

# Object-Oriented Programming (OOP)

## Lập trình Hướng đối tượng

TH.S ĐOÀN THIÊN MINH

# Nội dung môn học



1. 1.Đặc biệt hóa và sự tổng quát hóa



2. Sự thừa kế



3. Từ khóa Base



4. Từ khóa New



5. Tính đa hình

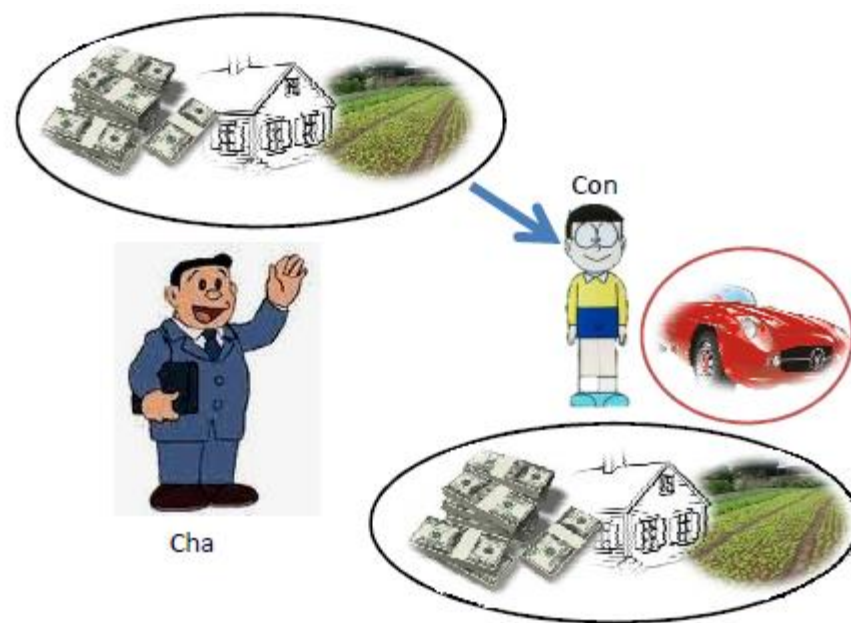


6. Ghi đè (override)

INHERITANCE

# 1. Đặc biệt hóa và sự tổng quát hóa

Trong thế giới thực, ta có thể hiểu kế thừa như sau:



# 1. Đặc biệt hóa và sự tổng quát hóa

Trong lập trình hướng đối tượng, Kế thừa thường được dùng theo 2 cách:

- Để phản ánh mối quan hệ giữa các lớp.
- Để phản ánh sự chia sẻ mã chương trình giữa các lớp.

# 1. Đặc biệt hóa và sự tổng quát hóa

Phân tích thiết kế hướng đối tượng:

Bước đầu tiên khi phân tích một hệ thống gồm các lớp đối tượng, ta xác định xem:

- Có các lớp đối tượng nào ?
- Chúng có các thuộc tính gì ?
- Chúng có quan hệ như thế nào ?
- Có các lớp đối tượng nào ?

# 1. Đặc biệt hóa và sự tổng quát hóa

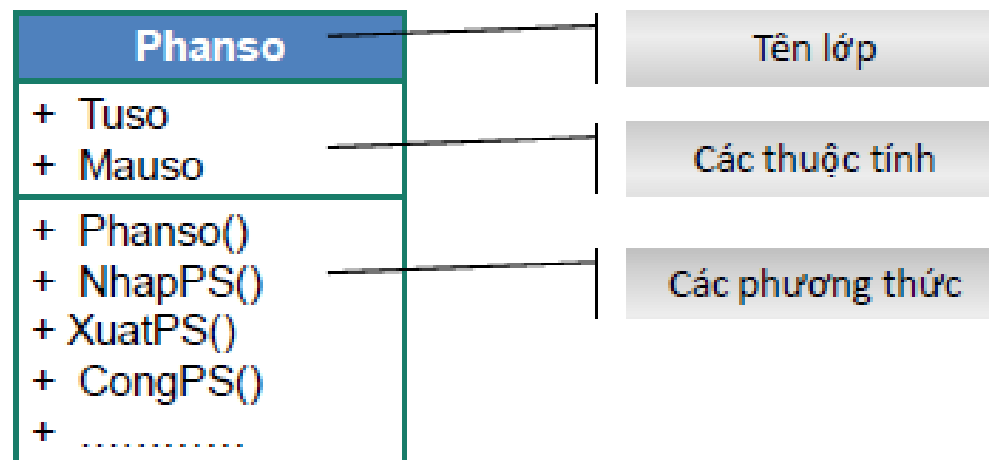
Ví dụ:

- Bài toán tính toán trên phân số, có thể xác định lớp đối tượng sẽ tạo ra là lớp phân số
- Bài toán quản lý nhân sự: cần có lớp đối tượng nhân viên, phòng ban, ....
- Muốn quản lý học sinh phải xây dựng lớp học sinh, ...

