浙江理工大学
ZHEJIANG SCI-TECH
UNIVERSITY

9. Files

《Python programming》 / Lecturer：Zhiyi Luo (罗志一)

School of Computer Science and Technology
计算机科学与技术学院

Types of File

- **Text File:** Text file usually we use to store character data. For example, test.txt
- **Binary File**: The binary files are used to store binary data such as images, video files, audio files, etc.

File Path

- A file path define the location of a file or folder in the computer system. There are two ways to specify a file path.
  - **Absolute path**: which always begins with the root folder
  - **Relative path**: which is relative to the program's current working directory
- The absoluate path includes the complete directory list required to locate the file. For example, /user/jack/data/sales.txt is an absolute path to discover the sales.txt. All of the information needed to fine the file is contained in the path string. After the filename, the part with a period(.) is called the file's extension, and that tells us the type of file.

Absolute Path

$$\underbrace{\underbrace{\texttt{/user/jack/data/}}_{\text{Directory Path}}\underbrace{\texttt{sales.txt}}_{\text{File Name}}}_{\text{Full Path}}$$

Read File

- To read or write a file, we need to open that file. For this purpose, Python provides a built-in function open().
- Pass file path and access mode to the open(file_path, access_mode) function. It returns the file object. This object is used to read or write the file according to the access mode.
- Access mode represents the purpose of opening the file. For example, r is for reading and w is for writing.

Create a File

- We will use the sample.txt file for manipulating all file operations.
- Create a sample.txt on your machine and write the bellow content in it to get started with file handling.

```
This is a sample.txt
Line 2
Line 3
Line 4
```

Create a File

- We will use the sample.txt file for manipulating all file operations.
- Create a sample.txt on your machine and write the bellow content in it to get started with file handling.

```python
# Opening the file with absolute path
fp = open('/user/jack/data/sample.txt', 'r')
# read file
print(fp.read())
# Closing the file after reading
fp.close()
```

Create a File

- When we open a file, the operating system gives a handle to read and write a file. Once we have done using the file, it is highly recommended to close it. Because a closed file reduces the risk of being unwarrantedly modified, it will also free up the resources tied with the file.

```
# Opening the file with absolute path
fp = open('/user/jack/data/sample.txt', 'r')
# read file
print(fp.read())
# Closing the file after reading
fp.close()
```

File Access Modes

Mode	Description
r	It opens an existing file to read-only mode. The file pointer exists at the beginning.
rb	It opens the file to read-only in binary format. The file pointer exists at the beginning.
r+	It opens the file to read and write both. The file pointer exists at the beginning.
rb+	It opens the file to read and write both in binary format. The file pointer exists at the beginning of the file.
w	It opens the file to write only. It overwrites the file if previously exists or creates a new one if no file exists with the same name.
wb	It opens the file to write only in binary format. It overwrites the file if it exists previously or creates a new one if no file exists.
w+	It opens the file to write and read data. It will override existing data.
wb+	It opens the file to write and read both in binary format
a	It opens the file in the append mode. It will not override existing data. It creates a new file if no file exists with the same name.
ab	It opens the file in the append mode in binary format.
a+	It opens a file to append and read both.
ab+	It opens a file to append and read both in binary format.

Writing to a File

- To write content into a file, use the access mode w to open a file in a write mode.
- If a file already exists, it truncates the existing content and places the filehandle at the beginning of the file. A new file is create if the mentioned file doesn’t exist.
- If you want to add content at the end of the file, use the access mode a to open a file in append mode.

Move File Pointer

- The seek() method is used to change or move the file’s handle position to the specified location. The cursor defines where the data has to be read or written in the file.
- The position (index) of the first character in files is zero, just like the string index.

```python
f = open("sample.txt", "r")
# move to 11 character
f.seek(11)
# read from 11th character
print(f.read())
```

Move File Pointer

- The tell() to return the current position of the file pointer from the beginning of the file.
- The position (index) of the first character in files is zero, just like the string index.

```
f = open("sample.txt", "r")
# read first line
f.readline()
# get current position of file handle
print(f.tell())
```

File Methods

- There are various methods available that we can use with the file object.

Method	**Description**
read([size])	Returns the file content.
readline()	Read single line 逐行读取文件
readlines()	Read file into a list 读取文件中的所有行
write()	Writes the specified string to the file.
writelines()	Writes a list of strings to the file.
close()	Closes the opened file.
seek()	Set file pointer position in a file
tell()	Returns the current file location.
flush()	Flushes the internal buffer.

