TRƯỜNG ĐẠI HỌC VINH
VIỆN KỸ THUẬT VÀ CÔNG NGHỆ

CHƯƠNG 7

THAO TÁC TRÊN TẬP TIN VÀ THƯ MỤC TRONG PYTHON

Nghệ An, 2019

Chương 7:
THAO TÁC TRÊN TẬP TIN VÀ THƯ MỤC TRONG PYTHON

NỘI DUNG GIẢNG DẠY:

7.1. Tập tin (File)

7.2. Thư mục (Directory)

Chương 7:
THAO TÁC TRÊN TẬP TIN VÀ THƯ MỤC TRONG PYTHON

NỘI DUNG GIẢNG DẠY:

7.1. Tập tin (File)

7.2. Thư mục (Directory)

TẬP TIN (FILE)

- **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

TẬP TIN (FILE)

- **MỞ FILE TRONG PYTHON:**

- Trước khi làm việc với bất cứ file nào, chúng ta phải mở file đó.

- Để mở một file, Python cung cấp hàm open(), hàm này trả về một đối tượng file mà được sử dụng với các hàm khác.

- Với file đã mở, bạn có thể thực hiện các hoạt động như đọc, ghi mới, ghi thêm … trên file đó.

TẬP TIN (FILE)

- **MỞ FILE TRONG PYTHON:**

```
file object = open(file_name [, access_mode][, buffering])
```

Trong đó:

- ***filename:*** là một giá trị chuỗi chứa tên của các file mà bạn muốn truy cập.

- ***access_mode:*** xác định các chế độ của file được mở ra như read, write, append,... , đây là thông số tùy chọn và chế độ truy cập file mặc định là read (r).

TẬP TIN (FILE)

- buffering:

+ Nếu buffer được thiết lập là 0, nghĩa là sẽ không có trình đệm nào diễn ra.

+ Nếu xác định là 1, thì trình đệm dòng được thực hiện trong khi truy cập một File.

+ Nếu là số nguyên lớn hơn 1, thì hoạt động đệm được thực hiện với kích cỡ bộ đệm đã cho.

+ Nếu là số âm, thì kích cỡ bộ đệm sẽ là mặc định.

THẢO LUẬN NHÓM

NỘI DUNG:

Danh sách các chế độ (mode) khác nhau khi mở một file. Ví dụ.

THẢO LUẬN NHÓM

MODE	MÔ TẢ
‘r’	Chế độ chỉ được phép đọc.
‘r+’	Chế độ được phép đọc và ghi
‘rb’	Mở file chế độ đọc cho định dạng nhị phân. Con trỏ tại phần bắt đầu của file
‘rb+’ ‘r+b’	Mở file để đọc và ghi trong định dạng nhị phân. Con trỏ tại phần bắt đầu của file
‘w’	Mở file để ghi. Nếu file không tồn tại thì sẽ tạo mới file và ghi nội dung, nếu file đã tồn tại thì sẽ bị cắt bớt (truncate) và ghi đè lên nội dung cũ
‘w+’	Mở file để đọc và ghi. Nếu file không tồn tại thì sẽ tạo mới file và ghi nội dung, nếu file đã tồn tại thì sẽ bị cắt bớt (truncate) và ghi đè lên nội dung cũ
‘wb’	Mở file để ghi cho dạng nhị phân. Nếu file không tồn tại thì sẽ tạo mới file và ghi nội dung, nếu file đã tồn tại thì sẽ bị cắt bớt (truncate) và ghi đè lên nội dung cũ
‘wb+’ ‘w+b’	Mở file để đọc và ghi cho dạng nhị phân. Nếu file không tồn tại thì sẽ tạo mới file và ghi nội dung, nếu file đã tồn tại thì sẽ bị cắt bớt (truncate) và ghi đè lên nội dung cũ

THẢO LUẬN NHÓM

MODE	MÔ TẢ
‘a’	Mở file chế độ ghi tiếp. Nếu file đã tồn tại rồi thì nó sẽ ghi tiếp nội dung vào cuối file, nếu file không tồn tại thì tạo một file mới và ghi nội dung vào đó.
‘a+’	Mở file chế độ đọc và ghi tiếp. Nếu file đã tồn tại rồi thì nó sẽ ghi tiếp nội dung vào cuối file, nếu file không tồn tại thì tạo một file mới và ghi nội dung vào đó.
‘ab’,	Mở file chế độ ghi tiếp ở dạng nhị phân. Nếu file đã tồn tại rồi thì nó sẽ ghi tiếp nội dung vào cuối file, nếu file không tồn tại thì tạo một file mới và ghi nội dung vào đó.
‘ab+’ ‘a+b’	Mở file chế độ đọc và ghi tiếp ở dạng nhị phân. Nếu file đã tồn tại rồi thì nó sẽ ghi tiếp nội dung vào cuối file, nếu file không tồn tại thì tạo một file mới và ghi nội dung vào đó.

