



Trường ĐH Khoa Học Tự Nhiên Tp. Hồ Chí Minh

TRUNG TÂM TIN HỌC



Lập trình Python cơ bản

Bài 11: Tập tin – Thư mục



Phòng Lập Trình - Mạng

2023

Nội dung

- 1. Thao tác với tập tin văn bản (Text File)**
- 2. Thao tác với tập tin CSV (CSV File)**
- 3. Thao tác với tập tin (File) / thư mục (Directory) qua thư viện OS**
- 4. Module os.path**



1. Thao tác với tập tin văn bản

- ❑ File là tập hợp của các thông tin được đặt tên và được lưu trữ trên bộ nhớ máy tính như đĩa cứng, đĩa mềm, CD, DVD,... Hiểu theo một cách khác thì File chính là một dãy bit có tên và được lưu trữ trên các thiết bị bộ nhớ của máy tính.
- ❑ Có 3 loại file thông dụng: văn bản, hình ảnh và âm thanh. Trong chương này chủ yếu chỉ hướng đến dạng file văn bản.
- ❑ Python cung cấp các phương thức cơ bản và cần thiết để thao tác với tập tin theo mặc định. Ta có thể thực hiện các thao tác với tập tin bằng cách sử dụng *file object*.

1. Thao tác với tập tin văn bản

□ Mở file:

- Trước khi muốn đọc hoặc ghi file, cần có thao tác mở file.
- Cú pháp: `fileObject = open(fileName [, accessMode] [, buffering] [, encoding='utf-8'])`

Trong đó:

- **fileName**: tên file sẽ truy cập (kèm đường dẫn nếu file không có trong thư mục hiện hành).
- **buffering**: gồm các giá trị:
 - 1 : có sử dụng buffer
 - 0 : không sử dụng buffer
 - >1 : buffer size
 - <0 : default size. Đây là giá trị mặc định.
- **encoding='utf-8'**: sử dụng khi mở file unicode.

1. Thao tác với tập tin văn bản

□ Mở file:

- Cú pháp: `fileObject = open(fileName [, accessMode] [, buffering] [, encoding='utf-8'])`

Trong đó:

- **accessMode**: chế độ mở tập tin: `read`, `write`, `append`,... Có một số chế độ mở file là:
 - `r` : mở để đọc nội dung (mặc định)
 - `w` : mở để ghi nội dung
 - `a` : mở để ghi thêm nội dung vào cuối file.
 - `r+` : mở để đọc và ghi. Con trỏ nằm ở đầu file.
 - `w+` : mở để đọc và ghi. Ghi đè nếu file đã tồn tại, nếu file chưa tồn tại thì tạo file mới để ghi.
 - `a+` : mở để đọc và thêm vào cuối file. Con trỏ nằm ở cuối file. Nếu file chưa tồn tại thì tạo file mới để ghi.
 - Mặc định là mở file text, nếu muốn mở file dạng nhị phân (binary) thì thêm `b`, như: `rb`, `wb`, `ab`, `rb+`, `wb+`, `ab+`.

1. Thao tác với tập tin văn bản

□ Mở file:

- Ví dụ:
`f = open('van_ban_1.txt', mode='r')`
`f = open('D:/LDS1/van_ban_1.txt', mode='r')`
- Sau khi gọi hàm **open()** thành công, kết quả sẽ trả về một **file object** hay còn gọi là “**handle**” có các thuộc tính:
 - **closed** : True nếu file đã đóng.
 - **mode** : chế độ khi mở.
 - **file name** : tên của file.
 - **softspace** : cờ đánh dấu softspace khi dùng với hàm *print*.

1. Thao tác với tập tin văn bản

❑ Đọc file: Sử dụng phương thức `read()`

- Cú pháp: `string_variable = fileObject.read([size])`
- Trong đó: **size** là số lượng byte muốn đọc, nếu bỏ qua (không truyền) tham số này thì đọc từ đầu đến cuối file.
- Lưu ý:
 - File đã được mở ở chế độ đọc.
 - Phương thức này đọc được tất cả các ký tự xuống dòng ('\n') có trong file.