# 1. Đặc biệt hóa và sự tổng quát hóa

Ví dụ:

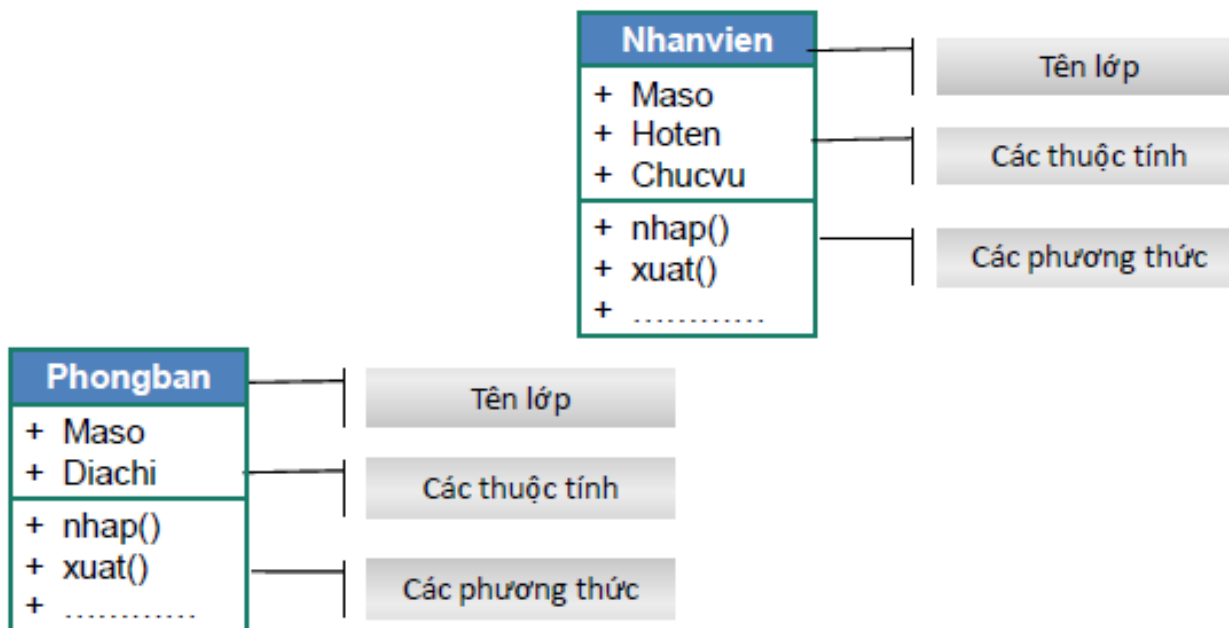
- Lớp phân số:



# 1. Đặc biệt hóa và sự tổng quát hóa

Ví dụ:

- Lớp Nhân viên:





# 1. Đặc biệt hóa và sự tổng quát hóa

Chúng có quan hệ thế nào?

## ❑ Quan hệ chứa - Aggregation

- Một đối tượng có thể là thuộc tính của một đối tượng khác
- Số lượng đối tượng chứa trong đối tượng khác được ràng buộc:
  - 1 có đúng một đối tượng tham gia
  - m..n có từ m đến n
  - 1..\* có một hoặc nhiều
  - 0..1 có một hoặc không

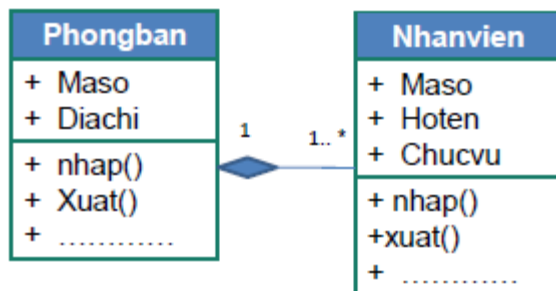
# 1. Đặc biệt hóa và sự tổng quát hóa

Chúng có quan hệ thế nào?

