

# Lớp và đối tượng

Khóa học: Python căn bản

# Kiểm tra bài trước

Hỏi và trao đổi về các khó khăn gặp phải trong bài “List & Tuples & Dictionary”

Tóm tắt lại các phần đã học từ bài “List & Tuples & Dictionary”

# Mục tiêu

---

- Trình bày được mô hình lập trình hướng đối tượng
- Trình bày được các khái niệm lớp, đối tượng, phương thức, thuộc tính, phương thức khởi tạo
- Trình bày được cú pháp khai báo lớp
- Trình bày được cú pháp khởi tạo đối tượng
- Trình bày được cách truy xuất thuộc tính và phương thức của lớp
- Tạo và sử dụng được các đối tượng đơn giản
- Mô tả được lớp bằng biểu đồ

# Thảo luận

Lập trình hướng đối tượng

# Ngôn ngữ lập trình

---

- Ngôn ngữ lập trình cho phép Lập trình viên phát triển các phần mềm
- Các dòng ngôn ngữ lập trình phổ biến:
  - Ngôn ngữ máy (machine languages)
  - Ngôn ngữ assembly
  - Ngôn ngữ bậc cao (high-level languages)

# Ngôn ngữ máy

---

- Được tạo thành từ các số 0 và số 1
- Là ngôn ngữ “native” của máy tính
- Khá khó để lập trình
- Ví dụ:

1110100010101	111010101110
10111010110100	10100011110111

# Ngôn ngữ assembly

---

- Ngôn ngữ assembly dễ lập trình hơn so với ngôn ngữ máy
- Bao gồm một tập các câu lệnh (command) cần thiết được quy định trong các bộ vi xử lý
- Ngôn ngữ assembly cần thiết được chuyển sang ngôn ngữ máy trước khi được thực thi
- Ví dụ:

**ADD 1001010, 1011010**

# Ngôn ngữ bậc cao

---

- Ngôn ngữ bậc cao rất dễ để lập trình so với ngôn ngữ assembly
- Cú pháp của ngôn ngữ bậc cao khá gần với tiếng Anh.
- Ngôn ngữ bậc cao có thể được phân loại thành 2 thể loại phổ biến:
  - Ngôn ngữ thủ tục (procedural languages)
  - Ngôn ngữ Lập trình Hướng đối tượng (OOP - Object Oriented Language)



# Ngôn ngữ Thủ tục

---

- Các ngôn ngữ thủ tục thường là các ngôn ngữ bậc cao ra đời từ rất sớm
- Đặc trưng bởi các tập lệnh tuyến tính nối tiếp nhau
- Tập trung vào cấu trúc (structure) của chương trình
- Ví dụ: C, COBOL, Fortran, LISP, Perl, VBScript

# Ngôn ngữ Hướng đối tượng (OOP)

---

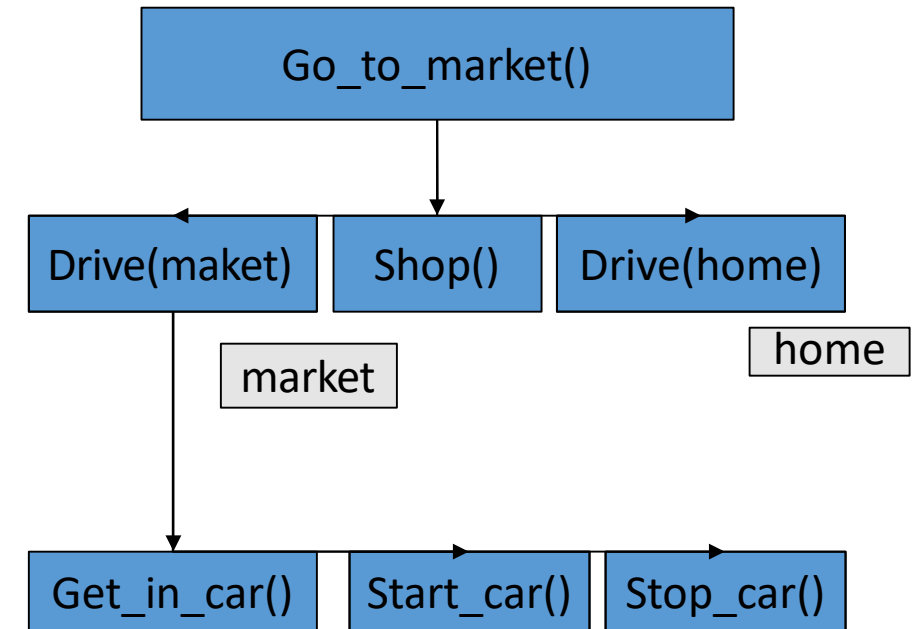
- Không tập trung vào cấu trúc như ngôn ngữ thủ tục, mà tập trung vào mô hình hoá dữ liệu (data modeling)
- Lập trình viên sử dụng các Lớp (class) khi lập trình
- Mô phỏng các đối tượng trong thế giới thực vào trong các chương trình.
- Class: khuôn mẫu của dữ liệu được mô hình hoá
- Ví dụ: C++, C#, Java, PHP, Javascript...

