



PHẦN 1: LẬP TRÌNH PYTHON		
	NỘI DUNG:	TRANG
1	BIẾN TRONG PYTHON	
2	CÁC KIỂU DỮ LIỆU và TOÁN TỬ	
3	CẤU TRÚC ĐIỀU KHIỂN, VÒNG LẶP	
4	HÀM TRONG PYTHON	
5	LÀM VIỆC VỚI FILE TRONG PYTHON	
6	MODULE, XỬ LÝ NGOẠI LỆ	
7	LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG	

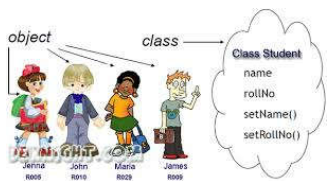
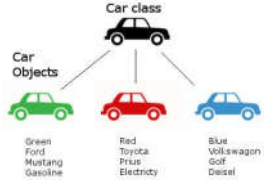
Tạ Quang Chiêu - [E]: quangchieu.ta@gmail.com - [M]: 0913 522 275

114/112

114

LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

Tạ Quang Chiêu - [E]: quangchieu.ta@gmail.com - [M]: 0913 522 275

115/112

115

A

D

A

I

Lớp, đối tượng trong Python

Một số khái niệm

- Python là một ngôn ngữ lập trình hướng đối tượng

➤ **Một số khái niệm hướng đối tượng**

- **Lớp:** được định nghĩa bởi người dùng cho một đối tượng gồm một tập hợp các thuộc tính mà xác định rõ bất kỳ đối tượng nào của lớp đó. Các thuộc tính là các thành viên dữ liệu (các biến class và biến instance) và các phương thức được truy cập thông qua toán tử dot (dấu chấm .).
- **Biến lớp - class:** Đây là một biến được chia sẻ bởi tất cả các instance (sự thể hiện) của một lớp. Các biến class được định nghĩa bên trong một lớp nhưng ở bên ngoài bất cứ phương thức nào của lớp đó. Biến class không được sử dụng thường xuyên như biến instance.
- **Thành viên dữ liệu:** Là một biến class hoặc biến instance mà giữ dữ liệu được liên kết với một lớp và các đối tượng của nó.

Tạ Quang Chiêu - [E]: quangchieu.ta@gmail.com - [M]: 0913 522 275

116/112

116

A

D

A

I

Lớp, đối tượng trong Python

➤ **Một số khái niệm hướng đối tượng**

- **Trình khởi tạo:** Là trình tạo một sự thể hiện của một lớp.
- **Phương thức:** Một loại hàm đặc biệt mà được định nghĩa trong một phần định nghĩa lớp.

➤ **Khai báo Class trong Python**

```
class ClassName:
    'Gồm các thuộc tính, phương thức'
    # Code ...
```

- Trong đó, **className** là tên của class cần khai báo.

Tạ Quang Chiêu - [E]: quangchieu.ta@gmail.com - [M]: 0913 522 275

117/112

117

A

D

A

I

Lớp, đối tượng trong Python

➤ **Ví dụ 1:**

Bất cứ phương thức nào của Python cũng đều phải có tham số đầu tiên là **self** rồi mới đến các tham số khác. **self** thực ra chỉ là biến đối tượng đã gọi phương thức này.

```
class Person:
    # thuộc tính
    name = "Vũ Thanh Tài";
    age = 22;
    male = "Nam"
    # phương thức
    def setName(self, name):
        self.name = name

    def getName(self):
        return self.name

    def setAge(self, age):
        self.age = age

    def getAge(self):
        return self.age

    def setMale(self, male):
        self.male = male

    def getMale(self):
        return self.male
```

EXAMPLE

Tạ Quang Chiêu - [E]: quangchieu.ta@gmail.com - [M]: 0913 522 275

118/112

118

A

D

A

I

Lớp, đối tượng trong Python

➤ **Ví dụ 2:**

```
1 class Sinhvien:
2     'Class co so chung cho tat ca sinh vien'
3     svCount = 0
4
5     def __init__(self, ten, hocphi):
6         self.ten = ten
7         self.hocphi = hocphi
8         Sinhvien.svCount += 1
9
10    def displayCount(self):
11        print ("Tong so Sinh vien %d" % Sinhvien.svCount)
12
13    def displaySinhvien(self):
14        print ("Ten : ", self.ten, ", Hoc phi: ", self.hocphi)
```

EXAMPLE

Tạ Quang Chiêu - [E]: quangchieu.ta@gmail.com - [M]: 0913 522 275

119/112

119

A

D

A

I

Lớp, đối tượng trong Python

➤ **Khởi tạo class.**

Sau khi đã khai báo được class trong Python rồi, thì để khởi tạo nó sử dụng cú pháp sau:

```
variableName = className()
```

Trong đó:

- **variableName** là biến thể hiện lại đối tượng.
- **className** là class muốn khởi tạo.