❑ Quan hệ chứa – Aggregation

Ví dụ: quan hệ giữa nhân viên với phòng ban

- Một phòng ban gồm nhiều nhân viên
- Một phòng ban có số lượng nhân viên được giới hạn



# 1. Đặc biệt hóa và sự tổng quát hóa

Chúng có quan hệ thế nào?

## □ Quan hệ là một – inheritance

- Trong hướng đối tượng, Sự kế thừa thường được biểu diễn bởi quan hệ “**là một**”.



Chó “**là một**” loại động vật



Gà “**là một**” loại động vật

## 2. Sự kế thừa

### 2.1 Khái niệm kế thừa:

- là cơ chế cho phép định nghĩa một lớp mới, kế thừa thuộc tính và phương thức của lớp khác. Sau đó xây dựng thêm các thuộc tính và các phương thức riêng của lớp đó.
- Tính kế thừa trong C# cho phép lập trình viên tạo ra một lớp mới kế thừa một lớp đã tồn tại.
  - lớp được kế thừa sẽ có tất cả những thuộc tính và phương thức với quyền truy cập không phải private của lớp cha.
- C# chỉ hỗ trợ đơn thừa kế
  - một lớp chỉ được phép kế thừa một lớp khác.

## 2. Sự kế thừa

Ví dụ 1:



Cả 2 lớp chó và gà đều có những biến và hàm giống hệt nhau về nội dung  
→ tạo ra một lớp Dongvat (animal) chứa các thông tin chung để sử dụng lại

## 2. Sự kế thừa

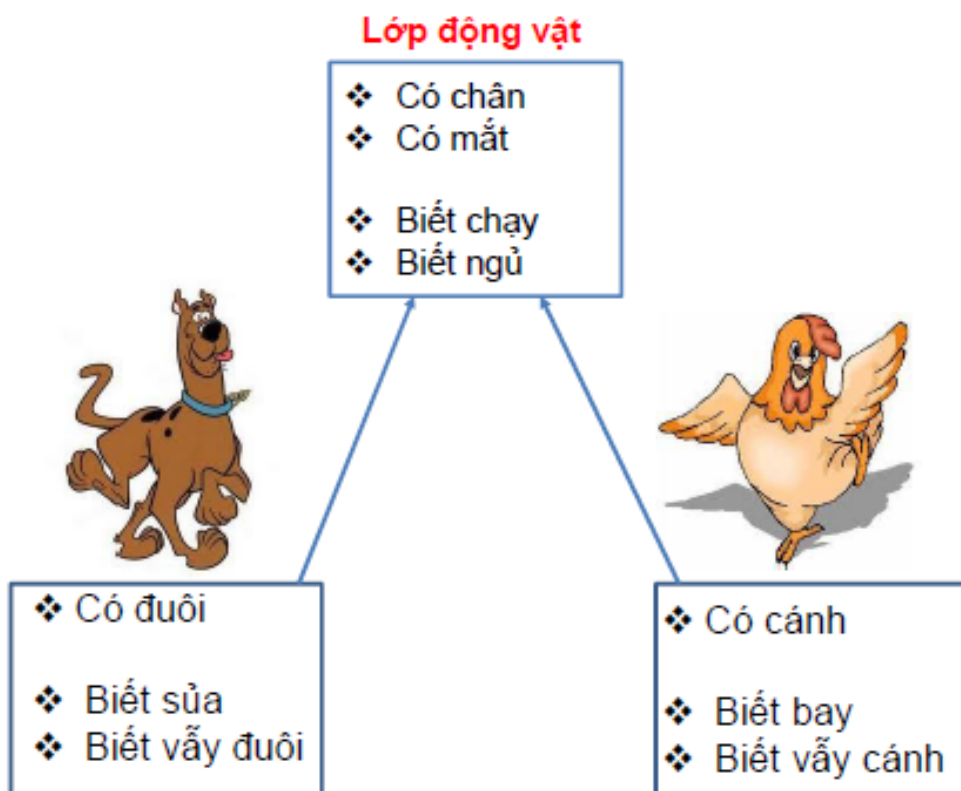
Ví dụ 1:

	Chó và gà là trường hợp đặc biệt của <b>lớp động vật</b>	Có thêm đặc điểm riêng là có đuôi, biết sủa, biết vẫy đuôi.
		Có thêm đặc điểm riêng là có cánh, biết bay, biết vẫy cánh.

