giữa các lớp trong hệ thống, chương trình.
- Xét ngữ cảnh trường PTTH Bạch Đằng trong niên học 2019-2020. Hãy vẽ sơ đồ lớp cho các lớp đối tượng sau:
  - + Lớp CGiaoVien
  - + Lớp CHocSinh
  - + Lớp CLopHoc
  - + Lớp CMonHọc
  - + Lớp CNhanVien
  - + Lớp CCongNhanVien
  - + Lớp CNhanVien: tất cả những nhân viên làm việc trong trường.
  - + Lớp CCongNhanVien: là các nhân viên làm việc trong nhà trường nhưng ko trực tiếp đứng lớp. Ví dụ: Bảo vệ, lao công, bảo mẫu.

Sơ đồ lớp


CNhanVien
CMonHoc
phụ trách
học
CCongNhanVien
CGiaoVien
CLopHoc
CHocSinh
có
Chủ nhiệm
Dạy học

Access control

ĐIỀU KHIỂN TRUY XUẤT

Điều khiển truy xuất

- Một thuộc tính hay một phương thức khi được khai báo trong một lớp ta có thể khai báo trong 3 phạm vi khác nhau: private, public hoặc protected.

private

public

protected

- Ví dụ:

```
11.class A
12.{
13.    private:
14.        int a;
15.        void f();
16.    protected:
17.        int b;
18.        void g();
19.    public:
20.        int c;
21.        void h();
22.};
```

Qui tắc truy xuất

PRIVATE

Các thuộc tính và phương thức được khai báo trong phạm vi private của một lớp thì chỉ được phép truy xuất bên trong lớp (accessible only inside the class) và không được quyền truy xuất bên ngoài lớp.

– Ví dụ:

```
11.class A
12.{
13.|   private:
14.|       int a;
15.|       void f();
16.};
17.void main()
18.{
19.|   A x;
20.|   x.a = 15;
21.|   x.f();
22.}
23.void A::f()
24.{
25.|   a = 15;
26.}
```

PRIVATE

Qui tắc truy xuất

PRIVATE

Các thuộc tính và phương thức được khai báo trong phạm vi private của một lớp thì chỉ được phép truy xuất bên trong lớp (accessible only inside the class) và không được quyền truy xuất bên ngoài lớp.

— Ví dụ:

```
11.class A
12.{
13.|   private:
14.|       int a;
15.|       void f();
16.};
17.void main()
18.{
19.|   A x;
20.|   x.a = 15;
21.|   x.f();
22.}
23.void A::f()
24.{
25.|   a = 15;
26.}
```

PRIVATE

ĐÚNG

Qui tắc truy xuất

PRIVATE

Các thuộc tính và phương thức được khai báo trong phạm vi private của một lớp thì chỉ được phép truy xuất bên trong lớp (accessible only inside the class) và không được quyền truy xuất bên ngoài lớp.

— Ví dụ:

```
11.class A
12.{
13.    private:
14.        int a;
15.        void f();
16.};
17.void main()
18.{
19.    A x;
20.    x.a = 15;
21.    x.f();
22.}
23.void A::f()
24.{
25.    a = 15;
26.}
```

SAI

ĐÚNG

PRIVATE

Qui tắc truy xuất

PRIVATE

Các thuộc tính và phương thức được khai báo trong phạm vi private của một lớp thì chỉ được phép truy xuất bên trong lớp (accessible only inside the class) và không được quyền truy xuất bên ngoài lớp.

– Ví dụ:

```
11.class A
12.{
13.|   private:
14.|      int a;
15.|      void f();
16.};
17.void main()
18.{
19.|   A x;
20.|   x.a = 15;  SAI
21.|   x.f();  SAI
22.}
23.void A::f()
24.{
25.|   a = 15;  ĐÚNG
26.}
```

PRIVATE

Qui tắc truy xuất

PROTECTED

Các thuộc tính và phương thức được khai báo trong phạm vi protected của một lớp thì chỉ được phép truy xuất bên trong lớp (accessible only inside the class) và không được quyền truy xuất bên ngoài lớp.

– Ví dụ:

```
11.class A
12.{
13.    protected:
14.        int a;
15.        void f();
16.};
17.void main()
18.{
19.    A x;
20.    x.a = 15; SAI
21.    x.f(); SAI
22.}
23.void A::f()
24.{
25.    a = 15; ĐÚNG
26.}
```

PROTECTED

Qui tắc truy xuất

PUBLIC

Các thuộc tính và phương thức được khai báo trong phạm vi public của một lớp thì không chỉ được phép truy xuất bên trong lớp và cả bên ngoài lớp.