➤ **Ví dụ:** Khởi tạo class person ở trên.

```
# instance
person = Person()
```

Tạ Quang Chiếu - [E]: quangchieu.ta@gmail.com - [M]: 0913 522 275

120/112

120

A

D

A

I

Lớp, đối tượng trong Python

- Sau khi đã khởi tạo được class rồi thì **biến** sẽ có thể truy cập được các phần tử được cho phép trong class đó.
- Bằng cách sử dụng dấu **.** theo cú pháp sau:

```
# truy cập den thuoc tinh
object.propertyName

#truy cap den phuong thuc
object.methodName()
```

Trong đó:

- **object** là biến thể hiện lại object.
- **propertyName** là tên thuộc tính muốn truy xuất.
- **methodName** là tên phương thức muốn truy xuất.

Tạ Quang Chiếu - [E]: quangchieu.ta@gmail.com - [M]: 0913 522 275

121/112

121

A

D

A

I

Lớp, đối tượng trong Python

➤ **Ví dụ:** sẽ truy xuất đến các thuộc tính và phương thức trong class Person

```
# properties
print(person.name) # Vũ Thanh Tài
print(person.age) # 22
print(person.male) # True
# methods
person.setName("Nguyễn Thị HH")
print(person.getName()) # Nguyễn Thị HH

person.setAge(22)
print(person.getAge()) # 22

person.setMale("Nữ")
print(person.getMale()) # False
```

EXAMPLE

Tạ Quang Chiêu - [E]: quangchieu.ta@gmail.com - [M]: 0913 522 275

122/112

122

A

D

A

I

Làm thế nào để tạo ra một class trống?

Để tạo ra một class trống (empty class) bằng cách sử dụng câu lệnh **pass** trong Python

```
1 # Tạo đối tượng trống trong Python
2 # An empty class
3 class Test:
4     pass
```

Tạ Quang Chiêu - [E]: quangchieu.ta@gmail.com - [M]: 0913 522 275

123/112

123

5

A
A

D
I

Che giấu dữ liệu

Sử dụng **hai dấu gạch dưới** liên tiếp (tức là `__`) **trước tên của các thuộc tính**, và các thuộc tính đó sẽ không thể được nhìn thấy trực tiếp, khi ở bên ngoài lớp.

Tạ Quang Chiêu - [E]: quangchieu.ta@gmail.com - [M]: 0913 522 275
124/112

124

A
A

D
I

Che giấu dữ liệu

➤ Ví dụ về che giấu dữ liệu

```

1 #Ví dụ về che giấu dữ liệu
2 class MyClass:
3     # Hidden member of MyClass
4     __hiddenVariable = 0
5     # A member method that changes
6     # __hiddenVariable
7     def add(self, increment):
8         self.__hiddenVariable += increment
9         print (self.__hiddenVariable)
10 # Driver code
11 myObject = MyClass()
12 myObject.add(2)
13 myObject.add(5)
14 # This line causes error
15 print (myObject.__hiddenVariable)

```

Output

```

2
7
-----
AttributeError                                Traceback (most recent call last)
<ipython-input-5-5e372b8e5544> in <module>()
     13 myObject.add(5)
     14 # This line causes error
--> 15 print (myObject.__hiddenVariable)
AttributeError: 'MyClass' object has no attribute '__hiddenVariable'

