SCHOOL OF
INFORMATION TECHNOLOGY
AND INNOVATION
BANGKOK UNIVERSITY

PYTHON READING & WRITING

TEXTFILE

Sirinthorn Cheyasak, Asst.Prof.

OUTLINE

- Textfile Definition
- Opening and Closing Files in Python
- Accessing python strings
- Python string operations
- Python string built-in methods/functions
- Python string example and class activity

PYTHON TEXTFILE

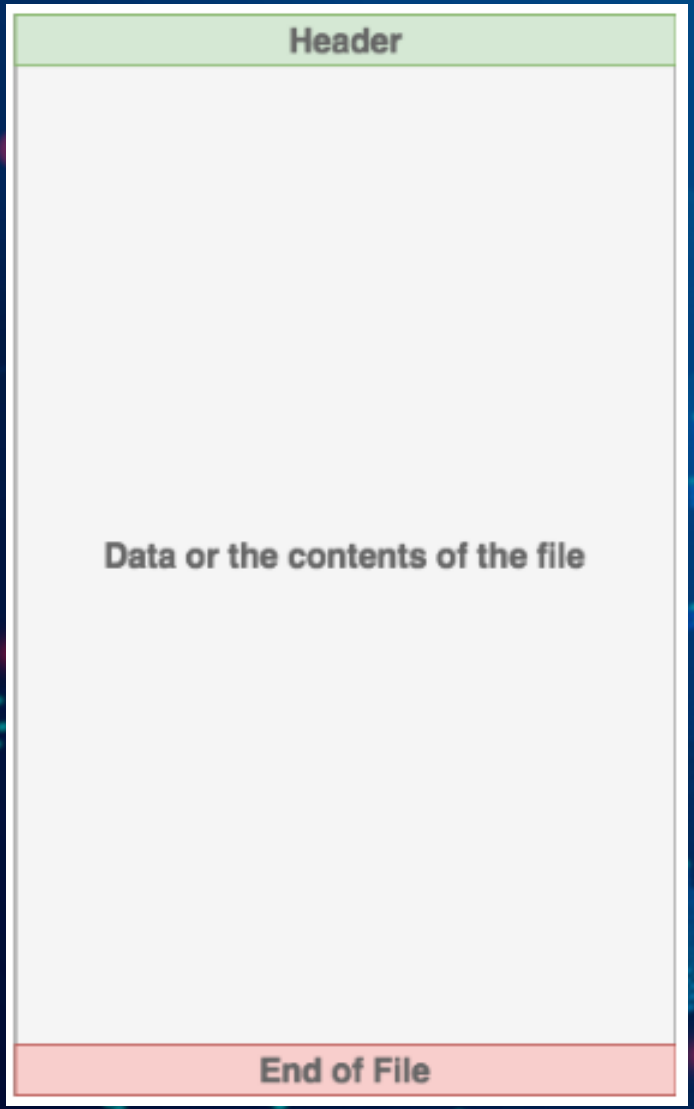
https://realpython.com/working-with-files-in-python/

TEXTFILE DEFINITION

A file is a contiguous set of bytes **used to store data**. This data is organized in a specific format and can be anything as simple as a text file or as complicated as a program executable. In the end, these byte files are then translated into binary 1 and 0 for easier processing by the computer.

https://realpython.com/working-with-files-in-python/

THREE MAIN PARTS OF MODERN FILE SYSTEM



Files on most modern file systems are composed of **three main parts**:

1. **Header:** metadata about the contents of the file (file name, size, type, and so on)
2. **Data:** contents of the file as written by the creator or editor
3. **End of file (EOF):** special character that indicates the end of the file