Lớp động vật là **sự tổng quát hóa** của lớp gà và lớp chó

## 2. Sự kế thừa

Ví dụ 1:

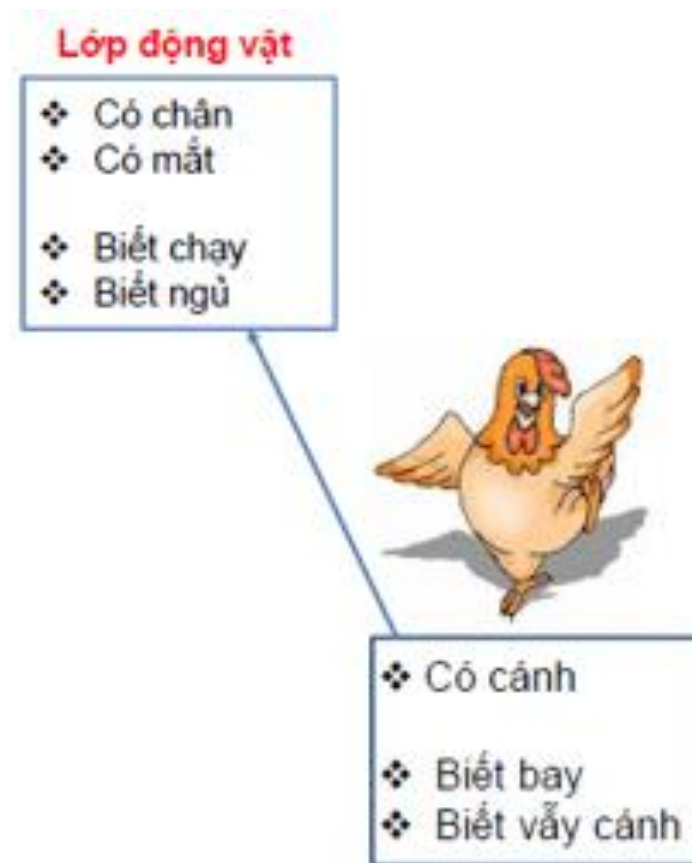


Lớp động vật (animal) chứa các thông tin của lớp chó(dog) và lớp gà(chicken)

## 2. Sự kế thừa

Ví dụ:

Lớp gà kế thừa lớp động vật (**có chân, có mắt, biết chạy, biết ngủ**) và xây dựng thêm các thuộc tính và phương thức riêng (**có cánh, biết bay, biết vỗ cánh**)





## 2. Sự kế thừa

### 2.2 Ý nghĩa sự kế thừa:

- ▶ Tạo ra các lớp từ các lớp cơ sở đã có
- ▶ Tái sử dụng mã chương trình
- ▶ Lớp thừa kế gọi là lớp dẫn xuất

## 2. Sự kế thừa

### 2.3 Cú pháp:

Trong ngôn ngữ C# để tạo một lớp dẫn xuất (kế thừa) từ một lớp ta **thêm dấu hai chấm** vào ***sau** tên lớp dẫn xuất* và ***trước** tên lớp cơ sở*


```
<access-specifier> class <base_class>
{
    ...
}
class <derived_class> : <base_class>
{
    ...
}
```

```
class TênLớpCha
{
    //...
}

class TênLớpCon : TênLớpCha
{
    //...
}
```

## 2. Sự kế thừa:

Ví dụ: Lớp cơ sở và lớp dẫn xuất

Lớp Động vật	Lớp cha trong sự kế thừa được gọi là <b>lớp cơ sở</b> (base class)
	Lớp con trong sự kế thừa được gọi là <b>lớp dẫn xuất</b> (derived class)

## 2. Sự kế thừa

Ví dụ:

- Đoạn lệnh trên khai báo một lớp mới là LopGa. Lớp này dẫn xuất từ LopDongVat. Dấu “:” có thể đọc là “dẫn xuất từ”.

```
class LopDongVat
```

```
{
```

```
//thuộc tính  
//hành vi
```