```

Tạ Quang Chiêu - [E]: quangchieu.ta@gmail.com - [M]: 0913 522 275
125/112

125

A

D

Che giấu dữ liệu

➤ Ví dụ về che giấu dữ liệu

- Chúng ta có thể truy cập đến giá trị của thuộc tính được che giấu bằng cách sau:

```

1 #Ví dụ về che giấu dữ liệu
2 class MyClass:
3     # Hidden member of MyClass
4     __hiddenVariable = 0
5     # A member method that changes
6     # __hiddenVariable
7     def add(self, increment):
8         self.__hiddenVariable += increment
9         print (self.__hiddenVariable)
10 # Driver code
11 myObject = MyClass()
12 myObject.add(2)
13 myObject.add(5)
14 # This line causes error
15 print (myObject._MyClass__hiddenVariable)
```

Output

2

7

7

Tạ Quang Chiêu – [E]: quangchieu.ta@gmail.com – [M]: 0913 522 275

126/112

126

A

D

Kế thừa

- Kế thừa là định nghĩa một lớp dựa trên một lớp đã được định nghĩa trước đó. Lớp kế thừa từ lớp khác được gọi là lớp dẫn xuất,
- Lớp được các lớp khác kế thừa mình thì gọi là lớp cơ sở.
- Lớp dẫn xuất có thể kế thừa hoặc mở rộng các tính năng của lớp cơ sở.

Tạ Quang Chiêu – [E]: quangchieu.ta@gmail.com – [M]: 0913 522 275

127/112

127

A

D

A

I

Kế thừa

Ví dụ:

```

1 #inherit.py
2 #Lớp cơ sở
3 class Animal:
4     def __init__(self):
5         print ("Animal created")
6
7     def whoAmI(self):
8         print ("Animal")
9
10    def eat(self):
11        print ("Eating")
12 #Lớp kế thừa
13 class Dog(Animal):
14     def __init__(self):
15         Animal.__init__(self)
16         print ("Dog created")
17
18     def whoAmI(self):
19         print ("Dog")
20
21     def bark(self):
22         print ("Woof!")
23 d = Dog()
24 d.whoAmI()
25 d.eat()
26 d.bark()

```

Output

```

Animal created
Dog created
Dog
Eating
Woof!

```

- Để kế thừa một lớp thì chúng ta đặt tên lớp đó bên trong cặp dấu ngoặc tròn () ngay phía sau phần định nghĩa tên lớp.
- Nếu bên trong lớp cơ sở đã định nghĩa phương thức __init__(), chúng ta phải gọi lại phương thức __init__() từ lớp cơ sở.

Tạ Quang Chiêu - [E]: quangchieu.ta@gmail.com - [M]: 0913 522 275

128/112

128

A

D

A

I

Abstraction

Abstract class là gì?

Abstract class là một class mà bên trong nó chứa một hoặc nhiều phương thức trừu tượng.

- **Phương thức trừu tượng** ở đây là một phương thức mà chúng ta chỉ được phép khai báo nó và không được phép viết code thực thi nó.
- Khi một class được khai báo ở dạng abstract thì nó sẽ không thể nào khởi tạo được, mà chỉ có thể khởi tạo được thông qua các class con của nó.
- Một class kế thừa lại abstract class thì phải khai báo lại toàn bộ các phương thức trừu tượng bên trong abstract class mà nó kế thừa.

Tạ Quang Chiêu - [E]: quangchieu.ta@gmail.com - [M]: 0913 522 275

129/112

129

A D
A I

Abstraction

Khai báo abstract class trong Python

- Để có thể khai báo được một **abstract class** trong Python, thì class này bắt buộc phải được kế thừa từ một ABC (Abstract Base Classes) của Python
- Và để gọi được class này trong chương trình thì bạn phải import nó.
- Cú pháp import như sau:

`from abc import ABC`
- Cú pháp khai báo abstract class:

`class ClassName(ABC):
code`

ClassName là tên của abstract class mà bạn muốn khai báo.

Tạ Quang Chiêu - [E]: quangchieu.ta@gmail.com - [M]: 0913 522 275
130/112

130

A D
A I

Abstraction

Khai báo abstract class trong Python

VD: Khai báo một lớp trừu tượng person.

```

from abc import ABC, abstractmethod
class PersonAbstract(ABC):
    name = None
    age = 0
    def getName(self):
        print(self.name)
    def getAge(self):
        print(self.age)

```

Vì abstract class là một class, nên các bạn có thể hoàn toàn khai báo thuộc tính và phương thức như một class bình thường.