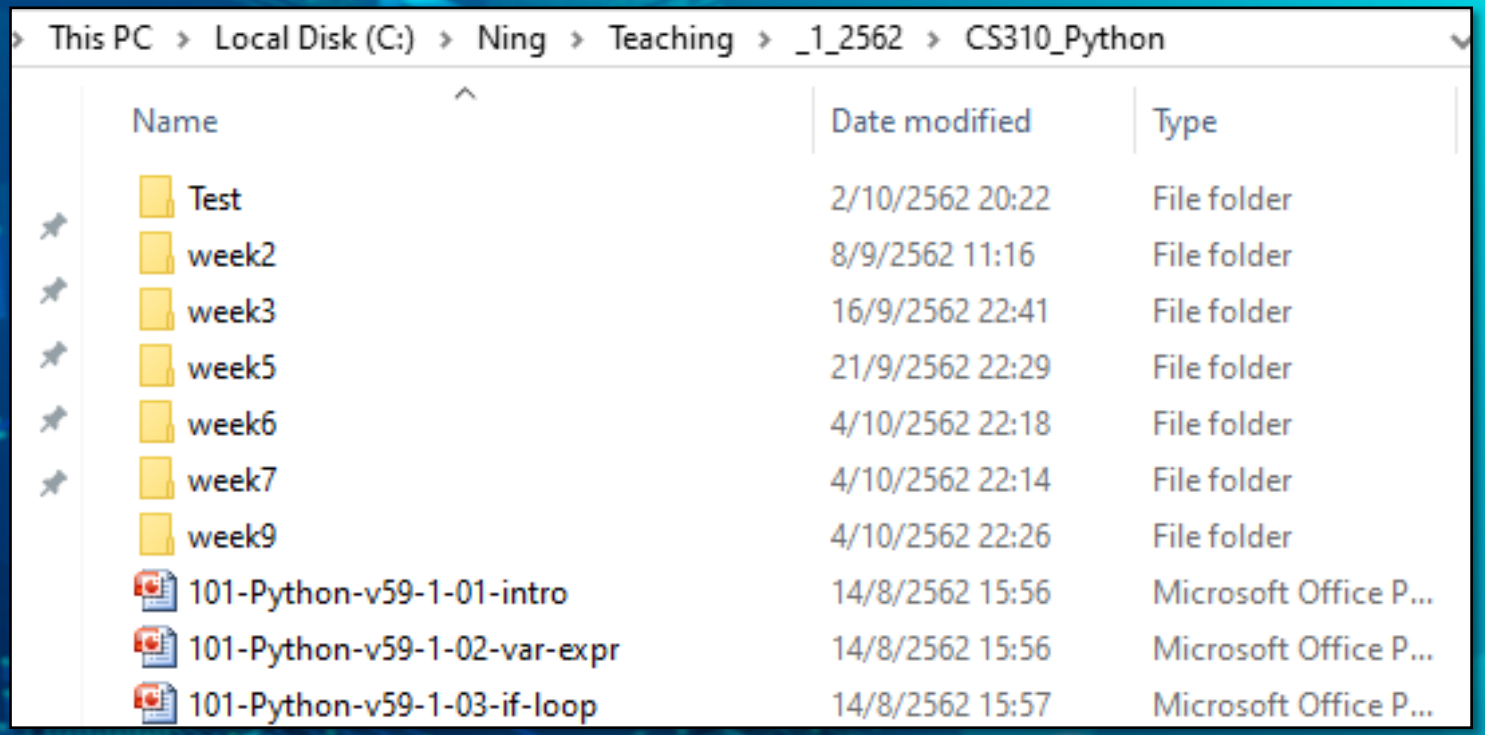
FILE PATHS

When you access a file on an operating system, a file path is required. The file path is a string that represents the location of a file. It's broken up into three major parts:

1. **Folder Path:** the file folder location on the file system where subsequent folders are separated by a forward slash / (Unix) or backslash \ (Windows)
2. **File Name:** the actual name of the file
3. **Extension:** the end of the file path pre-pended with a period (.) used to indicate the file type

FILE PATHS

```
/
│
├── path/
│   │
│   ├── to/
│   │   └── cats.gif
│   │
│   └── dog_breeds.txt
│
└── animals.csv
```



OPENING FILES IN PYTHON

OPEN FILE IN PYTON

- To open the file, use the built- in **open()** function
- Syntax:

```
file_ name = open(file_name [, access_mode][, buffering])
```

file_name คือ ตัวแปรที่ใช้เก็บข้อมูลไฟล์

open คือ ฟังก์ชันที่ใช้สำหรับขอเชื่อมต่อเพื่อเปิดใช้งาน

file_name คือ การระบุชื่อไฟล์ที่ต้องการเปิด

access_mode คือ การระบุโหมดการเข้าถึงไฟล์ default = read only

buffering คือ กำหนดขนาดหน่วยความจำชั่วคราวสำหรับพักข้อมูลก่อนประมวลผลไม่จำเป็นต้องระบุ