Lớp dẫn xuất

Lớp cơ sở

```
class LopGa : LopDongVat
```

```
{
```

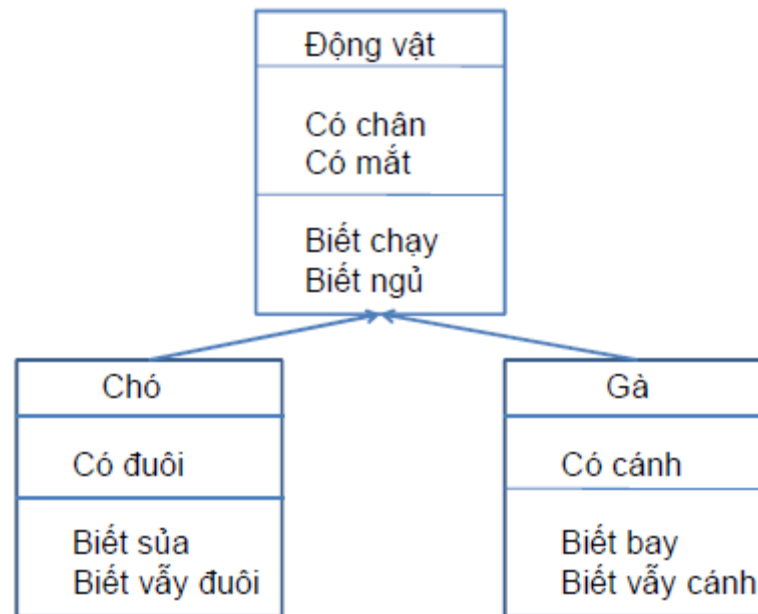
```
//thuộc tính  
//hành vi
```

```
}
```

## 2. Sự kế thừa

Ví dụ1:

Viết chương trình biểu diễn khái niệm lớp chó và lớp gà



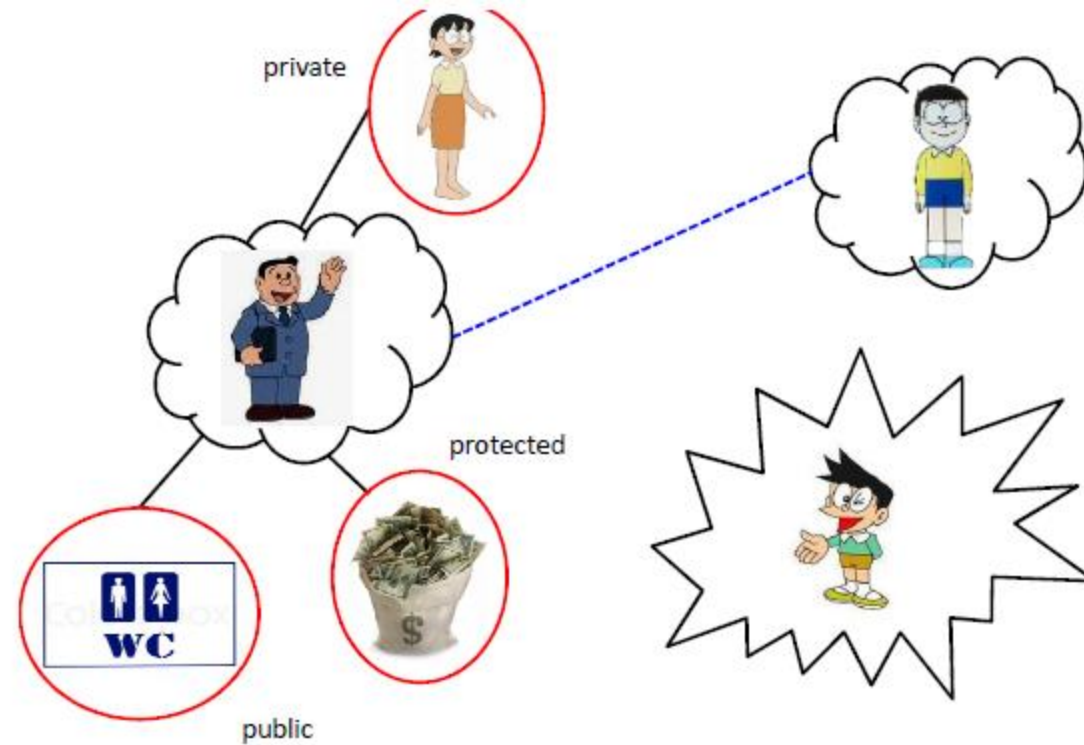
## 2. Sự kế thừa

### 2.3 Quyền thừa kế

Khi nói đến thừa kế, vấn đề đặt ra là một lớp dẫn xuất **có thể thừa kế được gì và không thừa kế được gì** từ lớp cơ sở

