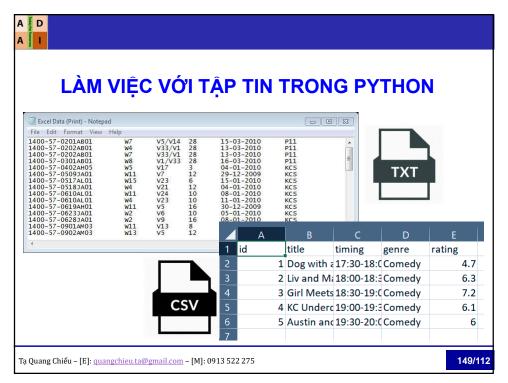
PHẦN 1: LẬP TRÌNH PYTHON		
	NỘI DUNG:	TRANG
1	BIÉN TRONG PYTHON	
2	CÁC KIỂU DỮ LIỆU và TOÁN TỬ	
3	CẦU TRÚC ĐIỀU KHIỂN, VÒNG LẶP	
4	HÀM TRONG PYTHON	
5	LÀM VIỆC VỚI FILE TRONG PYTHON	
6	MODULE, XỬ LÝ NGOẠI LỆ	
7	LẬP TRÌNH HƯỚNG ĐÓI TƯỢNG	
Tạ Quang Chiểu – [Е]: ф	uangchieu.ta@gmail.com - [M]: 0913 522 275	148/112



## A for D

### File là gì?

- File hay còn gọi là tệp, tập tin. File là tập hợp của các thông tin được đặt tên và lưu trữ trên bộ nhớ máy tính như đĩa cứng, đĩa mềm, CD, DVD,...
- Khi muốn đọc hoặc ghi file, chúng ta cần phải mở file trước. Khi hoàn thành, file cần phải được đóng lại để các tài nguyên được gắn với file được giải phóng.
- Do đó, trong Python, một thao tác với file diễn ra theo thứ tự sau:
  - ✓ Mở tệp tin
  - ✓ Đọc hoặc ghi
  - √ Đóng tệp

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

150/112

150



### Mở File

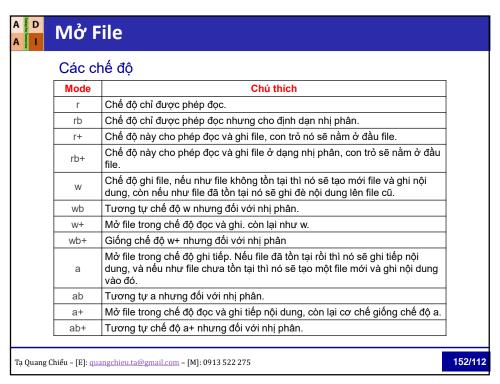
Để mở file trong Python chúng ta sử dụng hàm **open** với cú pháp như sau:

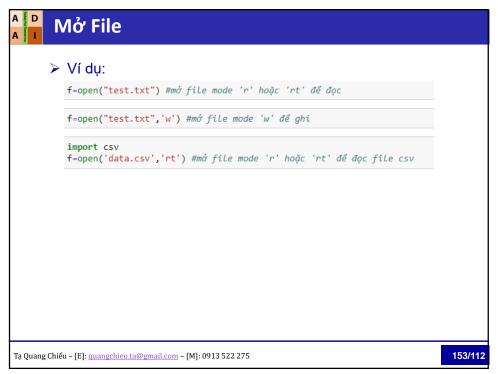
open(filePath, mode, buffer)

#### Trong đó:

- filePath là đường dẫn đến địa chỉ của file.
- mode là thông số thiết lập chế độ chúng ta mở file được cấp những quyền gì? Mặc định mode sẽ bằng r (xem các mode ở dưới).
- buffer là thông số đệm cho file mặc định thì nó sẽ là 0.

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





# A D A I

### Đóng File

Việc đóng file được xây dựng trong Python bằng hàm close() với cú pháp như sau:

### fileObject.close()

 Trong đó, fileObject là đối tượng mà chúng ta thu được khi sử dụng hàm open().