FILE ACCESSING MODE

โหมด	ความหมาย
r	เปิดไฟล์เพื่ออ่านอย่างเดียว (read only) ไฟล์ต้องมีอยู่จริงตามที่ระบุ
r+	เปิดไฟล์เพื่ออ่านและเขียนทับ แต่ไฟล์ต้องมีอยู่จริงตามที่ระบุ
w	เปิดไฟล์เพื่อเขียนอย่างเดียว (write only) โดยจะเขียนทับข้อมูลเดิมในไฟล์ที่มีอยู่แล้ว ถ้าไม่พบไฟล์ที่ระบุจะเปิดไฟล์ขึ้นมาใหม่
w+	เปิดไฟล์เพื่ออ่านและเขียน โดยจะเขียนทับข้อมูลเดิมในไฟล์ที่มีอยู่แล้ว ถ้าไม่พบไฟล์ที่ระบุจะเปิดไฟล์ขึ้นมาใหม่
a	เปิดไฟล์เพื่อเขียนต่อท้ายไฟล์เดิม ถ้าไม่พบไฟล์ที่ระบุจะสร้างไฟล์ขึ้นใหม่
a+	เปิดไฟล์เพื่ออ่านและเขียน ถ้าไม่พบไฟล์ที่ระบุจะสร้างไฟล์ขึ้นมาใหม่

HOW TO USE PYTHON OPEN FILE

```
f = open("test.txt")
file = open("test1.txt", "r")
infile = open("myfile.txt", "w")
```

mode r

ต้องสร้างไฟล์ข้อมูลก่อนเสมอ

- test.txt จะต้องสร้างไว้ก่อน
- test1.txt จะต้องสร้างไว้ก่อน
- myfile.txt สร้างไว้ก่อนหรือไม่ก็ได้เนื่องจากเป็น mode ของการเขียน w

mode w หรือ mode a

ไม่จำเป็นต้องสร้างไฟล์ข้อมูลก่อน

How to use Python open file

Question 1 : What is the correct of the following code for open file name lab1.txt for read?

A. Data = open(“lab1”)

B. Line = open(“lab1.txt”,”w”)

C. Infile = open(“lab1.txt”)

D. Syntax error

How to use Python open file

Question 2 : What is the correct of the following code for open file name out1.txt for write?

A. Data = open(“out1.txt”,”w”)

B. Line = open(“out1.txt”,”a”)

C. Infile = open(“out1.txt”,”r+”)

D. All of above

CLASS ACTIVITY #1
PYTHON FILE

Activity #1 : Python Texfile

1. จงเขียนคำสั่งเพื่อเปิดไฟล์ชื่อ mydata.txt สำหรับอ่านข้อมูล
2. จงเขียนคำสั่งเพื่อเปิดไฟล์ชื่อ firstdata.txt สำหรับเขียนข้อมูลแบบทับข้อมูลใหม่
3. จงเขียนคำสั่งเพื่อเปิดไฟล์ชื่อ testdata.txt สำหรับเขียนข้อมูลแบบต่อเนื่องไปเรื่อยๆ

CLOSING FILES IN PYTHON

CLOSING FILES IN PYTHON

- The **close()** method of a *file* object flushes any unwritten information and closes the file object, after which no more writing can be done.

Syntax:

```
File_name.close()
```

- file_name คือ ตัวแปรที่ใช้เก็บข้อมูลไฟล์
- close คือ เมธอดที่ใช้สำหรับการปิดไฟล์

CODE EXAMPLES

```
file1 = open("testfile.txt","r")
file1.close()
file2 = open (”studentinfo.txt","r+")
file2.close()
infile = open("datafile.txt", "w")
infile.close()
```

PYTHON OPEN FILES USING WITH

- The **with** statement, you get better syntax and exceptions handling. "The with statement simplifies exception handling by encapsulating common preparation and cleanup tasks."