— Ví dụ:

```
11.class A
12.{
13.    public:
14.        int a;
15.        void f();
16.};
17.void main()
18.{
19.    A x;
20.    x.a = 15; ĐÚNG
21.    x.f(); ĐÚNG
22.}
23.void A::f()
24.{
25.    a = 15; ĐÚNG
26.}
```

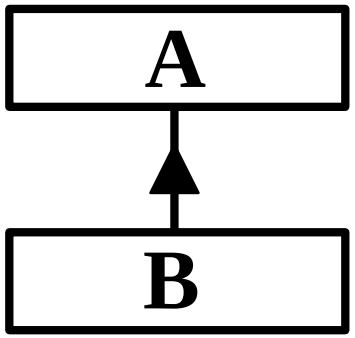
PUBLIC

Inheritance

6. KẾ THỪA - INHERITANCE

Thế giới thực

- Hình vẽ

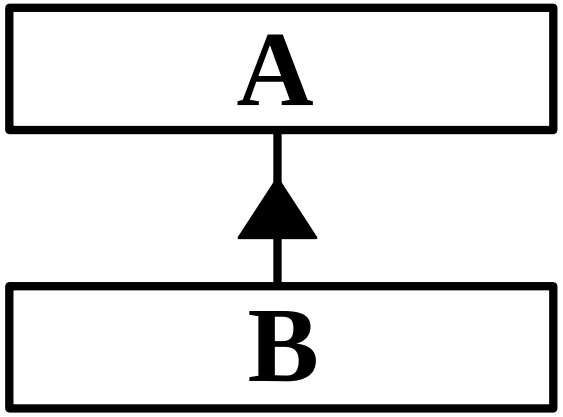


- Trong hình vẽ trên ta nói A và B có quan hệ đặc biệt hoá, tổng quát hoá với nhau.
- Trong đó B là trường hợp đặt biệt của A, và A là trường hợp tổng quát của B.

Lập trình hướng đối tượng

- Xét khai báo.

```
1. class A
2. {
3. |   ...
4. };
5. class B:<từ khóa dẫn xuất> A
6. {
7. |   ...
8. };
```

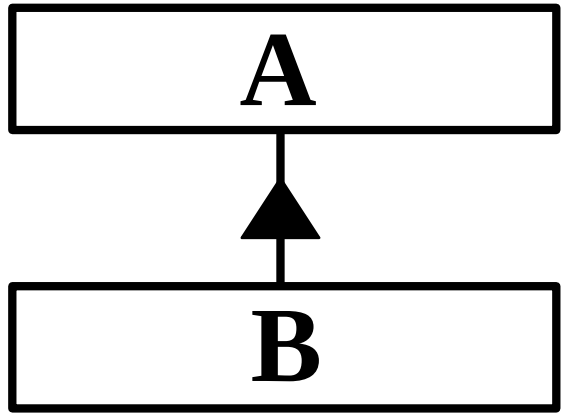


- Trong khai báo trên ta nói lớp B kế thừa từ lớp A.
- Lớp đối tượng A được gọi là lớp cơ sở.
- Lớp đối tượng B được gọi là lớp dẫn xuất từ lớp đối tượng A.

Từ khóa dẫn xuất

– Xét khai báo.

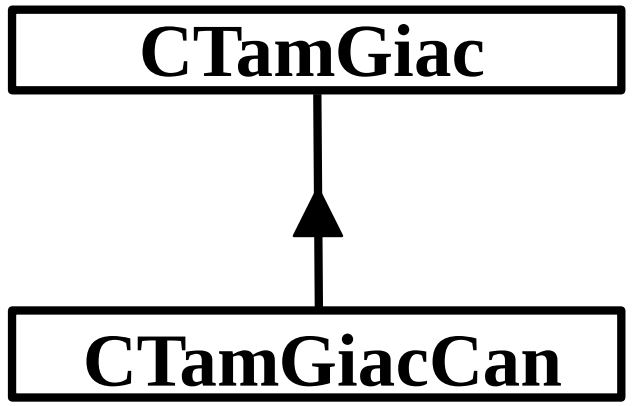
```
1. class A
2. {
3. |   ...
4. };
5. class B:<từ khóa dẫn xuất> A
6. {
7. |   ...
8. };
```



– Trong ngôn ngữ C++ có ba loại *từ khóa dẫn xuất* đó là: private, protected và public. Thông thường trong thực tế người ta hay sử dụng từ khóa dẫn xuất public là nhiều nhất.