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

154/112

154



### Đọc file

 Sau khi đã mở được file ra rồi, để đọc được file thì chúng ta sử dụng phương thức read với cú pháp:

### fileObject.read(length);

#### Trong đó:

- fileObject là đối tượng mà chúng ta thu được khi sử dụng hàm open().
- length là dung lượng của dữ liệu mà chúng ta muốn đọc, nếu để trống tham số này thì nó sẽ đọc hết file hoặc nếu file lớn quá thì nó sẽ đọc đến khi giới hạn của bộ nhớ cho phép.

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

# Đọc file

Ví dụ: đọc và xuất ra dữ liệu đọc được trong file readme ở trên

```
# Mở file để đọc dữ liệu

fo = open("E:\Python\File\data.txt", "r+")

# Đọc một chuỗi trong file

str = fo.read(20)

print("Chuỗi được đọc là: ", str)

str = fo.read()

# In ra chuỗi được đọc

print("Chuỗi được đọc là: ", str)

# Đóng file lại

fo.close()
```

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

156/112

156

### Ghi file

Để ghi được file thì bạn phải chắc chắn là đang mở file ở các chế độ cho phép ghi. Và sử dụng phương thức write với cú pháp sau:

### fileObject.write(data)

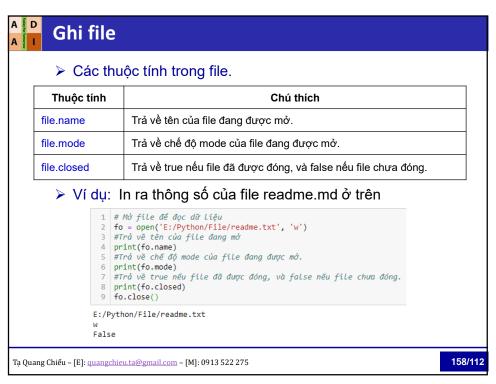
#### Trong đó:

- fileObject là đối tượng mà chúng ta thu được khi sử dụng hàm open().
- data là dữ liệu mà chúng ta muốn ghi vào trong file.
- ➤ Ví dụ:

```
# Mở file để đọc dữ liệu
fo = open('E:/Python/File/readme.txt', 'w')
#Ghi file
fo.write('Ta Quang Chieu - khoa CNTT - HUMG')
# Đóng file lại
fo.close()
```

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

157/112



```
Ví dụ
   Ví du 1: Ghi dữ liệu vào File "data.txt"
        # Mở file để ghi
        fo = open("data.txt", "w")
        # Ghi dữ liệu lên file
        fo.write("Tobe or not tobe. \n Nghi lon de thanh cong ! \n");
        # Close opened file
        fo.close()
        print("Ghi file thanh cong !")
   Ví dụ 2: Đọc và ghi dữ liệu từ một File
               obj=open("test.txt","w")
               obj.write("Chao mung cac ban den voi khoa CNTT")
               obj.close()
               obj1=open("test.txt","r")
               s=obj1.read()
               print (s)
               obj1.close()
               obj2=open("test.txt","r")
               s1=obj2.read(20)
               print (s1)
               obj2.close()
                                                                            159/112
Tạ Quang Chiếu - [E]: quangchieu.ta@gmail.com - [M]: 0913 522 275
```

# Một số hàm khác của file

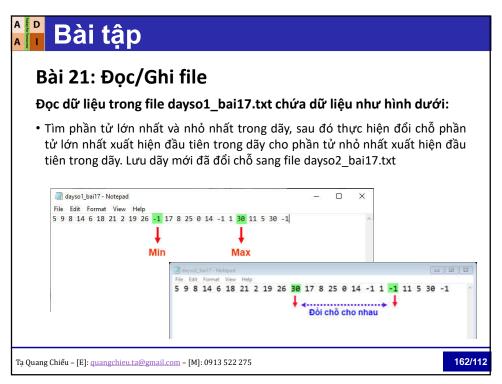
- flush(): ép đẩy các dữ liệu trên bộ nhớ tạm ra file
- tell(): trả về vị trí hiện tại của con trỏ file
- seek(n): dịch con trỏ file đến vị trí byte thứ n
  - Hàm có thêm tham số thứ 2, cho phép diễn giải cách hiểu của tham số n
  - Nếu không viết, hoặc =0: vị trí n tính từ đầu file
  - =1: vị trí n tính từ vị trí hiện tại
  - =2: vị trí n tính từ cuối file
- truncate(n): cắt file ở vị trí byte thứ n, hoặc vị trí hiện tại (nếu không viết giá trị n)