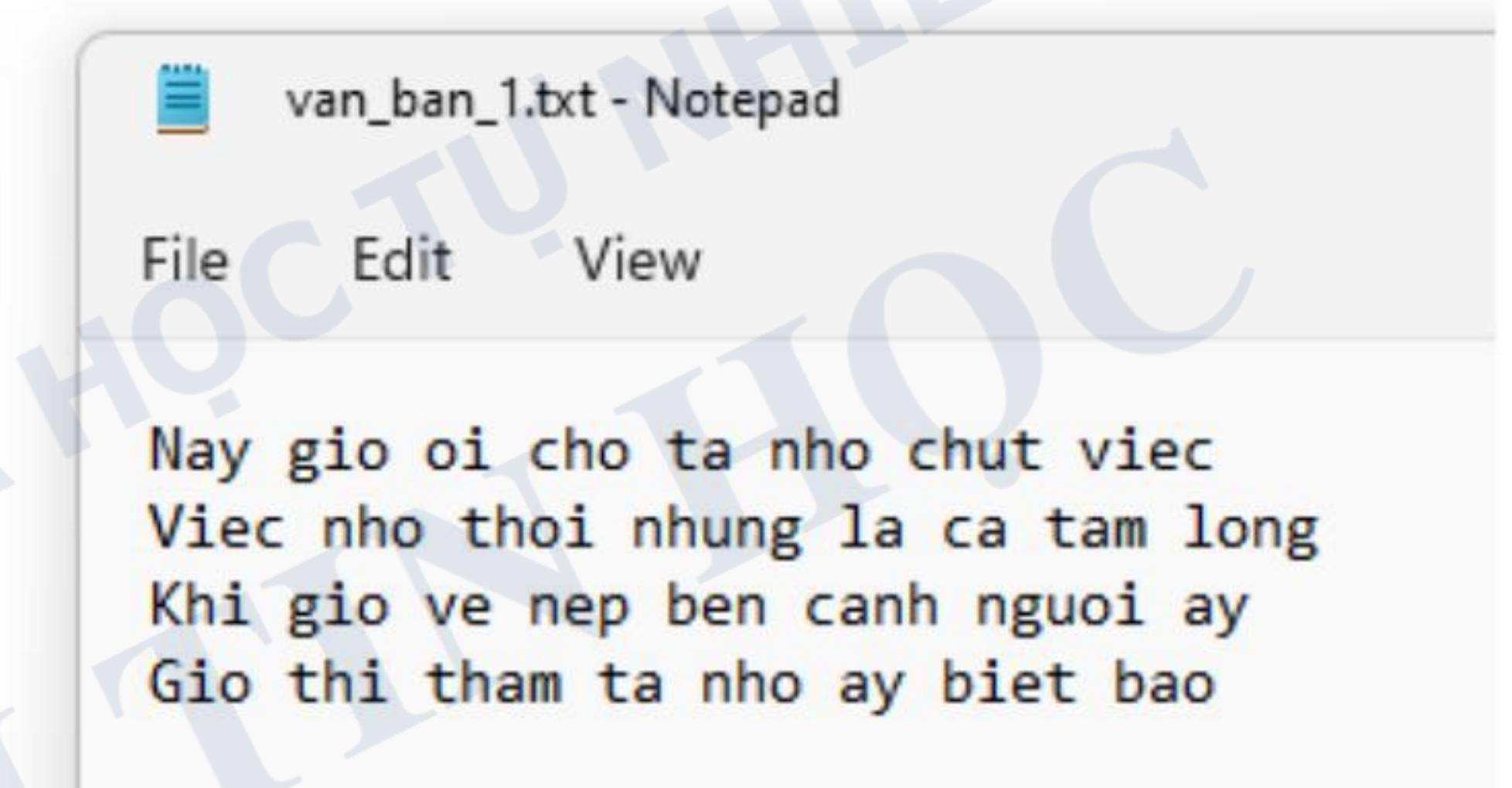
1. Thao tác với tập tin văn bản

❑ Đọc file: Sử dụng phương thức `read()`

- Cú pháp: `string_variable = fileObject.read([size])`
- Ví dụ 1: Đọc văn bản không có dấu tiếng việt

```
f = open('van_ban_1.txt', mode='r')
noi_dung = f.read()
print(noi_dung)
f.close()
```

Nay gio oi cho ta nho chut viec
Viec nho thoi nhung la ca tam long
Khi gio ve nep ben canh nguoi ay
Gio thi tham ta nho ay biet bao



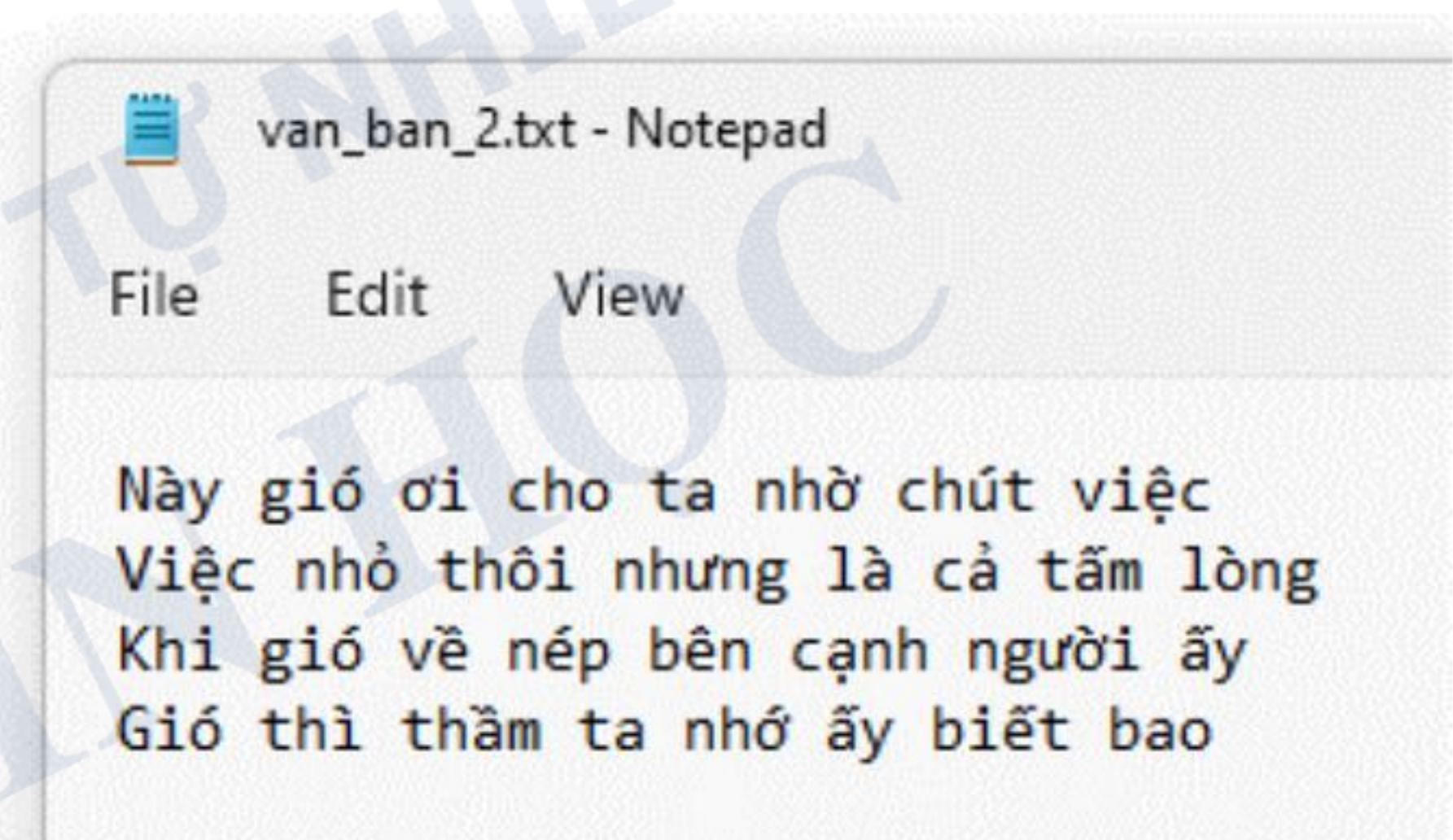
1. Thao tác với tập tin văn bản

❑ Đọc file: Sử dụng phương thức `read()`

- Cú pháp: `string_variable = fileObject.read([size])`
- Ví dụ 2: Đọc văn bản có dấu tiếng việt

```
f = open('van_ban_2.txt', mode='r', encoding='utf-8')
noi_dung = f.read()
print(noi_dung)
f.close()
```

Này gió ơi cho ta nhớ chút việc
Việc nhỏ thôi nhưng là cả tấm lòng
Khi gió về nép bên cạnh người ấy
Gió thì thăm ta nhớ ấy biết bao



1. Thao tác với tập tin văn bản

❑ Đọc file: Sử dụng phương thức `readline()` hoặc `readlines()`

- Cú pháp: `string_variable = fileObject.readline()`

```
string_variable = fileObject.readlines()
```

- Lưu ý:

- File đã được mở ở chế độ đọc.
- Phương thức `readline()`: cho phép đọc mỗi lần một dòng có trong file, trong đó ký tự xuống dòng ('\n') sẽ được đọc và ở cuối của chuỗi kết quả.
- Phương thức `readlines()`: được dùng để đọc tất cả các dòng 1 lần và sau đó trả về dưới dạng mỗi dòng là một phần tử trong list. Ký tự xuống dòng ('\n') sẽ được đọc và ở cuối của từng chuỗi kết quả.