Ví dụ kế thừa

– Ví dụ 01: Khai báo lớp tam giác (`CTamGiac`) và lớp tam giác cân (`CTamGiacCan`).

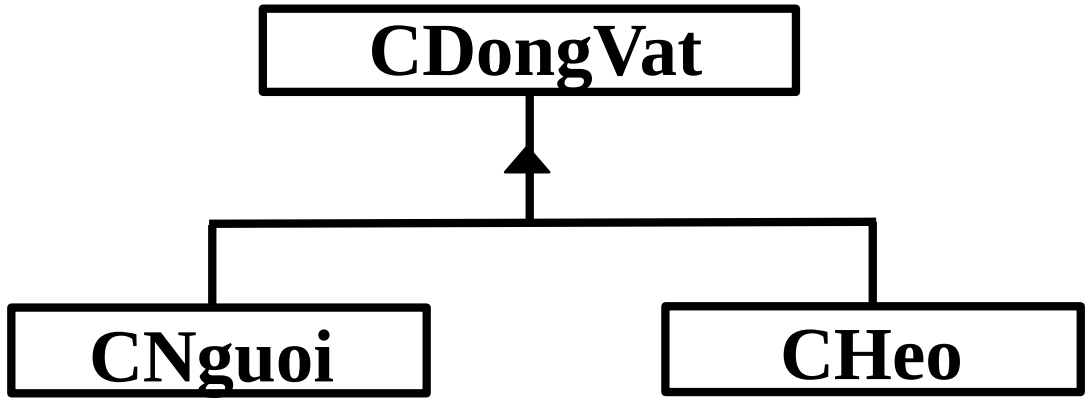



```
1. class CTamGiac
2. {
3. |    ...
4. };
5. class CTamGiacCan:public CTamGiac
6. {
7. |    ...
8. };
```

Ví dụ kế thừa

– Ví dụ 02: Khai báo lớp động vật, lớp heo và lớp người.

```
11.class CDongVat
12.{
13.|   ...
14.};
15.class CHeo:private CDongVat
16.{
17.|   ...
18.};
19.class CNguoi:public CDongVat
20.{
21.|   ...
22.};
```



Access Control and Inheritance

7. BẢNG QUI TẮC KẾ THỪA

Bảng quy tắc kế thừa

Từ khóa dẫn xuất / Phạm vi lớp cơ sở	Private	Public
Private	ll	ll
Protected	**private**	**protected**
Public	**private**	**public**

Bảng quy tắc kế thừa

Từ khóa dẫn xuất / Phạm vi lớp cơ sở	Private	Public
Private	ll	ll
Protected	**private**	**protected**
Public	**private**	**public**

– Các thuộc tính và phương thức được khai báo trong phạm vi *private* của lớp cơ sở thì sẽ không được hiểu ở lớp dẫn xuất.

Bảng quy tắc kế thừa

Từ khóa dẫn xuất / Phạm vi lớp cơ sở	Private	Public
Private	Ⅱ	Ⅱ
Protected	private	protected
Public	private	public

– Các thuộc tính và phương thức được khai báo trong phạm vi protected của lớp cơ sở nếu được dẫn xuất bằng từ khóa private thì các thuộc tính và phương thức đó sẽ được hiểu ở lớp dẫn xuất như là thành phần private của lớp dẫn xuất.

Bảng quy tắc kế thừa

Từ khóa dẫn xuất / Phạm vi lớp cơ sở	Private	Public
Private	ll	ll
Protected	**private**	**protected**
Public	**private**	**public**

– Các thuộc tính và phương thức được khai báo trong phạm vi protected của lớp cơ sở nếu được dẫn xuất bằng từ khóa public thì các thuộc tính và phương thức đó sẽ được hiểu ở lớp dẫn xuất như là thành phần protected của lớp dẫn xuất.

Bảng quy tắc kế thừa

Từ khóa dẫn xuất / Phạm vi lớp cơ sở	Private	Public
Private	Ⅱ	Ⅱ
Protected	**private**	**protected**
Public	**private**	**public**

– Các thuộc tính và phương thức được khai báo trong phạm vi public của lớp cơ sở nếu được dẫn xuất bằng từ khóa private thì các thuộc tính và phương thức đó sẽ được hiểu ở lớp dẫn xuất như là thành phần private của lớp dẫn xuất.