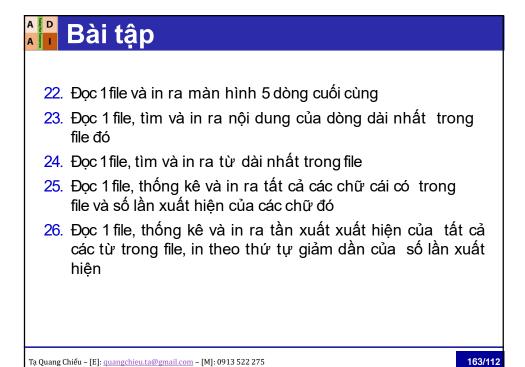
Tạ Quang Chiếu – [E]: quangchieu.ta@gmail.com – [M]: 0913 522 275

160/112

160







PHẦN 1: LẬP TRÌNH PYTHON		
	NỘI DUNG:	TRANG
1	BIÉN TRONG PYTHON	
2	CÁC KIỀU DỮ LIỆU và TOÁN TỬ	
3	CẦU 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 ĐÓI TƯỢNG	
Tạ Quang Chiếu – [Е]: <u>q</u> ı	uangchieu.ta@gmail.com - [M]: 0913 522 275	164/112



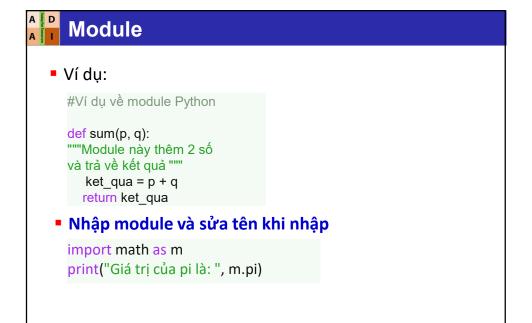
## Module

- Một file mã nguồn trong python được xem là một module
  - Có phần mở rộng .py
  - Mọi hàm, biến, kiểu trong file là các thành phần của module
- Sử dụng module:
  - Khai báo import module đó: import <tên-module>
  - Có thể khai báo import cùng lúc nhiều module cách nhau bởi dấu phẩy
  - Nếu muốn sử dụng các hàm, biến trong module thì cần viết tường minh tên module đó
  - Hoặc có thể import riêng một hàm hoặc nhiều hàm, cú pháp: from <tên-module> import fuc1, fuc2,..., fucN

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

166/112

166



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



Lệnh from...import trong Python

Có thể nhập một tên cụ thể từ module mà không cần nhập toàn bộ module như ví dụ

from math import pi print("Giá trị của pi là: ",pi)

Chúng ta có thể nhập nhiều thuộc tính của module như ví dụ dưới đây:

from math import pi, e

Nhập tất cả tên

form math import \* print("Giá trị của pi là: ",pi)

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

168/112

168

### Module

Đường dẫn tìm kiếm module Python

Nhập lần lượt các lệnh sau để xem đường dẫn:

>>> import sys >>> sys.path

Hàm dir() trong Python

Hàm dir() trong Python được sử dụng để tìm ra các tên được định nghĩa trong một module