1. Thao tác với tập tin văn bản

❑ Đọc file: Sử dụng phương thức `readline()` hoặc `readlines()`

- Cú pháp: `string_variable = fileObject.readline()`

```
string_variable = fileObject.readlines()
```

- Ví dụ 1: Sử dụng `readline()`

```
f = open('van_ban_2.txt', mode='r', encoding='utf-8')
dem = 0
while True:
    dem += 1
    dong = f.readline()
    if not dong:
        break
    print("Dòng {}: {}".format(dem, dong.strip()))
f.close()
```

Dòng 1: Này gió ơi cho ta nhờ chút việc
Dòng 2: Việc nhỏ thôi nhưng là cả tấm lòng
Dòng 3: Khi gió về nép bên cạnh người ấy
Dòng 4: Gió thì thăm ta nhớ ấy biết bao

1. Thao tác với tập tin văn bản

❑ Đọc file: Sử dụng phương thức `readline()` hoặc `readlines()`

- Cú pháp: `string_variable = fileObject.readline()`

```
string_variable = fileObject.readlines()
```

- Ví dụ 2: Sử dụng `readlines()`

```
f = open('van_ban_2.txt', mode='r', encoding='utf-8')
danh_sach_dong = f.readlines()
dem = 0
for dong in danh_sach_dong:
    dem += 1
    print("Dòng {}: {}".format(dem, dong.strip()))
f.close()
```

Dòng 1: Này gió ơi cho ta nhờ chút việc
Dòng 2: Việc nhỏ thôi nhưng là cả tấm lòng
Dòng 3: Khi gió về nép bên cạnh người ấy
Dòng 4: Gió thì thăm ta nhớ ấy biết bao

1. Thao tác với tập tin văn bản

□ **Ghi file:** Sử dụng phương thức `write()` hoặc `writelines()`

- Cú pháp: `fileObject.write(string)`

`fileObject.writelines(iterable)`

Trong đó:

- **string** (hàm `write()`): Chương trình sẽ ghi nội dung vào tập tin chỉ định. Nội dung ghi có thể là ký tự chữ, số, ký tự đặc biệt,... Hàm `write()` sẽ không bao gồm ký tự newline ('\n') ở cuối chuỗi. Giá trị trả về là `None`.
- **iterable** (hàm `writelines()`): Giá trị đầu vào của hàm này là một list các chuỗi. Cũng tương tự như hàm `write()`, hàm `writelines()` sẽ không bao gồm ký tự newline ('\n') ở cuối chuỗi.

1. Thao tác với tập tin văn bản

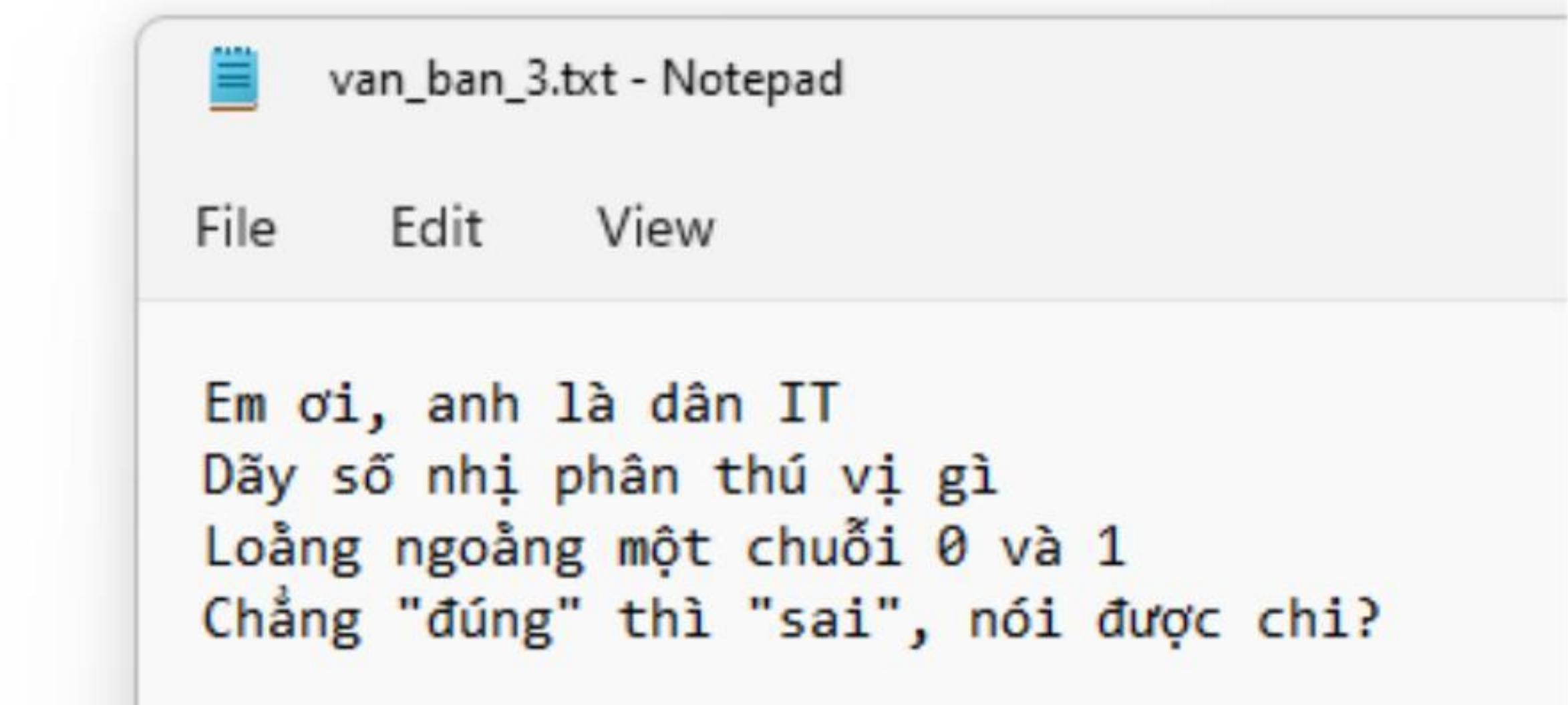
□ **Ghi file:** Sử dụng phương thức `write()` hoặc `writelines()`