THẢO LUẬN NHÓM

MODE	MÔ TẢ
‘x’	Mở file chế độ ghi. Tạo file độc quyền mới (exclusive creation) và ghi nội dung, nếu file đã tồn tại thì chương trình sẽ báo lỗi
‘x+’	Mở file chế độ đọc và ghi. Tạo file độc quyền mới (exclusive creation) và ghi nội dung, nếu file đã tồn tại thì chương trình sẽ báo lỗi
‘xb’	Mở file chế độ ghi dạng nhị phân. Tạo file độc quyền mới và ghi nội dung, nếu file đã tồn tại thì chương trình sẽ báo lỗi
‘xb+’ ‘x+b’	Mở file chế độ đọc và ghi dạng nhị phân. Tạo file độc quyền mới và ghi nội dung, nếu file đã tồn tại thì chương trình sẽ báo lỗi
‘b’	Mở file ở chế độ nhị phân
‘t’	Mở file ở chế độ văn bản (mặc định)

TẬP TIN (FILE)

- **THUỘC TÍNH CỦA FILE:**

Thuộc tính	Mô tả
file.closed	Trả về True nếu file đã đóng, ngược lại là False
file.mode	Trả về chế độ truy cập của file đang được mở
file.name	Trả về tên của file

```
file = open("plc.txt", "wb")
print "Tên của file là: ", file.name
print "File có đóng hoặc không? : ", file.closed
print "Chế độ mở file : ", file.mod
```

TẬP TIN (FILE)

- **ĐÓNG FILE**:

- Khi đã thực hiện xong các hoạt động trên file thì cuối cùng chúng ta cần đóng file đó.

- Python tự động đóng một file khi đối tượng tham chiếu của một file đã được tái gán cho một file khác. Tuy nhiên, sử dụng phương thức close() để đóng một file vẫn tốt hơn.

Cú pháp:

```
fileObject.close()
```

```
# Mở file
file = open("plc.txt", "r")
# Đóng file
file.close()
```

TẬP TIN (FILE)

- **ĐỌC FILE**:

- Phương thức read

Cú pháp:

```
fileObject.read([size])
```

+ Phương thức này trả về một chuỗi có kích thước bằng size.
+ Nếu không truyền size thì toàn bộ nội dung của file sẽ được đọc.

TẬP TIN (FILE)

- **ĐỌC FILE:**

- Phương thức read

Giả sử chúng ta có một file vidu.txt với nội dung như sau:

```
Hello all!
My name's Phuc.
```

Ví dụ:

```
f = open('vidu.txt', 'r')
str= f.read()
print ('Noi dung file la:\n', str)
```

Kết quả in ra màn hình:

```
Noi dung file la:
Hello all!
My name's Phuc.
```

TẬP TIN (FILE)

- **ĐỌC FILE:**

- Phương thức readline

Cú pháp:

```
fileObject.readline()
```

+ Phương thức này cho phép đọc một dòng trong file và trả về chuỗi.

TẬP TIN (FILE)

- **ĐỌC FILE:**

- Phương thức readline

Giả sử chúng ta có một file vidu.txt với nội dung như sau:

Ví dụ:

```
f = open('vidu.txt', 'r')
line1 = f.readline()
line2 = f.readline()
print ('Dòng 1: ', line1)
print ('Dòng 2: ', line2)
```

Kết quả in ra màn hình:

```
Dòng 1:  Hello all!
Dòng 2:  My name's Phuc.
```

TẬP TIN (FILE)

- **GHI FILE**:

Tương tự đọc file, để ghi một file ta cần mở file bằng cú pháp để ghi và sử dụng phương thức write để ghi vào.

Cú pháp:

```
fileObject.write(string)
```

- Phương thức này cho phép ghi một chuỗi có nội dung là string vào vị trí của con trỏ trong file.

TẬP TIN (FILE)

- **GHI FILE:**

Ví dụ:

```
# Mở file
file = open("plc.txt", "wb")
file.write( "Python là ngôn ngữ tốt nhất");
# Đóng file
file.close()
```

Nội dung bên trong file plc.txt sau khi thực hiện ghi file:

Python là ngôn ngữ tốt nhất

TẬP TIN (FILE)

- **THAY TÊN FILE:**

Phương thức rename() trong module os được sử dụng để thay tên file. Phương thức này nhận hai tham số là tên file cũ và tên file mới.