# Ngôn ngữ Hướng đối tượng (OOP)

- OOP (Object Oriented Programming) là một triết lý thiết kế chương trình
- Có nhiều ngôn ngữ lập trình hỗ trợ OOP
- Tất cả mọi thứ trong OOP đều là “đối tượng”. Một chương trình phần mềm được coi như là một thế giới bao gồm các đối tượng tương tác với nhau
- Mã lệnh và dữ liệu được kết hợp trong một thể thống nhất - đó là *đối tượng*.
- Đối tượng bao gồm:
  - Thuộc tính: các dữ liệu, tính chất của đối tượng
  - Hành vi: các khả năng, hành động mà đối tượng có thể thực hiện
- Các đối tượng có thể có quan hệ với nhau

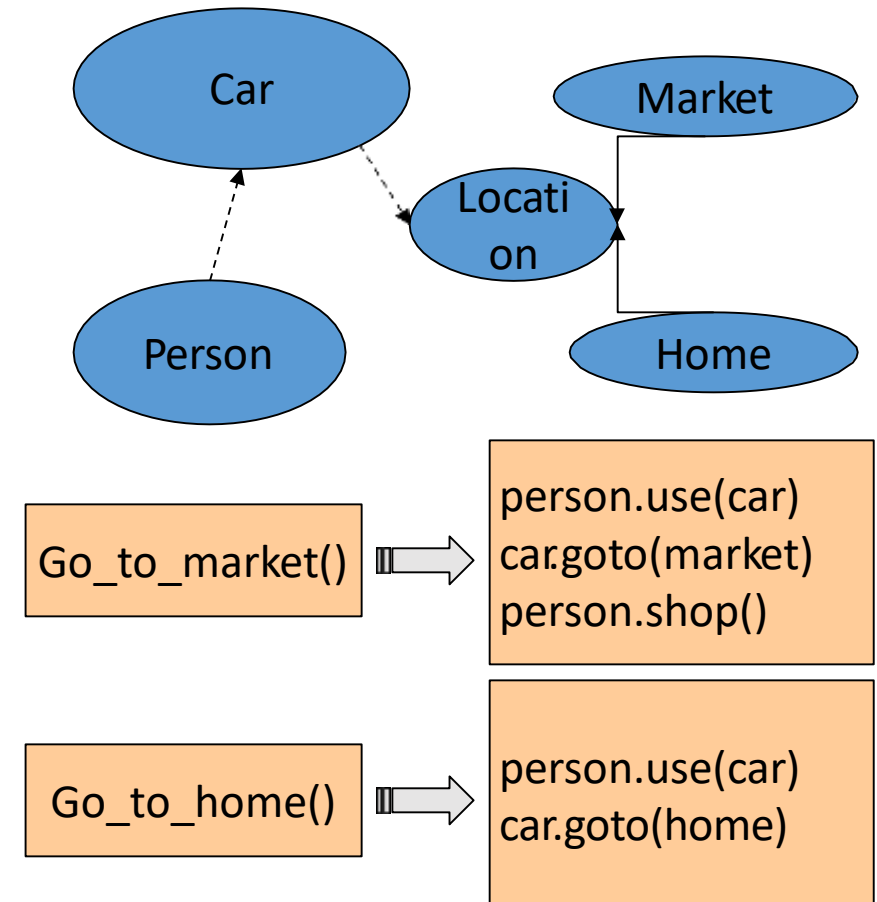
# So sánh Lập trình Thủ tục và OOP - 1

- Lập trình hướng cấu trúc:
  - Hướng tiếp cận: từ trên xuống (top down)
  - Chia nhỏ bài toán thành các module chức năng.
  - Dữ liệu và mã lệnh phân tán.
- Hạn chế:
  - Tính ổn định giảm khi hệ thống phát triển
  - Khó bảo trì và tái sử dụng
  - Chi phí phát triển cao