- Cú pháp: `fileObject.write(string)`

`fileObject.writelines(iterable)`

- Ví dụ 1: Sử dụng hàm `write()`

```
f = open('van_ban_3.txt', mode='w', encoding='utf-8')
noi_dung = '''Em ơi, anh là dân IT
Dãy số nhị phân thú vị gì
Loằng ngoẳng một chuỗi 0 và 1
Chẳng "đúng" thì "sai", nói được chi?'''
f.write(noit_dung)
f.close()
```



1. Thao tác với tập tin văn bản

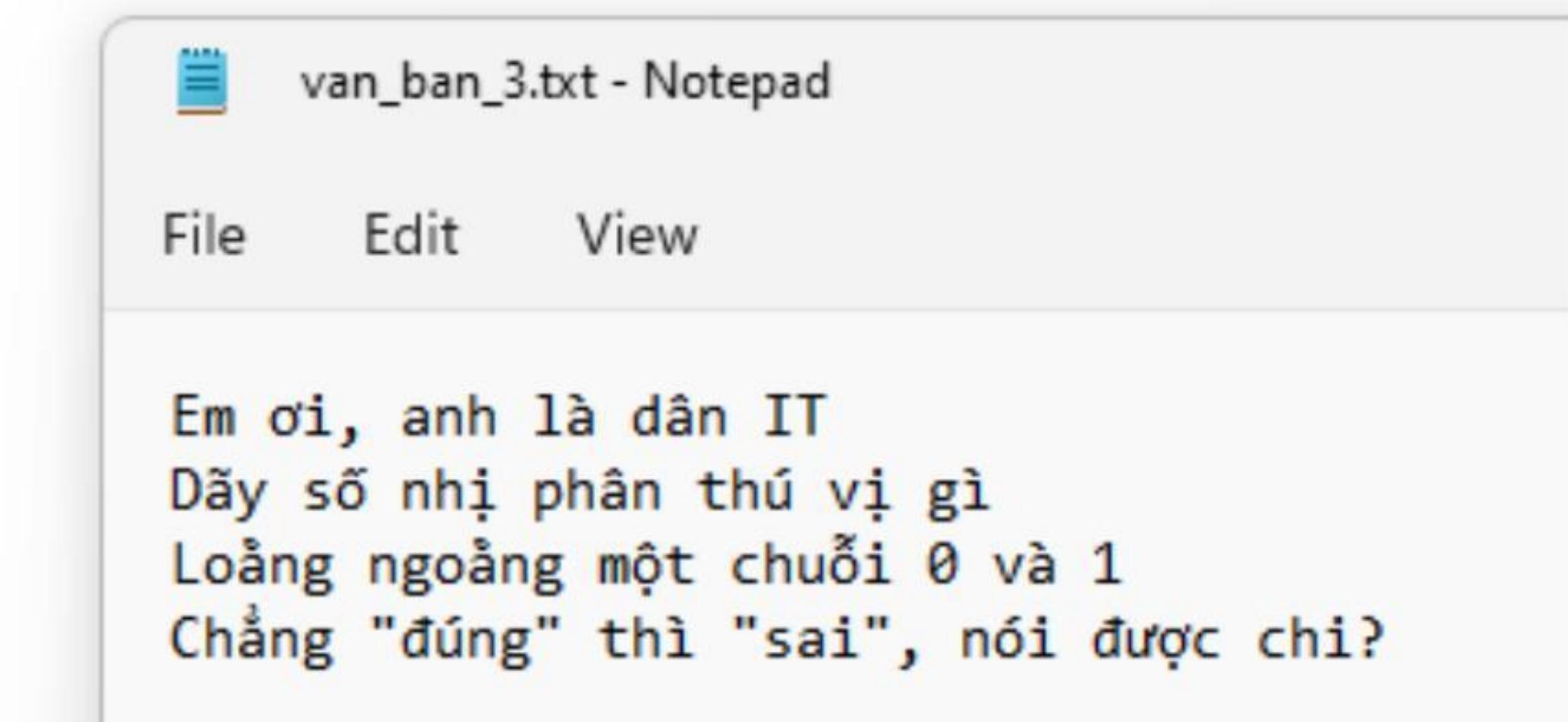
□ **Ghi file:** Sử dụng phương thức `write()` hoặc `writelines()`

- Cú pháp: `fileObject.write(string)`

`fileObject.writelines(iterable)`

- Ví dụ 2: Sử dụng hàm `writelines()`

```
f = open('van_ban_3.txt', mode='w', encoding='utf-8')
danh_sach_noi_dung = [
    'Em ơi, anh là dân IT\n',
    'Dãy số nhị phân thú vị gì\n',
    'Loằng ngoẳng một chuỗi 0 và 1\n',
    'Chẳng "đúng" thì "sai", nói được chi?'
]
f.writelines(danh_sach_noi_dung)
f.close()
```



1. Thao tác với tập tin văn bản

❑ Hàm hỗ trợ đọc và ghi file:

- **tell()**: Cho biết vị trí hiện tại trong file.
- **seek(offset[, from])**: thay đổi vị trí hiện tại của tập tin.

Trong đó:

- **offset**: khoảng cách di chuyển (tính bằng byte)
- **from**: vị trí làm mốc cho việc di chuyển, gồm các giá trị:
 - 0: đầu file
 - 1: vị trí hiện thời
 - 2: cuối file

1. Thao tác với tập tin văn bản

❑ Hàm hỗ trợ đọc và ghi file:

- Ví dụ:

```
f = open('van_ban_2.txt', mode='r', encoding='utf-8')

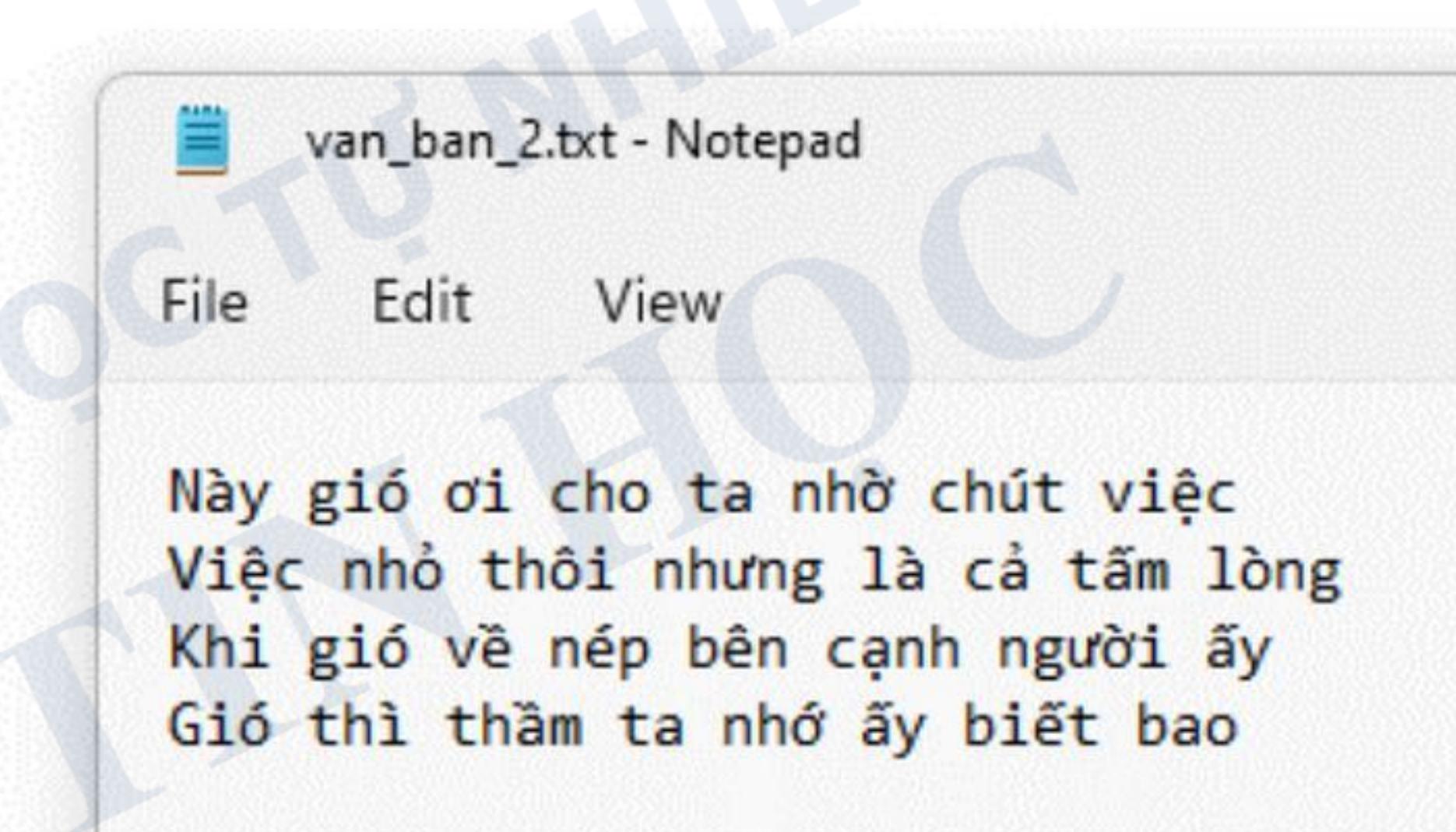
noi_dung = f.read(7)
print('Nội dung:', noi_dung)

vi_tri_hien_tai = f.tell()
print('Vị trí hiện tại:', vi_tri_hien_tai)

vi_tri_di_chuyen = f.seek(0, 0)
print('Vị trí di chuyển:', vi_tri_di_chuyen)

noi_dung = f.read(18)
print('Nội dung sau khi di chuyển:', noi_dung)

f.close()
```



Nội dung: Ngày gió
Vị trí hiện tại: 9
Vị trí di chuyển: 0
Nội dung sau khi di chuyển: Ngày gió ơi cho ta

1. Thao tác với tập tin văn bản

□ Đóng file:

- Sau khi hoàn tất các thao tác đọc ghi trên file, cần thực hiện đóng file để đảm bảo quy chế đóng mở file và giải phóng bộ nhớ cho chương trình.
- Thực hiện:
 - **Cách 1:** Sử dụng cú pháp `fileObject.close()` sau khi khởi tạo và thực hiện các thao tác với `fileObject`.

```
f = open('van_ban_2.txt', mode='r', encoding='utf-8')
danh_sach_dong = f.readlines()
dem = 0
for dong in danh_sach_dong:
    dem += 1
    print("Đòng {}: {}".format(dem, dong.strip()))
f.close()
```

1. Thao tác với tập tin văn bản

□ Đóng file:

- Sau khi hoàn tất các thao tác đọc ghi trên file, cần thực hiện đóng file để đảm bảo quy chế đóng mở file và giải phóng bộ nhớ cho chương trình.
- Thực hiện:
 - **Cách 2: Sử dụng lệnh with:** Lệnh with bảo đảm rằng file luôn luôn được đóng mà không cần biết những logic xử lý bên trong.

```
with open('van_ban_2.txt', mode='r', encoding='utf-8') as f:  
    dem = 0  
    while True:  
        dem += 1  
        dong = f.readline()  
        if not dong:  
            break  
        print("Đòng {}: {}".format(dem, dong.strip()))
```

Nội dung

1. Thao tác với tập tin văn bản (Text File)
2. Thao tác với tập tin CSV (CSV File)
3. Thao tác với tập tin (File) / thư mục (Directory) qua thư viện OS
4. Module os.path