Syntax: **with open**(“file-name.extension”,”mode”) as variablename **:**

- file_name คือ ชื่อข้อมูลไฟล์
- variablename คือ ชื่อตัวแปรไฟล์ที่จะใช้สำหรับเขียนโปรแกรม

PYTHON CODE EXAMPLES

```
with open("test.txt") as f :
    statements
with open("test1.txt", "r") as file :
    statements
with open("myfile.txt", "w") as infile :
    statements
```

test.txt	จำเป็นต้องสร้างไว้ก่อน
test1.txt	จำเป็นต้องสร้างไว้ก่อน
myfile.txt	สร้างไว้ก่อนหรือไม่ก็ได้เนื่องจาก เป็น mode ของการเขียน w

WITH VS OPEN TO OPEN FILE

```python
file = open("welcome.txt")
data = file.read()
print(data)
file.close()
#It's important to close the file when you're done with it
```

VS

```python
with open("welcome.text","r") as file :
    #Use file refer to the file object
    data = file.read()
    print(data)
```

HOW TO EASILY READ FILES IN PYTHON

1. Read a file using read() method
2. Read a file using readline() method
3. Read a file using readlines() method (multiple line)

1 USING READ() METHOD

- The ***read()*** method reads a string from an open file. It is important to note that Python strings can have binary data. apart from text data.

Syntax:	file_name.**read(count)**

file_name	คือ ตัวแปรที่ใช้เก็บข้อมูลไฟล์
read	คือ เมธอดที่ใช้สำหรับการอ่านข้อมูลจากไฟล์
count	คือ การระบุขนาดตัวอักษรที่จะทำการอ่าน หากไม่ระบุจะอ่านข้อมูลทุกตัวอักษรที่ปรากฏในไฟล์

1 CODE EXAMPLES

testfile.txt saved

```
1 2 3 4 5 6
7
8
9
10
```

```
file = open("testfile.txt","r")
data = file.read(10)
print(data)
file.close()
```

```
1 2 3 4 5
```

```
file = open("testfile.txt","r")
data = file.read(15)
print(data)
file.close()
```

```
1 2 3 4 5 6
7
8
```

```
file = open("testfile.txt","r")
data = file.read(-2)
print(data)
file.close()
```

```
1 2 3 4 5 6
7
8
9
10
```

1 CODE EXAMPLES

testfile.txt saved

```
1 2 3 4 5 6
7
8
9
10
```

```
file = open("testfile.txt","r")
data1 = file.read()
data2 = file.read(5)
data3 = file.read(-2)
data4 = file.read(50)
print("-"*20)
print(data1)
print("-"*20)
print(data2)
print("-"*20)
print(data3)
print("-"*20)
print(data4)
print("-"*20)
file.close()
```

```
--------------------
1 2 3 4 5 6
7
8
9
10
--------------------
--------------------
--------------------
--------------------
--------------------
```

data1

data2

data3

data4

1 CODE EXAMPLES

testfile.txt saved

```
1 2 3 4 5 6
7
8
9
10
```

```
file = open("testfile.txt","r")
data1 = file.read()
file = open("testfile.txt","r")
data2 = file.read(5)
file = open("testfile.txt","r")
data3 = file.read(-2)
file = open("testfile.txt","r")
data4 = file.read(50)
print("-"*20)
print(data1)
print("-"*20)
print(data2)
print("-"*20)
print(data3)
print("-"*20)
print(data4)
print("-"*20)
file.close()
```

```
--------------------
1 2 3 4 5 6
7
8
9
10
--------------------
1 2 3
--------------------
1 2 3 4 5 6
7
8
9
10
--------------------
1 2 3 4 5 6
7
8
9
10
--------------------
```

data1

data2

data3

data4

2 READ FILE USING READLINE()

- The ***readline()*** return one line by using the readline() method:

Syntax:	file_name.**readline( )**

- file_name คือ ตัวแปรที่ใช้เก็บข้อมูลไฟล์
- readline คือ เมธอดที่ใช้สำหรับการอ่านข้อมูลทีละบรรทัดจากไฟล์

2 CODE EXAMPLES

```
file = open("test2.txt","r")
data1 = file.readline()
data2 = file.readline()
data3 = file.readline()

print("-"*40)
print(data1)
print("-"*40)
print(data2)
print("-"*40)
print(data3)
print("-"*40)
```

test2.txt

```
1 the first line = 1
2 the second line = 2
3 the third line = 3
```

```
----------------------------------------
the first line = 1
----------------------------------------
the seconde line = 2
----------------------------------------
the third line = 3
----------------------------------------
```

data1

data2

data3

CODE EXAMPLES

```
file = open("test2.txt","r")
data1 = file.readline()
data2 = file.readline()
data3 = file.readline()

print("-"*40)
print(data1)
print("-"*40)
print(data2)
print("-"*40)
print(data3)
print("-"*40)
```

test2.txt

```
1 the first line = 1
2 the second line = 2
3 the third line = 3
```

```
----------------------------------------
the first line = 1

the seconde line = 2

the third line = 3

----------------------------------------
```