# So sánh Lập trình Thủ tục và OOP - 2

- Hướng đối tượng
  - Hướng tiếp cận đa dạng gần với thực tế.
  - Tìm và phân tích mối quan hệ giữa các đối tượng trong bài toán
  - Mã lệnh và dữ liệu liên kết trong thể thống nhất.
- Ưu thế:
  - Khả năng tái sử dụng cao
  - Ổn định và dễ bảo trì
  - Chi phí giảm dần.



# Các khái niệm

---

- Đối tượng (Object)
- Lớp (Class)
- Thuộc tính (Property)
- Phương thức (Method)/Hành vi (Behavior)/Hành động (Action)/Khả năng (Capability)
- Các đặc tính cơ bản:
  - Tính bao gói
  - Tính kế thừa
  - Tính trừu tượng
  - Tính đa hình

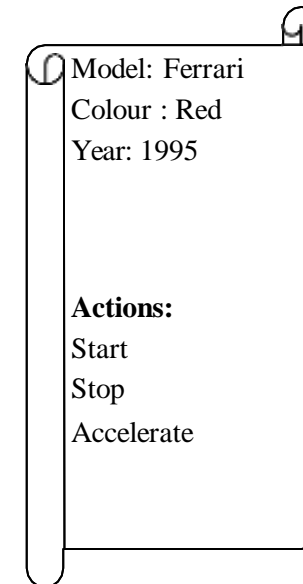
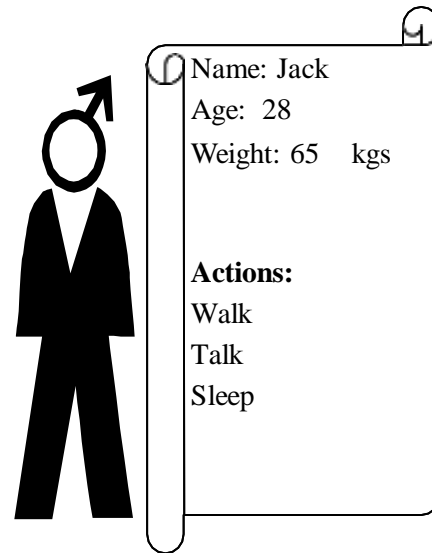
# Đối tượng

- Mỗi đối tượng có những **thuộc tính** hay những đặc điểm mô tả và những **hành vi** riêng nhằm phân biệt nó với các đối tượng khác.



# Thuộc tính và Hành vi

- Thuộc tính là những đặc điểm đặc trưng của đối tượng, thể hiện thông qua những giá trị cụ thể.
- Hành vi là những cách thức mà qua đó đối tượng thể hiện sự hoạt động hay chức năng của chúng.