Copy Files

- There are several ways to copy files. The shutil.copy() method

```
import shutil
shutil.copy(source_file_path, target_file_path)
```

Delete Files

- There are several ways to copy files. The shutil.copy() method

```
import os
os.remove(file_path)
```

Working with Bytes

- A byte consists of 8 bits, and bits consist of either 0 or 1. A Byte can be interpreted in different ways like binary octal or hexadecimal. Python stores files in the form of bytes on the disk. When we open a file in text mode, that file is decoded from bytes to a string object. When we open a file in the binary mode it returns contents as bytes object without decoding.

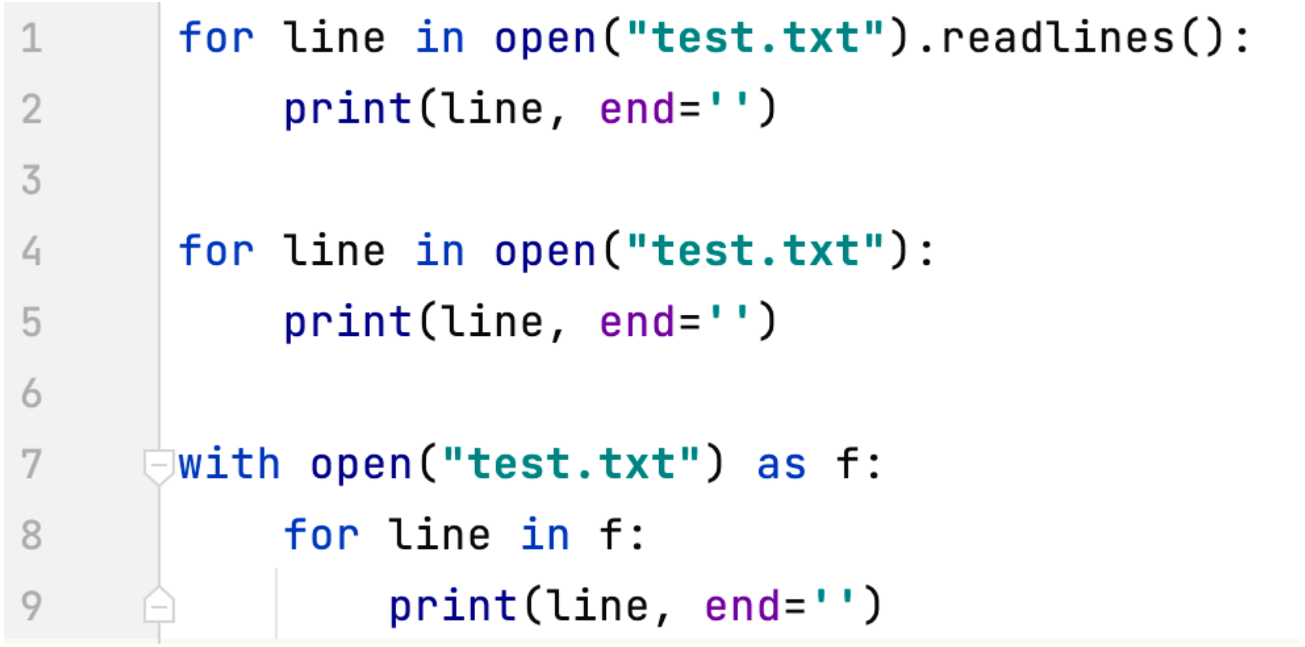
Working with Bytes

- Now let’s see the steps to write bytes to a file.
- First, open the file in binary write mode using wb.
- Second, specify contents to write in the form of bytes.
- Third, use the write() function to write byte contents to a binary file.

```python
bytes_data = b'\x21'

with open("test.txt", "wb") as fp:
    # Write bytes to file
    fp.write(bytes_data)
```

Scanning File Content



```python
for line in open("test.txt").readlines():
    print(line, end='')

for line in open("test.txt"):
    print(line, end='')

with open("test.txt") as f:
    for line in f:
        print(line, end='')
```

Exercise

- Count the number of lines in a file.

Exercise

- Define a function to tell whether a keyword exists in the given text file. If exists, return True, otherwise return False.