Bảng quy tắc kế thừa

Từ khóa dẫn xuất / Phạm vi lớp cơ sở	Private	Public
Private	ll	ll
Protected	**private**	**protected**
Public	**private**	**public**

– Các thuộc tính và phương thức được khai báo trong phạm vi public của lớp cơ sở nếu được dẫn xuất bằng từ khóa public thì các thuộc tính và phương thức đó sẽ được hiểu ở lớp dẫn xuất như là thành phần public của lớp dẫn xuất.

Bảng quy tắc kế thừa

8. TOÁN TỬ GÁN – KẾ THỪA

Toán tử gán trong kế thừa

Ví dụ dẫn nhập 01: Hãy cho biết trong chương trình dưới đây câu lệnh nào đúng câu lệnh nào sai.

– Chương trình

```
11.class A
12.{
13.|    ...
14.};
15.class B:public A
16.{
17.|    ...
18.};
19.void main()
20.{
21.|    A a;
22.|    B b;
23.|    a = b;
24.|    b = a;
25.}
```

Toán tử gán trong kế thừa

Ví dụ dẫn nhập 02: Hãy cho biết đoạn chương trình dưới đây câu lệnh nào đúng, câu lệnh nào sai.

– Chương trình

```
11.class A
12.{
13.|    ...
14.};
15.class B:public A
16.{
17.|    ...
18.};
19.void main()
20.{
21.|    A *a;
22.|    B *b;
23.|    A x;
24.|    B y;
25.|    a = &x;
26.|    b = &y;
27.|    a = &y;
28.|    b = &x;
29.}
```

Toán tử gán trong kế thừa

Ví dụ dẫn nhập 02: Hãy cho biết đoạn chương trình dưới đây câu lệnh nào đúng, câu lệnh nào sai.

Dòng 21 đọc là: a nhỏ là con trỏ đối tượng thuộc lớp A lớn. Miền giá trị của con trỏ đối tượng a nhỏ là địa chỉ ô nhớ.

– Chương trình

```
11.class A
12.{
13.|    ...
14.};
15.class B:public A
16.{
17.|    ...
18.};
19.void main()
20.{
21.|    A *a;
22.|    B *b;
23.|    A x;
24.|    B y;
25.|    a = &x;
26.|    b = &y;
27.|    a = &y;
28.|    b = &x;
29.}
```

Toán tử gán trong kế thừa

Ví dụ dẫn nhập 02: Hãy cho biết đoạn chương trình dưới đây câu lệnh nào đúng, câu lệnh nào sai.

Dòng 22 đọc là: b nhỏ là con trỏ đối tượng thuộc lớp B lớn. Miền giá trị của con trỏ đối tượng b nhỏ là địa chỉ ô nhớ.

– Chương trình

```
11.class A
12.{
13.|    ...
14.};
15.class B:public A
16.{
17.|    ...
18.};
19.void main()
20.{
21.|    A *a;
22.|    B *b;
23.|    A x;
24.|    B y;
25.|    a = &x;
26.|    b = &y;
27.|    a = &y;
28.|    b = &x;
29.}
```

Toán tử gán trong kế thừa

Ví dụ dẫn nhập 02: Hãy cho biết đoạn chương trình dưới đây câu lệnh nào đúng, câu lệnh nào sai.

Dòng 23 đọc là: x là một đối tượng thuộc lớp A lớn.

— Chương trình

```
11.class A
12.{
13.|    ...
14.};
15.class B:public A
16.{
17.|    ...
18.};
19.void main()
20.{
21.|    A *a;
22.|    B *b;
23.|    A x;
24.|    B y;
25.|    a = &x;
26.|    b = &y;
27.|    a = &y;
28.|    b = &x;
29.}
```

Toán tử gán trong kế thừa

Ví dụ dẫn nhập 02: Hãy cho biết đoạn chương trình dưới đây câu lệnh nào đúng, câu lệnh nào sai.

Dòng 24 đọc là: y là một đối tượng thuộc lớp B lớn.

– Chương trình

```
11.class A
12.{
13.|    ...
14.};
15.class B:public A
16.{
17.|    ...
18.};
19.void main()
20.{
21.|    A *a;
22.|    B *b;
23.|    A x;
24.|    B y;
25.|    a = &x;
26.|    b = &y;
27.|    a = &y;
28.|    b = &x;
29.}
```

Toán tử gán trong kế thừa

Ví dụ dẫn nhập 02: Hãy cho biết đoạn chương trình dưới đây câu lệnh nào đúng, câu lệnh nào sai.