2. Thao tác với tập tin CSV

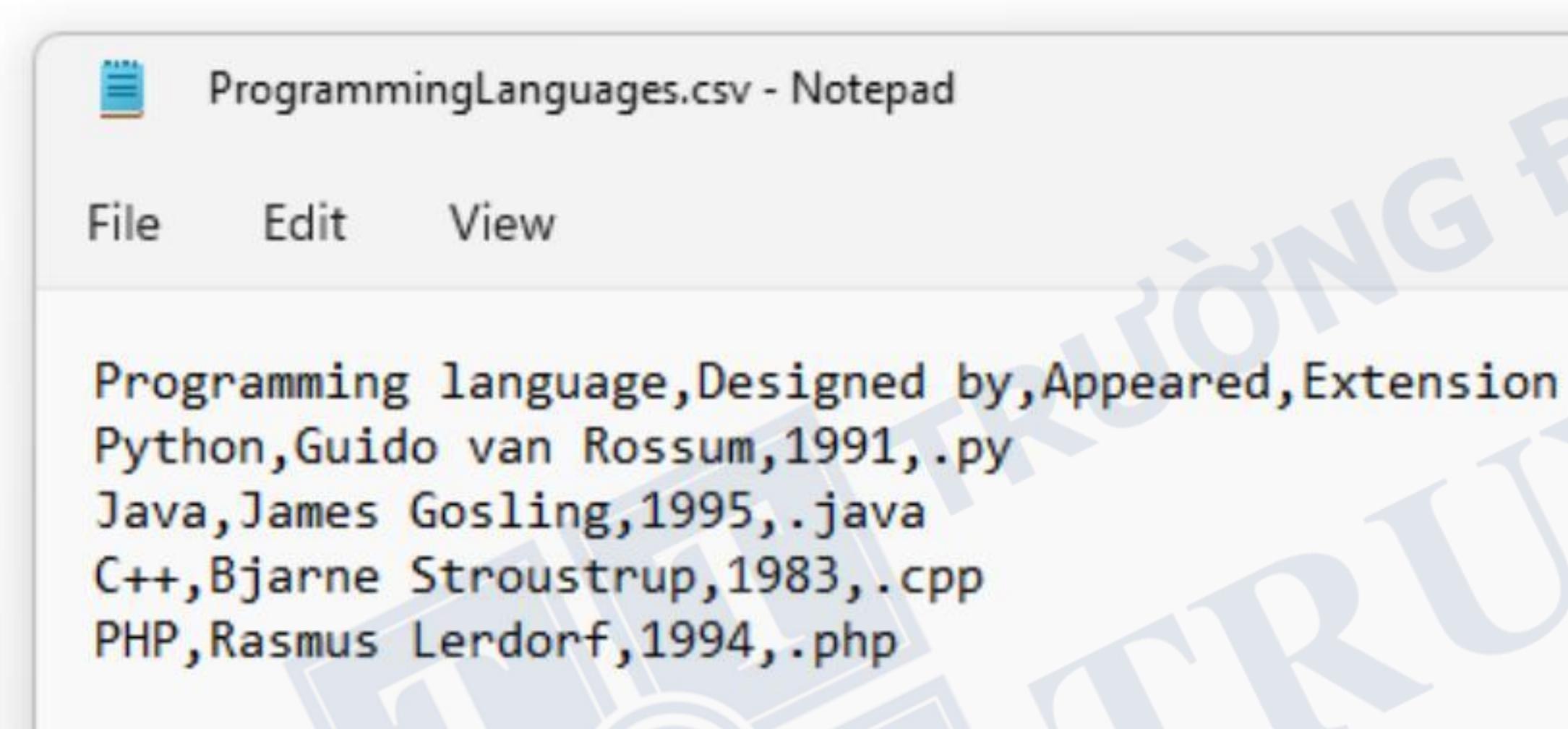
- ❑ CSV (Comma Separated Values – “giá trị được phân tách bằng dấu phẩy”) là một định dạng văn bản dành cho việc trình bày dữ liệu dạng bảng.
- ❑ File CSV thường được sử dụng để trao đổi dữ liệu giữa các ứng dụng khác nhau bằng cách xuất dữ liệu phức tạp từ một ứng dụng sang file CSV, sau đó nhập dữ liệu trong file CSV đó vào một ứng dụng khác.
- ❑ Trừ dòng đầu tiên trong file là tiêu đề các cột của bảng, tất cả các dòng còn lại mỗi dòng là một dòng dữ liệu của bảng. Các giá trị của các cột riêng lẻ được phân tách bằng ký hiệu dấu phân cách - dấu phẩy (,), dấu chấm phẩy (;) hoặc ký hiệu khác.

2. Thao tác với tập tin CSV

- ❑ Ví dụ: Để có bảng dữ liệu như sau:

	Programming language	Designed by	Appeared	Extension
	Python	Guido van Rossum	1991	.py
	Java	James Gosling	1995	.java
	C++	Bjarne Stroustrup	1983	.cpp
	PHP	Rasmus Lerdorf	1994	.php

- Sử dụng **Notepad** để tạo file:



```
Programming language,Designed by,Appeared,Extension
Python,Guido van Rossum,1991,.py
Java,James Gosling,1995,.java
C++,Bjarne Stroustrup,1983,.cpp
PHP,Rasmus Lerdorf,1994,.php
```

- Hoặc sử dụng phần mềm **MSExcel** để tạo file:

	A	B	C	D
1	Programming language	Designed by	Appeared	Extension
2	Python	Guido van Rossum	1991	.py
3	Java	James Gosling	1995	.java
4	C++	Bjarne Stroustrup	1983	.cpp
5	PHP	Rasmus Lerdorf	1994	.php

2. Thao tác với tập tin CSV

□ Đọc file với **csv.reader()**:

- Cú pháp: `csv.reader(csvfile[, delimiter=';', **fmtparams])`
- Giải thích:
 - Hàm `csv.reader()` được sử dụng để đọc nội dung file CSV theo từng dòng của file mỗi dòng gồm nội dung trên tất cả các cột theo thứ tự từ trái qua phải.
 - `csvfile`: có thể là file CSV hoặc các đối tượng dạng danh sách (string, list,...).
 - `delimiter`: ký tự được sử dụng sử dụng làm dấu phân cách giữa các cột. Mặc định là dấu phẩy (,).
 - `**fmtparams`: hằng số cho biết loại dữ liệu cần trích dẫn.

2. Thao tác với tập tin CSV

❑ Đọc file với `csv.reader()`:

- Kết quả trả về của hàm `csv.reader()` là một `csvreader object` chứa các dòng có trong `csvfile` đã cho, kiểu dữ liệu của các dòng là 1 `list`, nhờ vậy ta có thể truy cập các phần tử của `list` qua `index` (ví dụ: `listrow[index]`).
- Phương thức `csvreader.__next__()`: để bỏ qua 1 dòng nào đó trong kết quả. Kết quả sẽ tương tự khi sử dụng hàm `next(csvreaderName)`.

2. Thao tác với tập tin CSV

❑ Đọc file với `csv.reader()`:

- Ví dụ 1: Đọc tất cả các dòng trong tập tin `ProgrammingLanguages.csv`:

```
import csv
with open('ProgrammingLanguages.csv', 'r', encoding='utf-8') as f:
    csv_reader_obj = csv.reader(f)
    for row in csv_reader_obj:
        print(row)
print('Tổng số dòng: %i' % csv_reader_obj.line_num)
```

```
['Programming language', 'Designed by', 'Appeared', 'Extension']
['Python', 'Guido van Rossum', '1991', '.py']
['Java', 'James Gosling', '1995', '.java']
['C++', 'Bjarne Stroustrup', '1983', '.cpp']
['PHP', 'Rasmus Lerdorf', '1994', '.php']
Tổng số dòng: 5
```

2. Thao tác với tập tin CSV

❑ Đọc file với `csv.reader()`:

- Ví dụ 2: Đọc tất cả các dòng trong tập tin **ProgrammingLanguages.csv** ngoại trừ dòng tiêu đề (dòng đầu tiên):

```
import csv
with open('ProgrammingLanguages.csv', 'r', encoding='utf-8') as f:
    csv_reader_obj = csv.reader(f)
    csv_reader_obj.__next__() # Cách 1: Bỏ qua dòng tiêu đề
    # next(csv_reader_obj) # Cách 2: Bỏ qua dòng tiêu đề
    for row in csv_reader_obj:
        print(row)
    print('Tổng số dòng: %i' % (csv_reader_obj.line_num - 1))

['Python', 'Guido van Rossum', '1991', '.py']
['Java', 'James Gosling', '1995', '.java']
['C++', 'Bjarne Stroustrup', '1983', '.cpp']
['PHP', 'Rasmus Lerdorf', '1994', '.php']
Tổng số dòng: 4
```