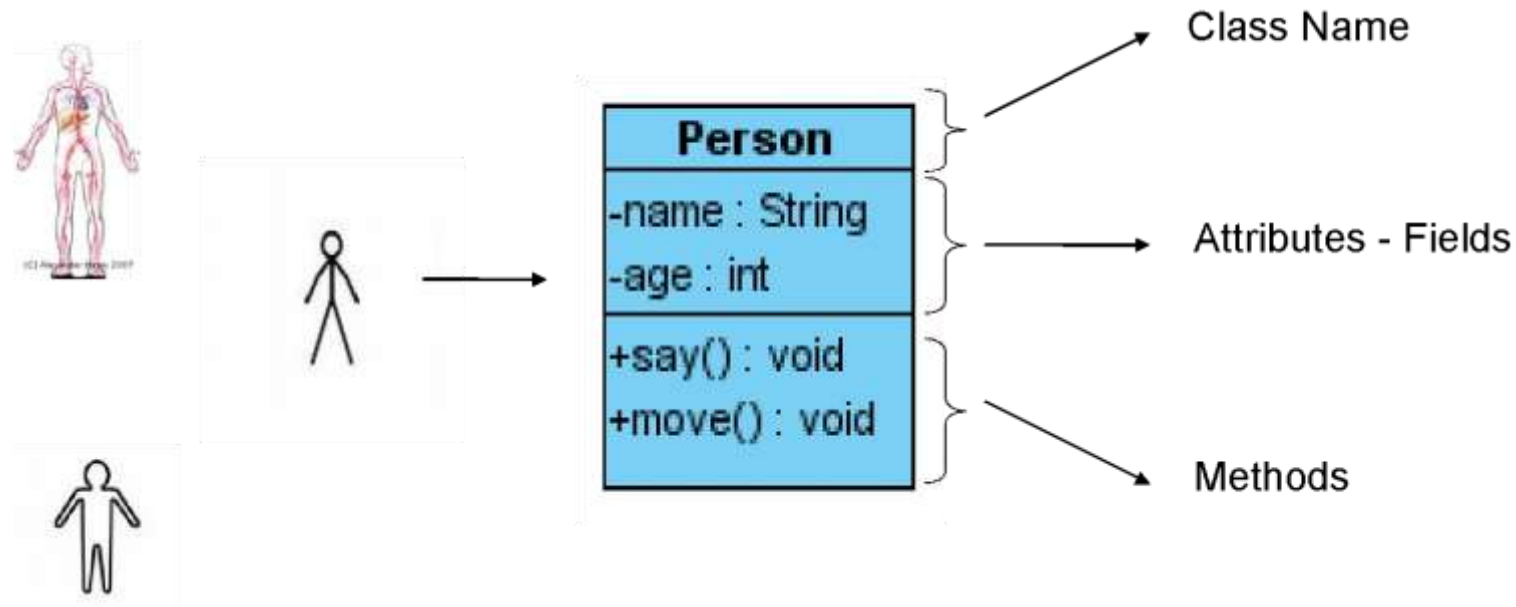
# Lớp

- **Lớp** là khái niệm dùng để mô tả một **loại đối tượng** có những thuộc tính, hành vi và những mối quan hệ thông thường tương tự nhau
- Thuật ngữ lớp có thể hiểu là cách nói vắn tắt của cụm từ “lớp các đối tượng”.
- Như vậy mỗi đối tượng được coi như là một thể hiện của lớp với những giá trị thuộc tính cũng như cách thức hoạt động đặc trưng.



# Sơ đồ mô tả lớp

- Sơ đồ lớp mô tả những đặc điểm khái quát nhất về lớp bao gồm: Tên lớp, danh sách các thuộc tính (tên và loại dữ liệu) và các phương thức