Dòng 25 đọc là (cách 01): con trỏ đối tượng a nhỏ giữ địa chỉ của đối tượng x.

– Chương trình

```
11.class A
12.{
13.|    ...
14.};
15.class B:public A
16.{
17.|    ...
18.};
19.void main()
20.{
21.|    A *a;
22.|    B *b;
23.|    A x;
24.|    B y;
25.|    a = &x;
26.|    b = &y;
27.|    a = &y;
28.|    b = &x;
29.}
```

Toán tử gán trong kế thừa

Ví dụ dẫn nhập 02: Hãy cho biết đoạn chương trình dưới đây câu lệnh nào đúng, câu lệnh nào sai.

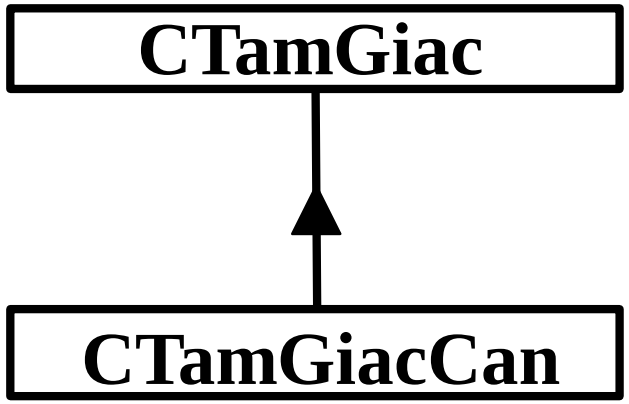
Dòng 25 đọc là (cách 02): địa chỉ của đối tượng x được gán cho con trỏ đối tượng a nhỏ.

– Chương trình

```
11.class A
12.{
13.|    ...
14.};
15.class B:public A
16.{
17.|    ...
18.};
19.void main()
20.{
21.|    A *a;
22.|    B *b;
23.|    A x;
24.|    B y;
25.|    a = &x;
26.|    b = &y;
27.|    a = &y;
28.|    b = &x;
29.}
```

Toán tử gán trong kế thừa

– Toán tử gán trong kế thừa được thực hiện theo nguyên tắc: trường hợp đặc biệt có thể được gán cho trường hợp tổng quát, và trường hợp tổng quát thì không thể gán cho trường hợp đặc biệt được.



Toán tử gán trong kế thừa

- Quy tắc trên áp dụng cho tất cả các ngôn ngữ hỗ trợ lập trình hướng đối tượng như C++, Java, VB.NET, C#, Python,...

- Áp dụng quy tắc trên cho ngôn ngữ lập trình hướng đối tượng C++ ta có thể nói như sau: một đối tượng thuộc lớp dẫn xuất có thể được gán cho một đối tượng thuộc lớp cơ sở. Điều ngược lại là sai, nghĩa là một đối tượng thuộc lớp cơ sở không được quyền gán cho một đối tượng thuộc lớp dẫn xuất.

Toán tử gán trong kế thừa

Ví dụ dẫn nhập 01: Hãy cho biết trong chương trình dưới đây câu lệnh nào đúng câu lệnh nào sai.

– Chương trình

```
11.class A
12.{
13.|    ...
14.};
15.class B:public A
16.{
17.|    ...
18.};
19.void main()
20.{
21.|    A a;
22.|    B b;
23.|    a = b;
24.|    b = a;
25.}
```

Toán tử gán trong kế thừa

- Mở rộng quy tắc trên cho con trỏ đối tượng ta có thể nói như sau: một con trỏ đối tượng thuộc lớp cơ sở có thể giữ địa chỉ của một đối tượng thuộc lớp dẫn xuất. Ngược lại, một con trỏ đối tượng thuộc lớp dẫn xuất không thể giữ địa chỉ của một đối tượng thuộc lớp cơ sở.

- Hiển nhiên, con trỏ đối tượng của một lớp được quyền giữ địa chỉ của đối tượng cùng thuộc về lớp đó.

Toán tử gán trong kế thừa

Ví dụ dẫn nhập 02: Hãy cho biết đoạn chương trình dưới đây câu lệnh nào đúng, câu lệnh nào sai.

– Chương trình

```
11.class A
12.{
13.|    ...
14.};
15.class B:public A
16.{
17.|    ...
18.};
19.void main()
20.{
21.|    A *a;
22.|    B *b;
23.|    A x;
24.|    B y;
25.|    a = &x;
26.|    b = &y;
27.|    a = &y;
28.|    b = &x;
29.}
```