Tạ Quang Chiêu - [E]: quangchieu.ta@gmail.com - [M]: 0913 522 275
131/112

131

A

D

A

I

Abstraction

Khai báo phương thức abstract trong Python

Để có thể khai báo một abstract method - phương thức trừu tượng trong Python thì chúng ta cần phải import thêm module `abstractmethod` ở trong package `abc`.

```
from abc import ABC, abstractmethod
```

Và một phương thức trừu tượng thì bắt buộc phải được khai báo ở trong lớp trừu tượng.

Tạ Quang Chiêu - [E]: quangchieu.ta@gmail.com - [M]: 0913 522 275

132/112

132

A

D

A

I

Abstraction

Khai báo phương thức abstract trong Python

Cú pháp:

```
from abc import ABC, abstractmethod
class ClassName(ABC):
    # khai bao phuong thuc tru tuong
    @abstractmethod
    def methodName(self):
        pass
```

Trong đó:

`@abstractmethod` là bắt buộc, đây là cú pháp khai báo cho Python biết phía dưới là phương thức trừu tượng.

`methodName` là tên của phương thức trừu tượng.

Tạ Quang Chiêu - [E]: quangchieu.ta@gmail.com - [M]: 0913 522 275

133/112

133

A D
A I

Abstraction

Khai báo phương thức abstract trong Python

VD: Khai báo thêm phương thức trừu tượng getFull vào trong PersonAbstract của VD trên.

```

from abc import ABC, abstractmethod
class PersonAbstract(ABC):
    name = None
    age = 0
    def getName(self):
        print(self.name)
    def getAge(self):
        print(self.age)
    @abstractmethod
    def getFull(self):
        pass

```

Tạ Quang Chiếu - [E]: quangchieu.ta@gmail.com - [M]: 0913 522 275
134/112

134

A D
A I

Abstraction

Các Ví Dụ

Đầu tiên, để chứng minh là abstract class không thể khởi tạo được một các trực tiếp thì mình sẽ thử khởi tạo class PersonAbstract ở ví dụ trên xem sao:

```

from abc import ABC, abstractmethod
class PersonAbstract(ABC):
    name = None
    age = 0
    def getName(self):
        print(self.name)
    def getAge(self):
        print(self.age)
    @abstractmethod
    def getFull(self):
        pass

PersonAbstract()
# Lỗi
# Can't instantiate abstract class PersonAbstract with abstract methods getFull

```

Tạ Quang Chiếu - [E]: quangchieu.ta@gmail.com - [M]: 0913 522 275
135/112

135

A
D

A
I

Abstraction

Các Ví Dụ

Sẽ tạo một class và kết thừa lại PersonAbstract này và đồng thời khởi tạo nó xem sao.

```
from abc import ABC, abstractmethod
class PersonAbstract(ABC):
    name = None
    age = 0
    def getName(self):
        print(self.name)
    def getAge(self):
        print(self.age)
    @abstractmethod
    def getFull(self):
        pass
class Person(PersonAbstract):
    pass
Person();
# Lỗi
# Can't instantiate abstract class Person with abstract methods getFull
```

Nó vẫn báo lỗi, bây giờ ở class con này chúng ta sẽ khai báo lại method getFull và khởi tạo lại nó.

Tạ Quang Chiêu - [E]: quangchieu.ta@gmail.com - [M]: 0913 522 275

136/112

136

A
D

A
I

Abstraction

```
from abc import ABC, abstractmethod
class PersonAbstract(ABC):
    name = None
    age = 0
    def getName(self):
        print(self.name)
    def getAge(self):
        print(self.age)
    @abstractmethod
    def getFull(self):
        pass
class Person(PersonAbstract):
    name = 'Vu Thanh Tai'
    age = 22
    def getFull(self):
        self.getName()
        self.getAge()
Person();
```

Lúc này chương trình đã không báo lỗi gì nữa. Bây giờ chúng ta sẽ gọi thử phương thức getFull xem sao.