# Demo

Phân tích các thuộc tính và phương thức của đối tượng

# Thảo luận

Tính bao gói

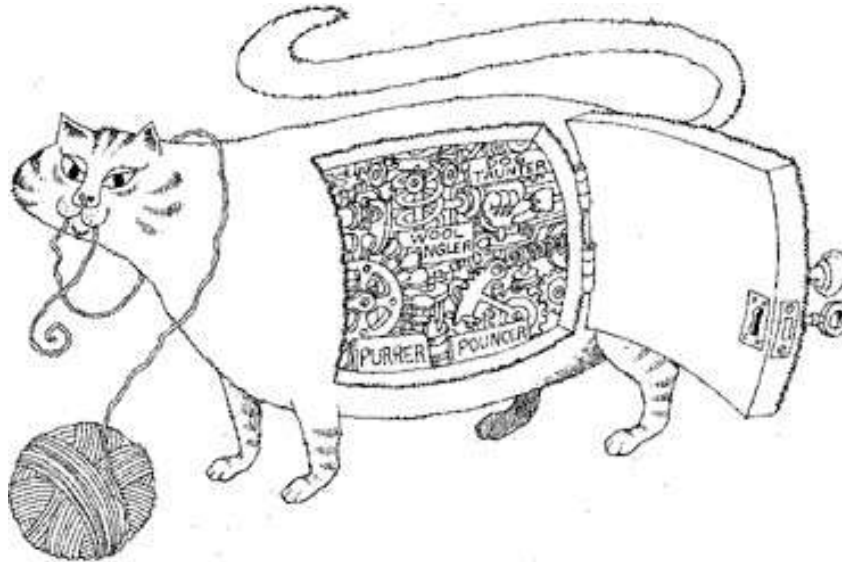
Tính kế thừa

Tính trừu tượng

Tính đa hình

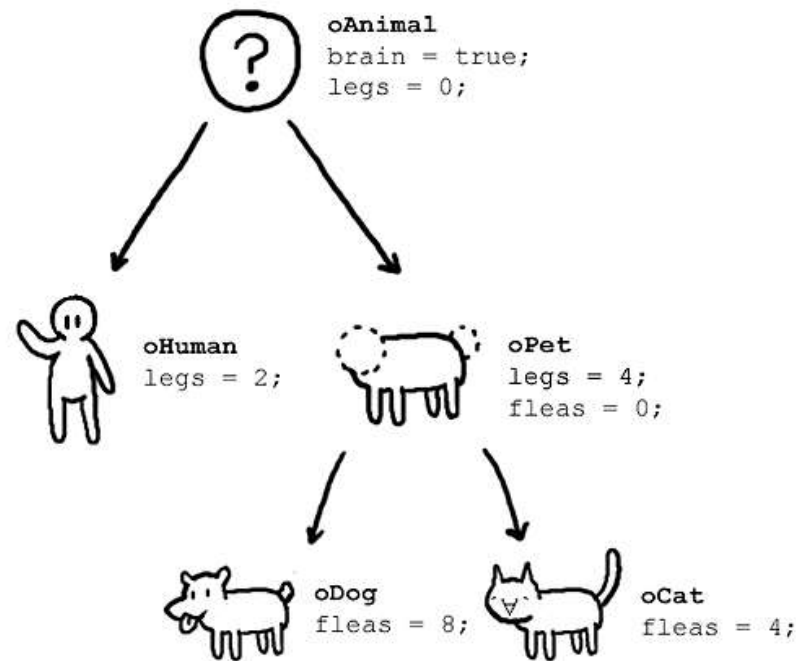
# Tính bao gói (encapsulation)

- Khả năng truy suất vào các thành phần của một đối tượng trong khi vẫn đảm bảo che giấu các đặc tính riêng tư bên trong đối tượng được gọi là tính bao gói.



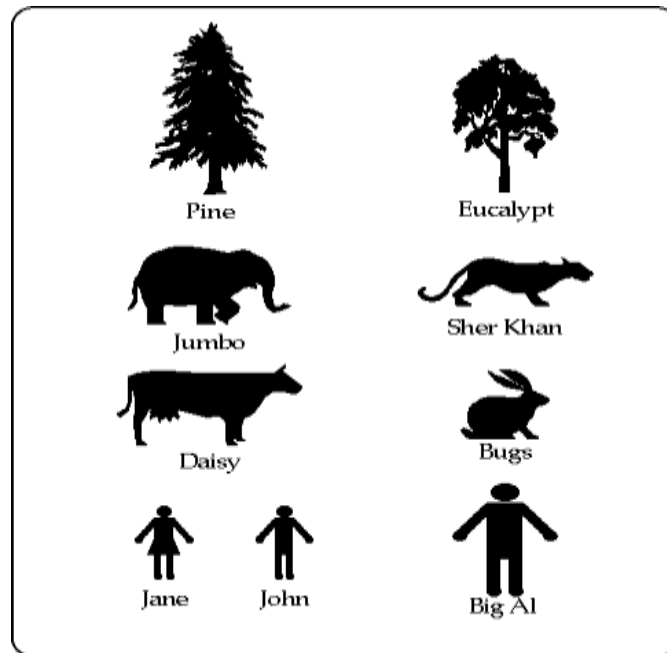
# Tính kế thừa

- Tính kế thừa cho phép các đối tượng có thể chia sẻ hay mở rộng các thuộc tính hoặc phương thức mà không phải tiến hành định nghĩa lại.



# Tính trừu tượng

- Loại bỏ những thuộc tính và hành vi không quan trọng của đối tượng, chỉ giữ lại những thuộc tính và hành vi có liên quan đến vấn đề đang giải quyết



# Tính đa hình

- Tính đa hình thể hiện khi với cùng một phương thức nhưng có thể có cách ứng xử khác nhau ở những lớp cùng giao diện