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

# Package

- Package = Thư mục các module (lưu trữ vật lý): Package trong Python là một thư mục chứa một hoặc nhiều modules hay các package khác nhau, nó được tạo ra nhằm mục đích phân bố các modules có cùng chức năng hay một cái gì đó, để dễ quản lý source code
- Module và Package giúp quản lý tốt hơn mã nguồn
- Gom, nhóm các hàm, biến, lớp xử lý cùng một chủ đề, giúp phân cấp và sử dụng dễ dàng hơn

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

170/112

170

```
Package chứa package.

|--demopackage/
| --modules.py
| --packagechild/
| | -- childmodule.py
| | __init_py
| --main.py

| Ta Quang Chiếu - [E]: quangchieuta@gmail.com - [M]: 0913 522 275
```



# Ngoại lệ là gì?

- Ngoại lệ = lỗi, đúng, nhưng không hẳn
- Thường người ta chia lỗi thành 3 nhóm
  - Lỗi khi viết chương trình: hệ quả là chương trình không chạy được nếu là thông dịch (hoặc không dịch được, nếu là biên dịch)
  - 2. Lỗi khi chương trình chạy: hệ quả là phải thực hiện lại
    - Chẳng hạn như nhập liệu không đúng, thì phải nhập lại
  - 3. Ngoại lệ: vẫn là lỗi, xảy ra khi có một bất thường và khiến một chức năng không thể thực hiện được
    - Chẳng hạn như đang ghi dữ liệu ra một file, nhưng file đó lại bị một tiến trình khác xóa mất

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

174/112

174

# Ngoại lệ là gì?

- Ranh giới giữa ngoại lệ và lỗi khá mong manh, thậm chí khó phân biệt trong nhiều tình huống
- Cách chia lỗi thành 3 nhóm có khuynh hướng cho rằng môi trường thực thi của chương trình là thân thiên và hoàn hảo
- Python có xu hướng chia lỗi thành 2 loại
  - Syntax error: viết sai cú pháp, khiến chương trình thông dịch không dịch được
  - Exception: xảy ra bất thường không như thiết kế
    - Như vậy xử lý exception sẽ khiến chương trình ổn định và hoạt động tốt trong mọi tình huống

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

```
Ngoại lệ là gì?

Ví dụ về syntax error:

>>> while True print('Hello world')

File "<stdin>", line 1

while True print('Hello world')

SyntaxError: invalid syntax

Ví dụ về exception:

>>> 10 * (1/0)

Traceback (most recent call last):

File "<stdin>", line 1, in ⟨module⟩

ZeroDivisionError: division by zero

Có vẻ như syntax error cũng chỉ là một exception!!!

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

176/112
```

```
Ngoại lệ là gì?
                                                                          Vòng lặp nhập X
   while True:
                                                                  cho đến khi người dùng
                                                                  nhập vào đúng giá trị số
         try:
               x = int(input("Nhập số X: "))
                                                                             Khối nhập X
                break
                                                                        (có thể nhập lỗi)
         except ValueError:
               print("Lỗi,
                                  hãy nhập lại.")
                                                                       Xử lý khi lỗi xảy ra
   print("X = ", x)
                                                                                      177/112
Tạ Quang Chiếu – [E]: <a href="mailto:quangchieu.ta@gmail.com">quangchieu.ta@gmail.com</a> – [M]: 0913 522 275
```

# Cú pháp try-except-else-finally

- Có thể gồm tới 4 khối:
  - Khối "try": đoạn mã có khả năng gây lỗi, khi lỗi xảy ra, khối này sẽ bị dừng ở dòng gây lỗi
  - Khối "except": đoạn mã xử lý lỗi, chỉ thực hiện nếu có lỗi xảy ra, nếu không sẽ bị bỏ qua
  - Khối "else": có thể xuất hiện ngay sau khối except cuối cùng, đoạn mã sẽ được thực hiện nếu không có except nào được thực hiện (đoạn try không có lỗi)
  - Khối "finally": còn được gọi là khối clean-up, luôn được thực hiện dù có xảy ra lỗi hay không