2. Thao tác với tập tin CSV

❑ Đọc file với `csv.DictReader()`:

- Ví dụ: Đọc tất cả các dòng trong tập tin **ProgrammingLanguages.csv**

```
import csv
with open('ProgrammingLanguages.csv', 'r', encoding='utf-8') as f:
    csv_reader_obj = csv.DictReader(f)
    for row in csv_reader_obj:
        print(row)
```

```
{'Programming language': 'Python', 'Designed by': 'Guido van Rossum', 'Appeared': '1991', 'Extension': '.py'}
{'Programming language': 'Java', 'Designed by': 'James Gosling', 'Appeared': '1995', 'Extension': '.java'}
{'Programming language': 'C++', 'Designed by': 'Bjarne Stroustrup', 'Appeared': '1983', 'Extension': '.cpp'}
{'Programming language': 'PHP', 'Designed by': 'Rasmus Lerdorf', 'Appeared': '1994', 'Extension': '.php'}
```

2. Thao tác với tập tin CSV

□ Ghi file CSV:

- Cú pháp:
 - Tạo file object: `fileObject = open(csvfile, mode='w')`
 - Tạo csvwriter object: `csvWriterObject = csv.writer(fileObject)`
 - Ghi 1 dòng dữ liệu vào csvwriter object: `csvWriterObject.writerow(listObject)`

2. Thao tác với tập tin CSV

□ Ghi file CSV:

- Ví dụ:

```
import csv

programming_languages = [
    ['Programming language', 'Designed by', 'Appeared', 'Extension'],
    ['Python', 'Guido van Rossum', '1991', '.py'],
    ['Java', 'James Gosling', '1995', '.java'],
    ['C++', 'Bjarne Stroustrup', '1983', '.cpp'],
    ['PHP', 'Rasmus Lerdorf', '1994', '.php']
]

f = open('ProgrammingLanguages_2.csv', 'w', encoding='utf-8', newline='')
csv_writer_obj = csv.writer(f)
for pro_language in programming_languages:
    csv_writer_obj.writerow(pro_language)
f.close()
```

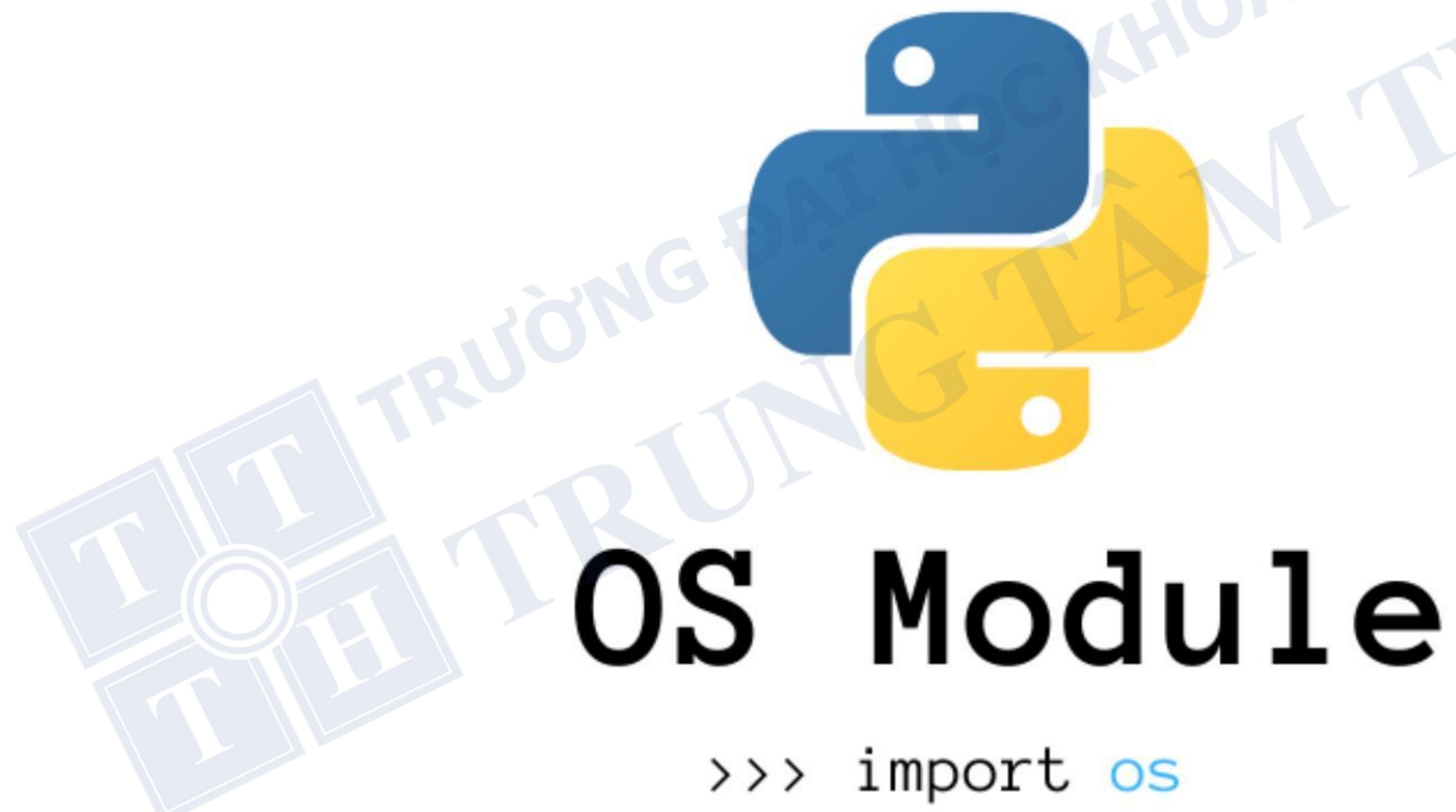
Nội dung

1. Thao tác với tập tin văn bản (Text File)
2. Thao tác với tập tin CSV (CSV File)
3. **Thao tác với tập tin (File) / thư mục (Directory) qua thư viện OS**
4. Module os.path



3. Thao tác với tập tin / thư mục qua thư viện OS

- Module **os** có nhiều phương thức hữu ích trong việc thao tác với các tập tin (file) và thư mục (directory).