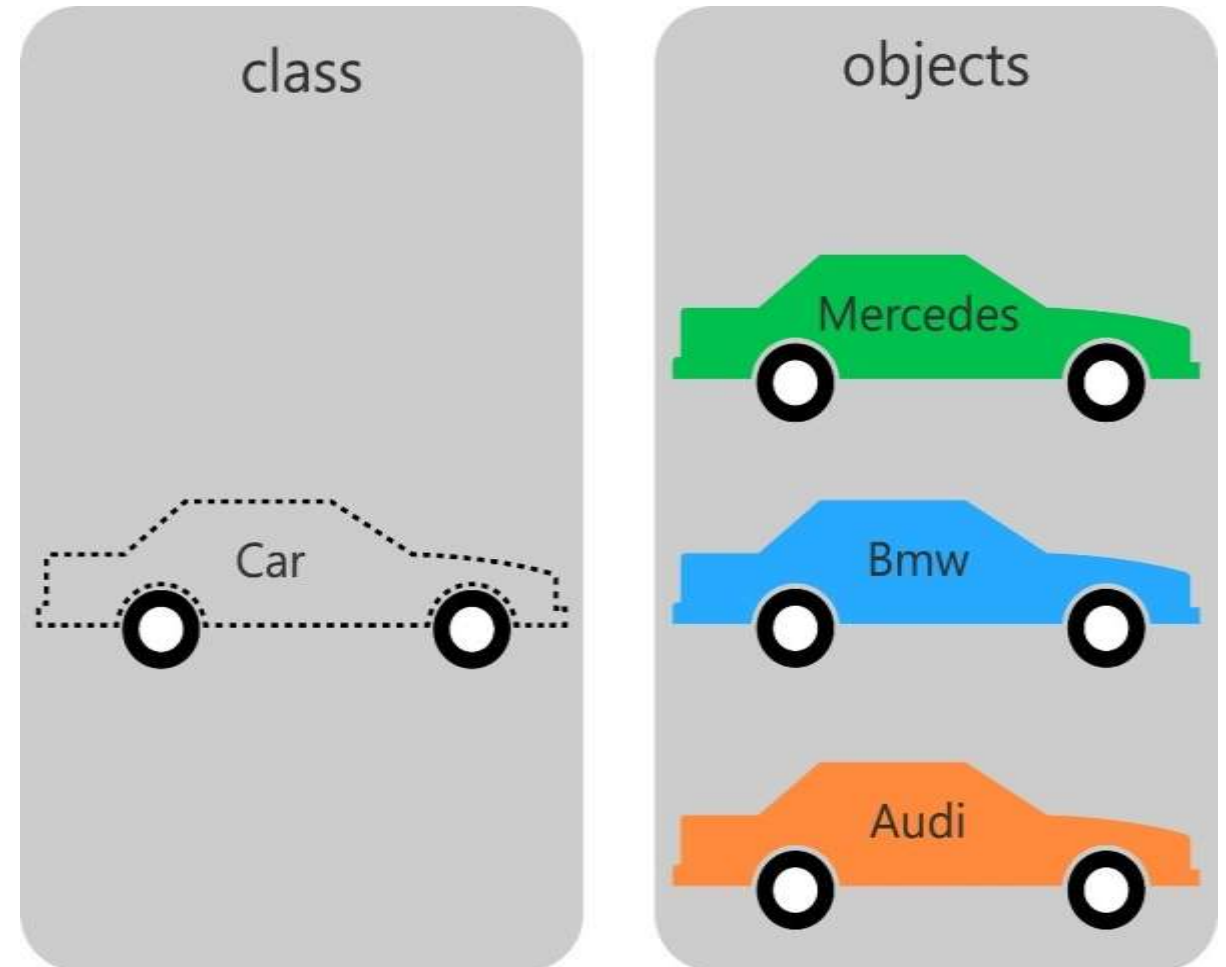


# Thảo luận

Phân biệt Lớp và Đối tượng

# Lớp & Đối tượng

- Lớp là khuôn mẫu chứa những khai báo định dạng và nguyên tắc hoạt động
- Đối tượng là sự thể hiện của một lớp và được coi như những sản phẩm thực sự được tạo ra từ khuôn mẫu đó (lớp).




# Đối tượng

- Các đối tượng là điểm cốt lõi để hiểu về công nghệ hướng đối tượng.
- Đối tượng trong thế giới thực có chung hai đặc điểm: trạng thái và hành vi.
- OOP xây dựng những đối tượng trong phần mềm với hai đặc điểm: thuộc tính và phương thức.
- Các đối tượng được tạo ra từ lớp (class) được xây dựng trước đó.