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

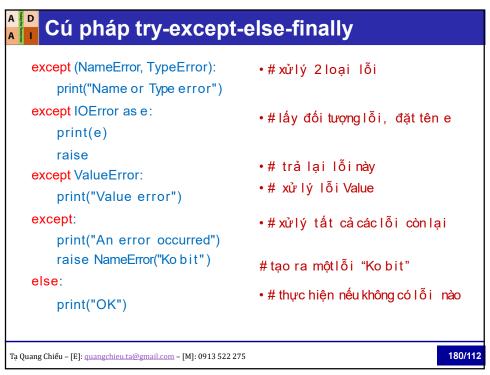
178/112

178

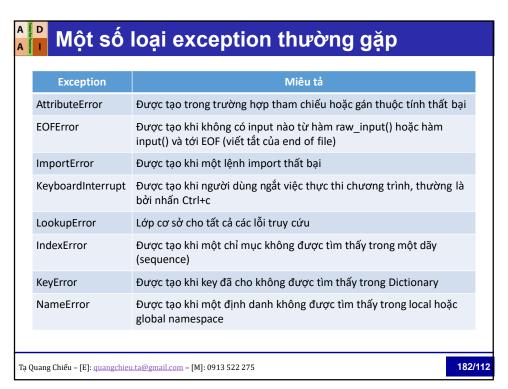
## Cú pháp try-except-else-finally

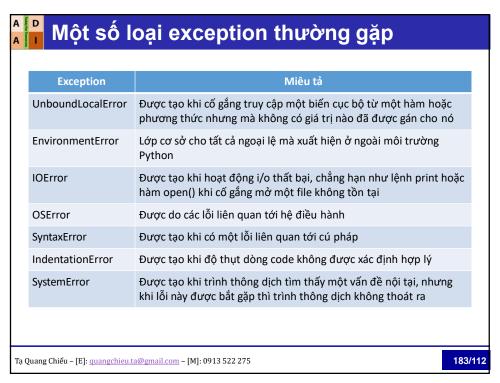
- Chú ý:
  - Khối try chỉ có 1 khối duy nhất, phải viết đầu tiên
  - Khối finally có thể có hay không, nếu có thì khối này phải viết cuối cùng
  - Khối except có thể không viết, có một khối, hoặc nhiều khối except (để xử lý nhiều tình huống lỗi khác nhau)
  - Một khối except có thể xử lý một loại lỗi, nhiều loại lỗi hoặc tất cả các loại lỗi

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



Exception	Miêu tả
Exception	Lớp cơ sở (base class) của tất cả các ngoại lệ
StopIteration	Được tạo khi phương thức next() của một iterator không trỏ tới bất kỳ đối tượng nào
StandardError	Lớp cơ sở của tất cả exception có sẵn ngoại trừ StopIteration và SystemExit
ArithmeticError	Lớp cơ sở của tất cả các lỗi xảy ra cho phép tính số học
OverflowError	Được tạo khi một phép tính vượt quá giới hạn tối đa cho một kiểu số
FloatingPointError	Được tạo khi một phép tính số thực thất bại
ZeroDivisonError	Được tạo khi thực hiện phép chia cho số 0 với tất cả kiểu số
AssertionError	Được tạo trong trường hợp lệnh assert thất bại





### Một số loại exception thường gặp Exception Miêu tả Được tạo khi trình thông dịch thoát ra bởi sử dụng hàm SystemExit sys.exit(). Nếu không được xử lý trong code, sẽ làm cho trình thông dịch thoát Được tạo khi một hoạt động hoặc hàm sử dụng một kiểu dữ TypeError liệu không hợp lệ Được tạo khi hàm đã được xây dựng sẵn có các kiểu tham số ValueError hợp lệ nhưng các giá trị được xác định cho tham số đó là không RuntimeError Được tạo khi một lỗi đã được tạo ra là không trong loại nào Not Implemented ErrorĐược tạo khi một phương thức abstract, mà cần được triển khai trong một lớp được kế thừa, đã không được triển khai thực sự 184/112 Tạ Quang Chiểu – [E]: quangchieu.ta@gmail.com – [M]: 0913 522 275