Cú pháp:

```
os.rename("<tên file hiện tại>", "<tên file mới>")
```

Ví dụ:

```
import os
#Thay tên plc1.txt thành plc2.txt:
os.rename( "plc1.txt", "plc2.txt" )
```

TẬP TIN (FILE)

- **XÓA FILE**:

Có thể sử dụng phương thức remove() của module os để xóa các file với tham số là tên file cần xóa.

Cú pháp:

```
os.remove("<tên file>")
```

Ví dụ:

```
import os
# Xóa plc2.txt
os.remove("plc2.txt")
```

TẬP TIN (FILE)

- **VÍ TRÍ FILE:**

- Phương thức tell() sẽ cho biết vị trí hiện tại bên trong file. Nói cách khác, việc đọc và ghi tiếp theo sẽ diễn ra trên các byte đó.

- Phương thức seek(offset[, from]) thay đổi vị trí hiện tại bên trong file.

+ Tham số offset là chỉ số byte để được di chuyển.

+ Tham số from xác định vị trí tham chiếu mà từ đó byte được di chuyển.

Nếu from là 0 thì sử dụng phần đầu file như là vị trí tham chiếu.

Nếu from là 2 thì sử dụng phần cuối file như là vị trí tham chiếu.

TẬP TIN (FILE)

- **VÍ TRÍ FILE:**

Ví dụ:

```
# Mở file
file = open("plc.txt", "r+")
str = file.read(10);
print "Chuỗi đã đọc là: ", str
# Kiểm tra con trỏ hiện tại
vitri = file.tell();
print "Con trỏ hiện tại: ", vitri
```

TẬP TIN (FILE)

- **VÍ TRÍ FILE:**

```
# Đặt lại vị trí con trỏ tại vị trí đầu file
vitri = file.seek(0, 0);
str = file.read(10);
print "Chuỗi đã đọc là : ", str
# Đóng file
file.close()
```

TẬP TIN (FILE)

- **VÍ TRÍ FILE:**

Kết quả hiện thị trên màn hình như sau:

Chuỗi đã đọc là : Python là

Con trỏ hiện tại : 10

Chuỗi đã đọc là : Python là

THẢO LUẬN NHÓM

Phương thức	Mô tả
close()	Đóng một file đang mở. Nó không thực thi được nếu tập tin đã bị đóng.
fileno()	Trả về một số nguyên mô tả file (file descriptor).
flush()	Xóa sạch bộ nhớ đệm của luồng file.
isatty()	Trả về TRUE nếu file được kết nối với một thiết bị đầu cuối.
read(n)	Đọc n kí tự trong file.
readable()	Trả về TRUE nếu file có thể đọc được.
readline(n=-1)	Đọc và trả về một dòng từ file. Đọc nhiều nhất n byte/ký tự nếu được chỉ định.

THẢO LUẬN NHÓM

Phương thức	Mô tả
readlines(n=-1)	Đọc và trả về một danh sách các dòng từ file. Đọc nhiều nhất n byte/ký tự nếu được chỉ định.
seek(offset, from=SEEK_SET)	Thay đổi vị trí hiện tại bên trong file.
seekable()	Trả về TRUE nếu luồng hỗ trợ truy cập ngẫu nhiên.
tell()	Trả về vị trí hiện tại bên trong file.
truncate(size=None)	Cắt gọn kích cỡ file thành kích cỡ tham số size.
writable()	Trả về TRUE nếu file có thể ghi được.
write(s)	Ghi s kí tự vào trong file và trả về.
writelines(lines)	Ghi một danh sách các dòng và file.

Chương 7:
THAO TÁC TRÊN TẬP TIN VÀ THƯ MỤC TRONG PYTHON

NỘI DUNG GIẢNG DẠY:

7.1. Tập tin (File)

7.2. Thư mục (Directory)

THƯ MỤC (DIRECTORY)

Python cũng cung cấp rất nhiều phương thức để xử lý các hoạt động đa dạng liên quan tới thư mục. Module os được xây dựng để cung cấp các phương thức giúp tạo, xóa, và thay đổi các thư mục.

- **HIỂN THỊ THƯ MỤC HIỆN TẠI:**

- Phương thức getcwd() hiển thị thư mục đang làm việc hiện tại, trả về kết quả dưới dạng một chuỗi.

- Có thể sử dụng phương thức getcwdb() để nhận về kết quả dưới dạng byte.