Object	Properties	Methods
	<code>car.name = Fiat</code> <code>car.model = 500</code> <code>car.weight = 850kg</code> <code>car.color = white</code>	<code>car.start()</code> <code>car.drive()</code> <code>car.brake()</code> <code>car.stop()</code>


# Thuộc tính

- Đối tượng có các thuộc tính (property) dùng để xác định các đặc tính của nó.
- VD: *xe hơi có thuộc tính dòng xe, mẫu xe, trọng lượng, màu sắc*

Object	Properties	Methods
	<div>car.name = Fiat</div> <div>car.model = 500</div> <div>car.weight = 850kg</div> <div>car.color = white</div>	<div>car.start()</div> <div>car.drive()</div> <div>car.brake()</div> <div>car.stop()</div>

# Phương thức

- Đối tượng còn có các phương thức (method) dùng để định nghĩa cách thức hoạt động của nó.
- VD: *xe hơi có phương thức start(), drive(), break(), stop()*

Object	Properties	Methods
	<p>car.name = Fiat</p> <p>car.model = 500</p> <p>car.weight = 850kg</p> <p>car.color = white</p>	<p>car.start()</p> <p>car.drive()</p> <p>car.brake()</p> <p>car.stop()</p>

# Định nghĩa một lớp

- Trong OOP, định nghĩa một lớp đồng nghĩa với việc tạo ra một kiểu dữ liệu mới.
- VD định nghĩa một lớp với các thuộc tính:

```
class Car:
    name = ""                //Thuộc tính
    model = 0                //Thuộc tính
    def __init__(self, name, model): //Phương thức
        self.name = name
        self.model = model
    def start():              //Phương thức
        print("Starting ...")
```

- Từ khóa **self** để chỉ chính đối tượng đó, thuộc sở hữu của đối tượng đó.

# Tạo đối tượng từ lớp

---

- Các đối tượng được khai báo như là một biến có kiểu là lớp tương ứng
- VD tạo đối tượng từ lớp Car và gọi phương thức start() của nó:

```
carobj = Car("Hyundai", "i10", 1200, "White")
```

```
carobj.start()
```

# Sử dụng thuộc tính

---

- Truy xuất giá trị của thuộc tính của đối tượng

`objectName.property     //carobj.name`



# Sử dụng phương thức

---

- Gọi một phương thức của đối tượng  
`objectName.methodName(parameters) ;`
- Truyền tham số khi gọi phương thức của đối tượng

```
myMother.changeName("Doe");
```

# Demo

Lớp Car

# Thảo luận

Quy tắc đặt tên lớp

Phương thức khởi tạo

# Quy tắc đặt tên lớp

---

- Tên lớp nên là một danh từ
- Ký tự đầu tiên của tên lớp phải viết hoa và được đặt tên theo quy tắc Pascal tức ký tự đầu tiên của mỗi từ phải viết hoa
- Tên lớp đơn giản, có ý nghĩa
- Không dùng từ khoá để đặt tên lớp (từ khoá: var, function, new ...)
- Tên lớp không được bắt đầu bởi một số, nên bắt đầu là một ký tự trong bảng chữ cái (A-Z\_)
- Ví dụ: Student, Employee, NullPointerException ...

# Phương thức khởi tạo

---

- Phương thức khởi tạo (Constructor) là một phương thức đặc biệt với các đặc điểm:
  - Tên phương thức trùng với tên lớp
  - Không có kiểu dữ liệu trả về
  - Có các tham số đầu vào như phương thức thông thường khác
- Phương thức khởi tạo được sử dụng để:
  - Tạo đối tượng
  - Khởi tạo các giá trị ban đầu cho các thuộc tính của lớp