3 PYTHON READ MULTIPLE LINES

- How to write statements to read data in multiple lines

test2.txt test3.txt

```
1 the first line = 1
2 the second line = 2
3 the third line = 3
4 the forth line = 4
5 the fifth line = 5
6 the sixth line = 6
7 the seven line = 7
```

3 PYTHON READ MULTIPLE LINES

- The ***readlines()*** return one line by using the readline() method:

Syntax:	file_name.**readlines**( )

- file_name คือ ตัวแปรที่ใช้เก็บข้อมูลไฟล์
- readlines คือ เมธอดที่ใช้สำหรับการอ่านข้อมูลทีละบรรทัดมาทุกบรรทัด จากไฟล์เก็บไว้ใน buffer

3 CODE EXAMPLES

Example: Case of included \n:

test2.txt test3.txt

```
1 the first line = 1
2 the second line = 2
3 the third line = 3
4 the forth line = 4
5 the fifth line = 5
6 the sixth line = 6
7 the seven line = 7
```

```
file = open("test3.txt","r")
data = file.readlines()
for line in data :
    print(line)
```

```
the first line = 1

the second line = 2

the third line = 3

the forth line = 4

the fifth line = 5

the sixth line = 6

the seventh line = 7
```

3 CODE EXAMPLES

Example: Case of excluded the \n :

read() provide attribute **split()**

```
file = open("test3.txt","r")
data = file.read().split("\n")
for line in data :
    print(line)
```

test2.txt test3.txt

```
1 the first line = 1
2 the second line = 2
3 the third line = 3
4 the forth line = 4
5 the fifth line = 5
6 the sixth line = 6
7 the seven line = 7
```

```
the first line = 1
the second line = 2
the third line = 3
the forth line = 4
the fifth line = 5
the sixth line = 6
the seventh line = 7
```

3 CODE EXAMPLES

Example: Case of excluded the \n :

read() provide attribute **splitlines()**

```
with open("test3.txt","r") as file:
    line = file.read().splitlines()
    for data in (line) :
        print(data)
```

test2.txt test3.txt

```
1 the first line = 1
2 the second line = 2
3 the third line = 3
4 the forth line = 4
5 the fifth line = 5
6 the sixth line = 6
7 the seven line = 7
```

```
the first line = 1
the second line = 2
the third line = 3
the forth line = 4
the fifth line = 5
the sixth line = 6
the seventh line = 7
```

CLASS ACTIVITY #2
PYTHON FILE

CLASS ACTIVITY #2

Write a python program to read data from textfile name information.txt and display all of items on screen

information.txt

```
10001 student1 surname1 23
10002 student2 surname2 33
10003 student3 surname3 19
10004 student4 surname4 27
```

```
23 10001 student1 surname1
33 10002 student2 surname2
19 10003 student3 surname3
27 10004 student4 surname4
```

HOW TO ACCESS DATA ELEMENTS

How to check the student age?

information.txt

```
10001 student1 surname1 23
10002 student2 surname2 33
10003 student3 surname3 19
10004 student4 surname4 27
```

Use read().splitlines()

['student1 surname1 18', 'student2 surname2 19', 'student3 surname3 18', 'student4 surname4 20']

Split each lines into the list

['student1', 'surname1', '18']
['student2', 'surname2', '19']
['student3', 'surname3', '18']
['student4', 'surname4', '20']

Finally, you can check the student age

HOW TO ACCESS DATA ELEMENTS

How to check the student age between 18-19 ?

```
f = open(”information.txt","r")
data = f.read().splitlines()
print(data)
for line in data :
    item = line.split()
    print(item)
    if item[3] > "17" and item[3] < "20" :
        print("You are young")
```

```
['10001', ‘student1', 'surname1', '23’]
['10002', ‘student2', 'surname2', '33’]
['10003', ‘student3', 'surname3', '19’]
You are young
['10004', ‘student4', 'surname4', '27’]
```

