Hướng dẫn lập trình OOP với Python Part I – Định nghĩa class
Giáo viên: Nguyễn Hùng Cường

Phần I – Định nghĩa class

Để định nghĩa một class trong Python, ta sử dụng từ khóa class. Sau đó, ta định nghĩa thân của lớp, trong thân lớp, ta có thể định nghĩa các phương thức của lớp, hoặc khai báo các thuộc tính của lớp.

Bước 1: Định nghĩa class

Mở PyCharmIDE, tạo mới một Project Python. Đặt tên và chọn đường dẫn chứa Project vừa tạo.

Tiếp theo, ta tạo một class trong Project, ở đây ta đặt tên class là nhanvien, sau đó ta định nghĩa 2 method thuộc class nhanvien.

```
#dinh nghia class nhanvien
class nhanvien():
    def hienthi(self):
        print("Chao mung cong ty !")

    #dinh nghia method trong class nhanvien
    def hienthichitiet(self, hoten, luong):
        print("Ho ten la: " + str(hoten) + ". Luong la: " + str(luong))
```

Bước 2: Khởi tạo đối tượng của lớp và gọi method

Sau khi đã định nghĩa nội dung của class, ta khởi tạo đối tượng của lớp như hình bên dưới.

```
#khoi tao doi tuong thuoc lop nhanvien
nv1 = nhanvien()
```

Sau đó ta gọi các method của đối tượng vừa khởi tạo.

```
#goi cac method cua doi tuong nv
nv1.hienthi()
nv1.hienthichitiet("Tony Nguyen", 1000000)
```

Bước 3: Thực thi chương trình và xem kết quả
Sau khi đã viết mã xong, hãy thực thi chương trình và xem kết quả.

```
Chao mung cong ty !
Ho ten la: Tony Nguyen. Luong la: 1000000
```

Phần II – Định nghĩa constructor của lớp

Constructor: Là method sẽ được sử dụng để khởi tạo các thuộc tính của lớp. Constructor sẽ được gọi mỗi khi ta khởi tạo một đối tượng của lớp. Constructor trong Python được định nghĩa với method __init__()

- Phương thức _init_(): Tất cả các class sẽ đều có một function được gọi là _init_(), đây là function sẽ luôn được thực thi khi ta khởi tạo một đối tượng của lớp.
Ta có thể sử dụng _init_() để gán giá trị cho các thuộc tính của đối tượng.

Bước 1: Định nghĩa class và constructor
Ta định nghĩa một class trong Python. Ở đây ta định nghĩa một class với tên là khachhang, cùng với một constructor. Trong class khachhang ta đã khai báo 3 thuộc tính của lớp như hình bên dưới.

```
class khachhang:
    hoten = ""
    tuoi = 0
    diachi = ""

    #dinh nghia constructor cua class
    def __init__(self, name, age, address):
        self.hoten = name
        self.tuoi = age
        self.diachi = address

    def mota(self):
        print("Ho ten = " + self.hoten)
        print("Tuoi = " + str(self.tuoi))
        print("Dia chi = " + self.diachi)
```

Bước 2: Sử dụng class và sau đó xem kết quả

Sau khi đã định nghĩa xong, hay khởi tạo đối tượng của lớp bằng cách gọi constructor vừa định nghĩa. Sau đó gọi method mota() đã được định nghĩa trong lớp.

```
obj = khachhang("Diem My 9X", 23, "Nha Trang")
obj.mota();
```

Tiếp theo, khi thực thi chương trình ta sẽ thấy chương trình đã hiển thị thông tin chi tiết về đối tượng như hình bên dưới.

```
Ho ten = Diem My 9X
Tuoi = 23
Dia chi = Nha Trang
```

* Chú ý về từ khóa self trong Python

- Đối số self đề cập đến bản thân đối tượng hiện tại. Từ khóa self sẽ đại diện đến đối tượng đang được thao tác.

- Đối số self theo quy ước của Python để thể hiện tham số đầu tiên của các instance method trong Python.

* Những điều chú ý khi gọi method trong class:
- Khi gọi các method của lớp, ta không cần phải cung cấp tham số self. Đối số này sẽ được tự động xử lý bởi Python runtime.
- Python runtime sẽ truyền giá trị self khi ta gọi một instance method trên một instance của lớp, cho dù ta có truyền hay không.
- Ta chỉ cần quan tâm đến các non-self arguments khi gọi method.

Chú ý:
Mỗi lớp trong Python đều có sẵn một số thuộc tính cơ bản dựng sẵn (built-in attributes) để cung cấp thông tin mô tả về lớp đó.
Ý nghĩa của một số built-in attributes thông dụng như sau:

Tên thuộc tính	Ý nghĩa của thuộc tính
__dict__	Trả về một đối tượng dictionary chứa thông tin về namespace mô tả về class.
__doc__	Trả về một chuỗi mô tả về lớp, hoặc none nếu lớp không có mô tả documentation.
__name__	Trả về tên lớp.
__module__	Trả về tên module trong đó lớp được định nghĩa.
__bases__	Trả về một tuple trống chứa danh sách tên các base class, theo thứ tự

Tiếp theo bài định nghĩa class khachhang ở trên, ta đã truy cập và hiển thị thông tin về các thuộc tính thông dụng của class như sau:

```
#hien thi cac built-in attributes cua class
print("khachhang.doc = " + str(khachhang.__doc__))
print("khachhang.dict = " + str(khachhang.__dict__))
print("khachhang.module = " + str(khachhang.__module__))
print("khachhang.name = " + str(khachhang.__name__))
print("khachhang.base = " + str(khachhang.__bases__))
```

Sau đó khi thực thi chương trình, ta có thể thấy thông tin về các thuộc tính của class đã được hiển thị như hình bên dưới.

```
khachhang.doc = None
khachhang.dict = {'__module__': '__main__', 'hoten': '', 'tuoi': 0, 'diachi': '', '__init__': <function khachhang.__init__ at 0x019AFD20>, 'mota': <function khacl
khachhang.module = __main__
khachhang.name = khachhang
khachhang.base = (<class 'object'>,)
```