# Phương thức khởi tạo

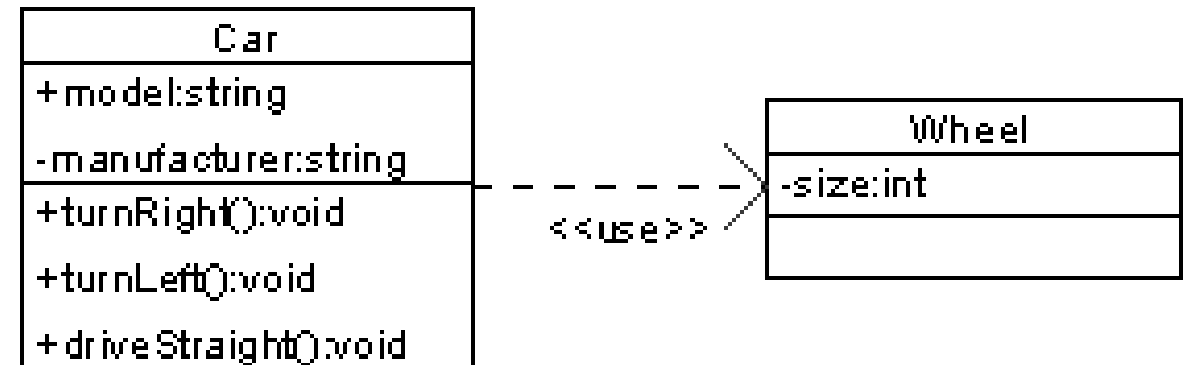
---

- Phương thức khởi tạo được gọi ngay khi một đối tượng được tạo ra
- Ví dụ:
  - Trong lớp Car có phương thức khởi tạo là

```
def __init__(self, name, model):    //Phương thức khởi tạo
    self.name = name
    self.model = model
```

# Mô tả lớp bằng biểu đồ

- Có thể sử dụng những cách khác nhau để mô tả lớp
- Các ký hiệu UML là cách thông dụng để mô tả lớp và nhiều ký hiệu khác
- Để mô tả một lớp cơ bản, cần xác định:
  - Tên lớp
  - Các thuộc tính
  - Các phương thức
  - Các hàm khởi tạo



# Sơ đồ lớp Car

## Car

- name: string
- model: string
- weight: double
- color: string

- + Car(name, model, weight, color)
- + start()
- + dive()
- + stop()
- + brake()
- + toString()

Trong sơ đồ lớp Car gồm:

- 4 thuộc tính name kiểu chuỗi, model kiểu chuỗi, weight kiểu số thực, color kiểu chuỗi.
- Phương thức khởi tạo Car() có 4 tham số truyền vào
- Các phương thức: start(), dive(), stop(), brake(), toString()

Hãy xây dựng lớp hình tròn theo sơ đồ lớp trên



# Sơ đồ lớp Apple

## Apple

```
- type: string  
- color: string  
  
+ Apple(type)  
+ getInfo()  
+ setColor(color)  
+ setType(type)
```

Trong sơ đồ lớp Apple gồm:

- 2 thuộc tính type kiểu chuỗi, color kiểu chuỗi.
- Phương thức khởi tạo Apple() có 1 tham số truyền vào
- Các phương thức:
  - getInfo() trả về thông tin của táo gồm type và color
  - setColor() thiết đặt màu khác cho táo
  - setType() thiết đặt loại khác cho táo

Hãy xây dựng lớp hình tròn theo sơ đồ lớp trên

# Sơ đồ lớp Hình Tròn

## Circle

- radius: double  
- color: string

+ Circle()  
+ getRadius(): double  
+ getArea(): double

Lớp Circle (hình tròn) gồm:

- Thuộc tính:
  - bán kính (radius) sẽ nhận vào giá trị dạng số thực
  - màu sắc (color) sẽ nhận vào giá trị dạng chuỗi.
- Phương thức:
  - Circle() là phương thức khởi tạo để tạo đối tượng không tham số.
  - getRadius() là phương thức trả về bán kính của hình tròn
  - getArea() là phương thức trả về diện tích hình tròn theo công thức  $S = \text{Math.PI} * \text{radius} * \text{radius}$

Hãy xây dựng lớp hình tròn theo sơ đồ lớp trên

# Demo

Tạo lớp và đối tượng

# Destructor

---

- Destructors được gọi khi hủy đối tượng của lớp.
- Trong Python hàm hủy đóng vai trò không quan trọng vì Python có trình dọn rác sẽ đảm nhiệm chức năng tự hủy đối tượng khi kết thúc chương trình và quản lý bộ nhớ một cách tự động

```
def del(self) :
```

```
# Thân của hàm hủy
```

# Tóm tắt bài học

---

- OOP (Object Oriented Programming) là một triết lý thiết kế chương trình
- Lớp là khuôn mẫu chứa những khai báo định dạng và nguyên tắc hoạt động
- Đối tượng là sự thể hiện của một lớp và được coi như những sản phẩm thực sự được tạo ra từ khuôn mẫu đó (lớp).
- Phương thức khởi tạo được sử dụng để:
  - Tạo đối tượng
  - Khởi tạo các giá trị ban đầu cho các thuộc tính của lớp

# Hướng dẫn

Hướng dẫn làm bài thực hành và bài tập

Chuẩn bị bài tiếp theo