THƯ MỤC (DIRECTORY)

Python cũng cung cấp rất nhiều phương thức để xử lý các hoạt động đa dạng liên quan tới thư mục. Module os được xây dựng để cung cấp các phương thức giúp tạo, xóa, và thay đổi các thư mục.

- **HIỂN THỊ THƯ MỤC HIỆN TẠI:**

- Phương thức getcwd() hiển thị thư mục đang làm việc hiện tại, trả về kết quả dưới dạng một chuỗi.

- Có thể sử dụng phương thức getcwdb() để nhận về kết quả dưới dạng byte.

```
>>> import os


>>> os.getcwd()
'C:\\Program Files\\PyScripter'


>>> os.getcwdb()
b'C:\\Program Files\\PyScripter'
```

THƯ MỤC (DIRECTORY)

- **THAY ĐỔI THƯ MỤC HIỆN TẠI**

Thư mục làm việc hiện tại có thể được thay đổi bằng phương thức chdir().

- chdir() nhận một tham số là tên thư mục muốn tới từ thư mục hiện tại.
- Có thể sử dụng cả dấu gạch chéo (/) hoặc dấu gạch chéo ngược (\) để tách các phần tử trong đường dẫn, nhưng tốt nhất vẫn nên sử dụng dấu gạch ngược (\).

```
>>> os.chdir('C:\\Python33')

>>> print(os.getcwd())
C:\Python33
```

THƯ MỤC (DIRECTORY)

- **DANH SÁCH THƯ MỤC VÀ FILE**

Có thể liệt kê tất cả các tệp và thư mục con bên trong một thư mục bằng cách sử dụng phương thức listdir().

- Phương thức này nhận một đường dẫn và trả về danh sách thư mục con và các file trong đường dẫn đó.

- Nếu không có đường dẫn nào được chỉ định, kết quả trả về sẽ truy xuất từ thư mục làm việc hiện tại.

```
>>> print(os.getcwd())
C:\Python33
>>> os.listdir()
['DLLs',
'Doc',
'include',
'Lib',
'libs',
'LICENSE.txt',
'NEWS.txt',
'python.exe',
'pythonw.exe',
'README.txt',
'Scripts',
'tcl',
'Tools']
```

THƯ MỤC (DIRECTORY)

- **TẠO MỘT THƯ MỤC MỚI**

Để tạo các thư mục mới, bạn sử dụng phương thức mkdir() của Module os.

- Có thể chọn nơi chứa thư mục mới bằng cách ghi đầy đủ đường dẫn tới nơi muốn tạo.

- Nếu đường dẫn đầy đủ không được chỉ định, thư mục mới sẽ được tạo trong thư mục làm việc hiện tại.

```
>>> os.mkdir('test')

>>> os.listdir()
['test']
```

THƯ MỤC (DIRECTORY)

- **ĐỔI TÊN THƯ MỤC HOẶC TÊN FILE**

Sử dụng phương thức rename() để đổi tên một thư mục hoặc một tập tin.

```
>>> os.listdir()
['test']

>>> os.rename('test','new_one')

>>> os.listdir()
['new_one']
```

THƯ MỤC (DIRECTORY)

- **XÓA BỎ THƯ MỤC HOẶC FILE**

- Để gỡ bỏ một file, sử dụng phương thức remove()

- Để xóa toàn bộ thư mục, sử dụng phương thức rmdir().

- Phương thức rmdir() chỉ có thể xóa các thư mục rỗng.

```
>>> os.listdir()
['new_one', 'old.txt']

>>> os.remove('old.txt')
>>> os.listdir()
['new_one']

>>> os.rmdir('new_one')
>>> os.listdir()
[]
```

THƯ MỤC (DIRECTORY)

- **XÓA BỎ THƯ MỤC HOẶC FILE**

- Để loại bỏ một thư mục không rỗng, chúng ta có thể sử dụng phương thức rmtree() bên trong module shutil.

```
>>> os.listdir()
['test']

>>> os.rmdir('test')
Traceback (most recent call last):
...
OSError: [WinError 145] The directory is not empty: 'test'

>>> import shutil

>>> shutil.rmtree('test')
>>> os.listdir()
[]
```

BÀI TẬP

NỘI DUNG:

Tạo một thư mục và các file. Sử dụng các hàm thực hiện đổi tên của file và thư mục.

BÀI TẬP

CHUẨN BỊ CHO BUỔI HỌC TIẾP THEO:

1. Đọc các tài liệu về nội dung mục 8.1; 8.2 và 8.3.
2. Tìm hiểu về cách tổ chức giao diện người dùng.