WRITE FILE IN PYTHON

- The ***write()*** writes a single string line to file descriptor:

Syntax:	file_name.**write()**

file_name	คือ ตัวแปรที่ใช้เก็บข้อมูลไฟล์
write	คือ เมธอดที่ใช้สำหรับการเขียนข้อมูลที่เป็นสตริงลงไฟล์

Write mode :
"a" - **Append** - will append to the end of the file
"w" - **Write** - will overwrite any existing content

CODE EXAMPLES FOR WRITING

- Instruction: run this program in 5 times

```
file = open("demofile1.txt","w")
file.write("Now the file has more content.\n")
file.close()
```

demofile1.txt

Now the file has more content.

```
f = open("demofile2.txt","a")
f.write("Now the file has more content!\n")
f.close()
```

demofile2.txt

Now the file has more content!
Now the file has more content!
Now the file has more content!
Now the file has more content!
Now the file has more content!.

WRITE MULTIPLE LINE IN PYTHON

- The **writelines()** method writes the items of a list to the file. :

```
file_name.writelines(list)
```

file_name	คือ ตัวแปรที่ใช้เก็บข้อมูลไฟล์
writelines	คือ เมธอดที่ใช้สำหรับการเขียนข้อมูลที่เป็นสตริงหลายๆ บรรทัดซึ่งเก็บเป็นลิสลงไฟล์
list	คือ ลิสต์ของข้อมูลสตริงที่ต้องการเขียนลงไฟล์

"a" - Append - will append to the end of the file
"w" - Write - will overwrite any existing content

WRITING MULTIPLE LINE EXAMPLE

```
f = open("demofile3.txt", "a")
f.writelines(["Hi!\n", "Hello\n"])
f.close()
```

demofile3.txt

```
Hi!
Hello.
```

```
with open("demofile4.txt","w") as out1 :
    out1.writelines(["Hi\n","Hello\n"])
```

demofile4.txt

```
Hi!
Hello.
```

CLASS ACTIVITY #3
PYTHON FILE

CLASS ACTIVITY #3

Write python program to ask the users (3 users) to type their informations such as Name, Age, Email and create a file as name "personal_info.txt" containing all the information entered.

Example Input Data:

```
Enter Name : Sirinthorn Cheyasak
Enter Age : 27
Enter Email : sirinthorn.c@bu.ac.th
-------------------------------------------
Enter Name : Malee Sukdee
Enter Age : 22
Enter Email : malee.sukd@bumail.net
-------------------------------------------
Enter Name : Somjai Sandee
Enter Age : 19
Enter Email : somjai2566@gmail.com
-------------------------------------------
```

In Data File:

personal_info.txt

```
1  Sirinthorn Cheyasak 27 sirinthorn.c@bu.ac.th
2  Malee Sukdee 22 malee.sukd@bumail.net
3  Somjai Sandee 19 somjai2566@gmail.com
4
```

HOMEWORK OF WEEK9 PYTHON FILE

HOMEWORK: READING AND WRITING FILE FOR STUDENT RETENTION INFORMATION

e > students.txt

```
S0001 Sirinthorn Cheyasak          F 1.98
S0002 Jones Kivell                 M 1.89
S0003 Jardine Gill                 M 3.45
S0004 Sorvino Jones                F 2.56
S0005 Andrews Jardine              M 3.56
S0006 Thompson Morgan              M 2.65
S0007 Howard Parent                M 1.79
S0008 Smith Marent                 M 2.98
S0009 Yimyim Smilly                F 3.12
S0010 Sookdee Makmesuk             F 1.92
```

screen

```
Retention >>  1.98  Informaiton :  S0001 Sirinthorn Cheyasak
Retention >>  1.89  Informaiton :  S0002 Jones Kivell
Retention >>  1.79  Informaiton :  S0007 Howard Parent
Retention >>  1.92  Informaiton :  S0010 Sookdee Makmesuk
```

Output file

e > retention.txt

```
S0001 Sirinthorn Cheyasak F 1.98
S0002 Jones Kivell M 1.89
S0007 Howard Parent M 1.79
S0010 Sookdee Makmesuk F 1.92
```

THANK YOU

Any question?
contact me at sirinthorn.c@bu.ac.th
or MS Teams Chat