→ **phụ thuộc vào các thuộc tính truy cập của dữ liệu thành viên trong lớp cơ sở**

## 2.3 Quyền thừa kế



## 2.3 Quyền thừa kế

	Public	Private	Protected	Constructor	Destructor
Inheritance	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>



## 2.4 Toán tử is

### Toán tử is

Sử dụng toán tử 'is' bạn có thể kiểm tra một đối tượng có phải là kiểu của một class hay không

```
bool isAnimal=ga is Animal;  
Console.WriteLine("\n ga la con cua lop animal? {0}", isAnimal);
```

Kết quả

```
ga la con cua lop animal? True
```

#### Lớp động vật

- ❖ Có chân
- ❖ Có mắt
- ❖ Biết chạy
- ❖ Biết ngủ



- ❖ Có cánh
- ❖ Biết bay
- ❖ Biết vẫy cánh

## 2.5 chuyển kiểu dữ liệu (Type Conversion)

### 2.5 Chuyển kiểu (Type Conversion) trong c sharp:

Chuyển kiểu dữ liệu là chuyển đổi từ một kiểu dữ liệu này sang một kiểu dữ liệu khác. Nó còn được gọi là ép kiểu. Có 2 cách chuyển giá trị từ kiểu này sang kiểu khác đó là:

- Chuyển kiểu ngầm định (implicit type-cast)
- Chuyển kiểu chỉ định (explicit type-cast)

## 2.5 chuyển kiểu dữ liệu (Type Conversion)

### 2.5 Chuyển kiểu (Type Conversion) trong c sharp:

➤ Chuyển kiểu ngầm định (implicit type-cast):

Là chuyển đổi từ kiểu có phạm vi nhỏ đến lớn hơn và chuyển đổi từ các lớp kế thừa thành các lớp cơ sở.

Ví dụ1: Ép từ kiểu nhỏ qua kiểu lớn

```
int i = 59;
```

```
double x = i;
```

Ví dụ 2:

```
string s = "Hello";
```

```
object o = s;
```

## 2.5 chuyển kiểu dữ liệu (Type Conversion)

### 2.5 Chuyển kiểu (Type Conversion) trong c sharp:

- Chuyển kiểu ngầm định (implicit type-cast):

From	To
sbyte	short, int, long, float, double, decimal
byte	short, ushort, int, uint, long, ulong, float, double, decimal
short	int, long, float, double, decimal
ushort	int, uint, long, ulong, float, double, decimal
int	long, float, double, decimal
uint	long, ulong, float, double, decimal
long, ulong	float, double, decimal
float	double

## 2.5 chuyển kiểu dữ liệu (Type Conversion)

### 2.5 Chuyển kiểu (Type Conversion) trong c sharp:

- Chuyển kiểu chỉ định (explicit type-cast) :

Loại này được thực hiện tường minh bởi người dùng bằng cách sử dụng những hàm định nghĩa trước. Nó yêu cầu một toán tử ép kiểu.

Ví dụ 1: Ép từ kiểu lớn qua kiểu nhỏ: có thể mất giá trị

```
double x = 74.86;
```

```
int i = (int)x; // i = 74
```

Ví dụ 2: Ép từ lớp cơ sở qua lớp dẫn xuất

```
string s = "Hello";
```

```
object o = s;
```

```
string s2 = (string)o;
```

## 2.5 Từ khóa **new**

2.5 Từ khóa **new** định nghĩa **mới** phương thức trong lớp dẫn xuất

- ▶ Trong C#, muốn định nghĩa lại một phương thức của lớp cơ sở trong lớp dẫn xuất ta sử dụng từ khóa **new**.

Ví dụ:

```
public new void Tinhluong()  
{  
    //some thing here  
}
```

## 2. Sự kế thừa

### 2.6 Từ khóa **base**

Gọi phương thức được định nghĩa trong lớp cơ sở ta sử dụng từ khóa **base**.