Tạ Quang Chiêu - [E]: quangchieu.ta@gmail.com - [M]: 0913 522 275

137/112

137

A

D

A

I

Abstraction

```

from abc import ABC, abstractmethod
class PersonAbstact(ABC):
    name = None
    age = 0
    def getName(self):
        print(self.name)
    def getAge(self):
        print(self.age)
    @abstractmethod
    def getFull(self):
        pass
class Person(PersonAbstact):
    name = 'Vu Thanh Tai'
    age = 22
    def getFull(self):
        self.getName()
        self.getAge()
Person().getFull();

```

Kết Quả:
Vu Thanh Tai
22

Tạ Quang Chiêu - [E]: quangchieu.ta@gmail.com - [M]: 0913 522 275

138/112

138

A

D

A

I

Thực hành

Tạ Quang Chiêu - [E]: quangchieu.ta@gmail.com - [M]: 0913 522 275

139/112

139

13

A
D
I
I
Bài tập

Bài 16: Xây dựng lớp phân số

Xây dựng lớp phân số gồm các thành phần:

- **Thuộc tính:** Tử số, Mẫu số
- **Phương thức:** Nhập phân số, in phân số, tính tổng 2 phân số.

Yêu cầu: Nhập 2 phân số, tính tổng và in ra kết quả

Tạ Quang Chiêu – [E]: quangchieu.ta@gmail.com – [M]: 0913 522 275

140/112

140

A
D
I
I
Bài tập

Bài 17: Giải phương trình bậc nhất

Bài 18: Giải phương trình bậc hai

Bài 19: Viết chương trình quản lý các CD(Công Đoàn) như sau:

Người dùng lần lượt nhập thông tin của các CD, chương trình sẽ quản lý thông tin các CD này và in ra danh sách các CD kèm theo tổng số tiền của các CD

```

Thông tin CD:
Nhập tên CD: Happy New Year
Nhập tên ca sỹ: ABBA
Nhập số bài hát: 7
Nhập giá thành: 185000
--- Danh sách CD: ---
# Happy New Year - ABBA - 7 - 185000
Tổng giá thành: 185000
Tiếp tục nhập: 1: Có, 0: Không 1
Nhập tên CD: Chat với Mozart
Nhập tên ca sỹ: Mỹ Linh
Nhập số bài hát: 8
Nhập giá thành: 245000
--- Danh sách CD: ---
# Happy New Year - ABBA - 7 - 185000
# Chat với Mozart - Mỹ Linh - 8 - 245000
Tổng giá thành: 430000
Tiếp tục nhập: 1: Có, 0: Không 0
          
```

Tạ Quang Chiêu – [E]: quangchieu.ta@gmail.com – [M]: 0913 522 275

141/112

141

A D		A I		Bài tập
<p>Bài 20: Tính chu vi & diện tích các hình (abstract)</p> <p>Viết chương trình tính chu vi và diện tích của một số hình như sau:</p> <p>Hình tròn</p> <p>Hình chữ nhật</p> <p>Hình tam giác</p> <p>Bài 21: Xây dựng lớp số phức gồm các thành phần:</p> <ul style="list-style-type: none"> - Dữ liệu gồm phần thực, phần ảo - Phương thức: nhập, in, tính trị tuyệt đối của số phức, tổng, hiệu 2 số phức. <p>Thực hiện:</p> <ul style="list-style-type: none"> - Nhập 2 số phức - Tính và in tổng, hiệu hai số phức 				
Tạ Quang Chiêu – [E]: quangchieu.ta@gmail.com – [M]: 0913 522 275				142/112

142

A D		A I		Bài tập
<p>Bài 22. Xây dựng lớp vector gồm các thành phần:</p> <ul style="list-style-type: none"> - Dữ liệu gồm số chiều n và giá trị của n chiều - Phương thức: nhập, in, tổng 2 vector, tích vô hướng 2 vector <p>Thực hiện:</p> <ul style="list-style-type: none"> - Nhập 2 vector a, b - Tính và in $a+b$, $a*b$ <p>Bài 23. Xây dựng lớp hóa đơn gồm các thành phần:</p> <ul style="list-style-type: none"> - Dữ liệu: mã vật tư, tên vật tư, loại phiếu (nhập/xuất), ngày lập, khối lượng, đơn giá, thành tiền - Phương thức: nhập, in hóa đơn. <p>Hàm main:</p> <ul style="list-style-type: none"> - Nhập danh sách hóa đơn - Tính thành tiền cho các hóa đơn và in tổng thành tiền - In danh sách hóa đơn sau khi sắp xếp theo số tiền giảm dần. 				
Tạ Quang Chiêu – [E]: quangchieu.ta@gmail.com – [M]: 0913 522 275				143/112

143