3. Thao tác với tập tin / thư mục qua thư viện OS

□ Thao tác với tập tin (File): **import os**

- Đổi tên file:

- Sử dụng phương thức: **os.rename(oldFileName, newFileName)** để đổi tên một file.
 - Ví dụ: `os.rename('D:/LDS1/van_ban_1.txt', 'D:/LDS1/van_ban.txt')`

- Xóa file:

- Sử dụng phương thức: **os.remove(fileName)** để xóa một file khỏi hệ thống.
 - Ví dụ: `os.remove('D:/LDS1/van_ban.txt')`

3. Thao tác với tập tin / thư mục qua thư viện OS

□ Thao tác với thư mục (Directory): import os

Cú pháp	Chức năng
<code>os.mkdir(path)</code>	Tạo thư mục.
<code>os.makedirs(path)</code>	Tạo thư mục theo đường dẫn. Tương tự như <code>mkdir</code> nhưng nếu các thư mục con trên <code>path</code> chưa tồn tại thì phương thức này cũng tạo ra luôn.
<code>os.rmdir(path)</code>	Xóa thư mục.
<code>os.chdir(path)</code>	Thay đổi thư mục hiện hành.
<code>os.listdir(path)</code>	Lấy danh sách tập tin, thư mục.
<code>os.getcwd()</code>	Lấy đường dẫn và tên thực mục hiện hành.
<code>os.removedirs(path)</code>	Xóa một đường dẫn. Tương tự như <code>rmdir</code> , nhưng nếu xóa thành công thư mục lá, <code>removedirs</code> sẽ cố gắng xóa liên tiếp mọi thư mục cha được hiển thị trong <code>path</code> .
<code>os.basename(path)</code>	Trả về tên file sau khi loại bỏ phần mở rộng.

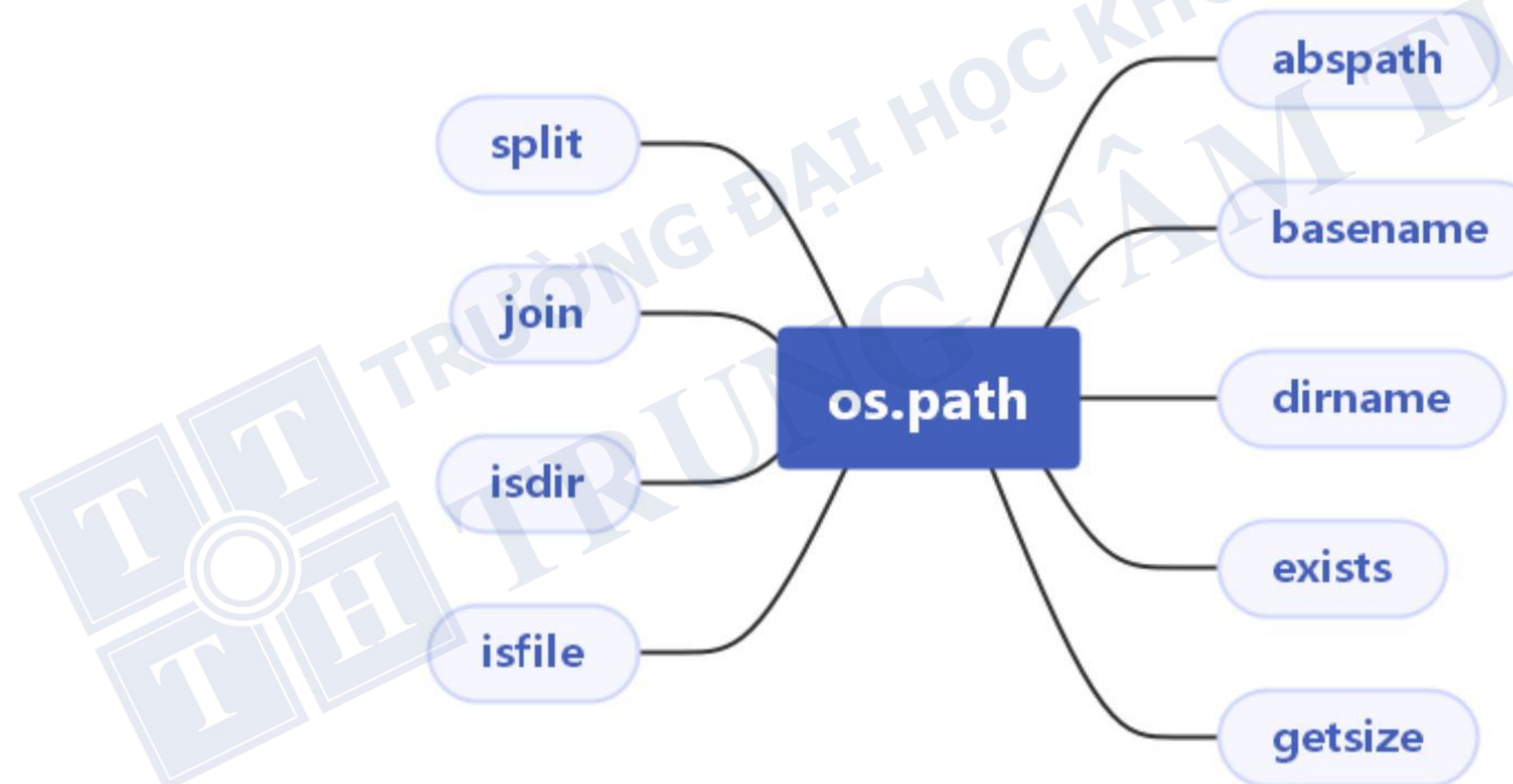
Nội dung

1. Thao tác với tập tin văn bản (Text File)
2. Thao tác với tập tin CSV (CSV File)
3. Thao tác với tập tin (File) / thư mục (Directory) qua thư viện OS
4. Module os.path



4. Module os.path

- Do các hệ điều hành khác nhau có các quy ước tên đường dẫn khác nhau, vì vậy module **os.path** hỗ trợ các phương thức giúp thao tác nhanh chóng và thuận tiện hơn trên đường dẫn.



4. Module os.path

□ Thao tác với thư mục (Directory): import os

Cú pháp	Chức năng
os.path.exists(path)	Kiểm tra đường dẫn của <i>path</i> có tồn tại hay không
os.path.isfile(path)	Kiểm tra xem path có phải là <i>tập tin</i> thông thường không?
os.path.isdir(path)	Kiểm tra xem path có phải là <i>thư mục</i> không?
os.path.dirname(path)	Trả về tên thư mục của <i>path</i> .
os.path.getctime(path)	Trả về thời gian chỉnh sửa cuối cùng của metadata trên hệ thống. Hoặc trả về thời gian tạo trên Windows.
os.path.getsize(path)	Lấy dung lượng của file theo <i>path</i> .
os.path.getatime(path)	Trả về thời gian truy cập mới nhất.
os.path.getmtime(path)	Trả về thời gian chỉnh sửa cuối cùng.
os.path.basename(path)	Tách lấy tên file ra khỏi chuỗi đường dẫn + tên file + đuôi mở rộng.



TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
TIT TRUNG TÂM TIN HỌC
TTH