Ví dụ:

```
base.tinhluong();
```

## 2. Sự kế thừa

### 2.6 Từ khóa **override**

Override method (ghi đè) là cách giúp lớp con thay đổi phương thức được kế thừa từ lớp cha. Tức là tạo một phương thức có số lượng tham số và kiểu trả về giống y hệt lớp cha.

Để có thể ghi đè bạn phải đặt từ khóa `virtual` trong phương thức của lớp cha bạn cho phép ghi đè. Nếu phương thức trong lớp cha không có từ khóa `virtual` bạn phải sử dụng từ khóa `new` trong lớp con để override phương thức đó.



# Bài tập

Một công ty lập trình game cần xây dựng chương trình có nhiều nhân vật là Chiến binh, Phù thủy, Nông dân. Mỗi nhân vật cần quản lý các dữ liệu là các thuộc tính:

- ▶ Tên nhân vật (kiểu chuỗi ký tự)
- ▶ Năng lực chiến đấu (damage) kiểu số nguyên
- ▶ Lượng máu tồn tại (hitpoints) kiểu số nguyên,
- ▶ Và phương thức thực thi nhiệm vụ nhân vật

Ngoài ra:

- ▶ Nhân vật chiến binh có thêm thuộc tính vũ khí (kiểu chuỗi ký tự)
- ▶ Nhân vật phù thủy có thêm thuộc tính phép thuật (kiểu chuỗi ký tự)
- ▶ Nhân vật Nông dân có thêm thuộc tính dụng cụ (kiểu chuỗi ký tự)

# Bài tập

Hãy cài đặt các yêu cầu sau:

- ❑ Yêu cầu 1: Xây dựng các lớp Chienbinh, Phuthuy, Nong dan kế thừa từ lớp Nhanvat.
- ❑ Yêu cầu 2: xây dựng các hàm khởi tạo cho các nhân vật.
- ❑ Yêu cầu 3: xây dựng phương thức thực thi nhiệm vụ thể hiện tính ghi đè (override) với phương thức thực thi nhiệm vụ lớp Nhanvat. Sau cho:
  - ▶ phương thức thực thi nhiệm vụ lớp chiến binh xuất thông báo ra màn hình như sau “ Tôi là chiến binh và tôi đang chiến đấu”
  - ▶ phương thức thực thi nhiệm vụ lớp phù thủy xuất thông báo ra màn hình như sau “ Tôi là phù thủy và tôi làm phép”
  - ▶ phương thức thực thi nhiệm vụ lớp nông dân xuất thông báo ra màn hình như sau “ Tôi là nông dân và tôi đang gặt lúa ”

Xuất thông tin

```
Chien binh co vu khi la Ten lua, 100 don vi nang luong va 100 don vi mau ton tai
Nong dan co co cong cu la Luoi liem, 20 don vi nang luong va 20 don vi mau ton tai
Phu thuy co co phép thuật la Cay Choi, 50 don vi nang luong va 50 don vi mau ton tai

Toi la chien binh va Toi dang chien dau
Toi la nong dan va Toi dang gat lua

Toi la phu thuy va Toi lam phep
```

# Bài tập

Một đơn vị sản xuất gồm có các cán bộ là công nhân, kỹ sư, nhân viên. Mỗi cán bộ cần quản lý các dữ liệu: Họ tên, tuổi, giới tính(name, nữ, khác), địa chỉ.

Cấp công nhân sẽ có thêm các thuộc tính riêng: Bậc (1 đến 10).

Cấp kỹ sư có thuộc tính riêng: Ngành đào tạo.

Các nhân viên có thuộc tính riêng: công việc.

- ▶ Yêu cầu 1: Xây dựng các lớp CongNhan, KySu, NhanVien kế thừa từ lớp CanBo.
- ▶ Yêu cầu 2: Xây dựng lớp QLCB(quản lý cán bộ) cài đặt các phương thức thực hiện các chức năng sau:
  - Thêm mới cán bộ.
  - Tìm kiếm theo họ tên.
  - Hiện thị thông tin về danh sách các cán bộ.
  - Thoát khỏi chương trình.

# Reference (Tham khảo)

- ▶ PGS. TS. Trần Văn Lăng, Lập trình hướng đối tượng (Object – Oriented Programming)
- ▶ Ths Nguyễn Minh Phúc, Bài giảng Lập trình hướng đối tượng với csharp

# TIME TO RELAX

