mindX
Technology & Startup School

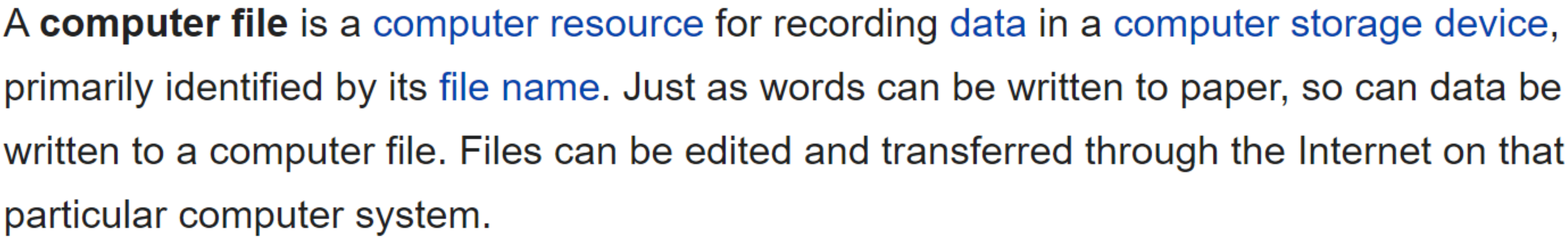
Session 12

Files

mindX
Technology & Startup School

Working with files in Python

What is a file?



A **computer file** is a computer resource for recording data in a computer storage device, primarily identified by its file name. Just as words can be written to paper, so can data be written to a computer file. Files can be edited and transferred through the Internet on that particular computer system.

WIKIPEDIA

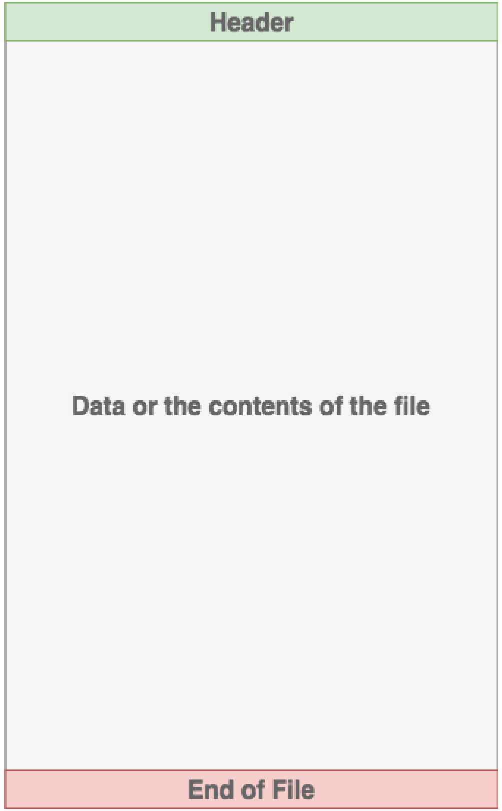
What is a file?

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

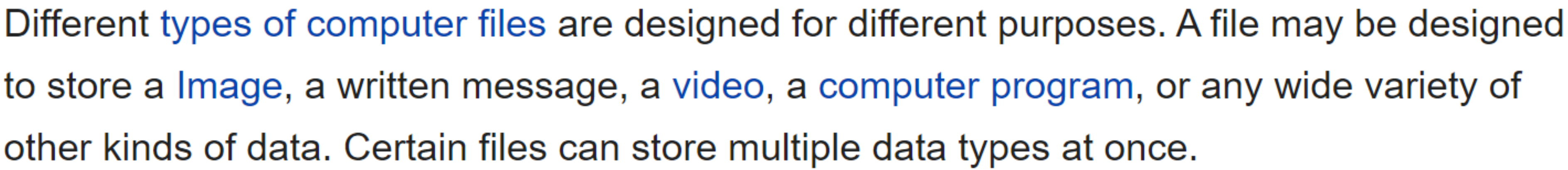


Header

Data or the contents of the file

End of File

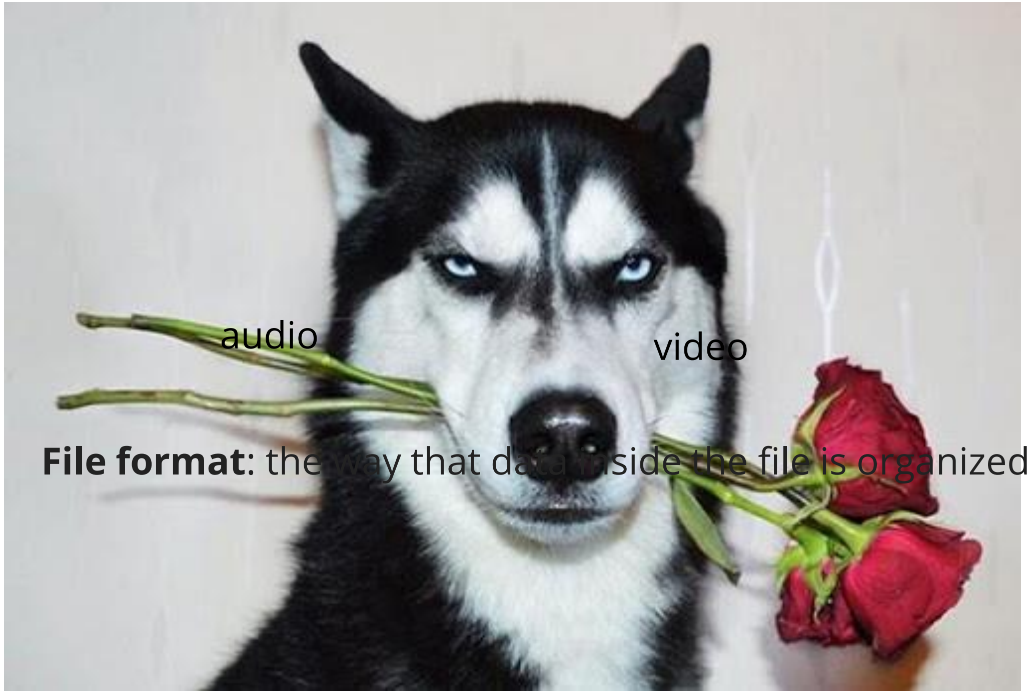
What is a file?



Different types of computer files are designed for different purposes. A file may be designed to store a Image, a written message, a video, a computer program, or any wide variety of other kinds of data. Certain files can store multiple data types at once.

WIKIPEDIA

What is a file?



image

audio

video

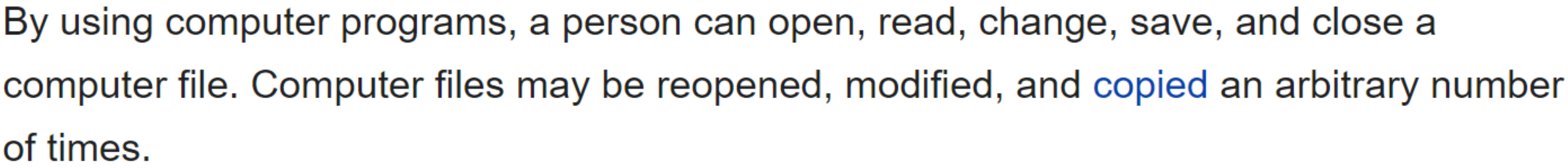
text

File format: the way that data inside the file is organized

folder

gentlemen.jpg

What is a file?



By using computer programs, a person can open, read, change, save, and close a computer file. Computer files may be reopened, modified, and copied an arbitrary number of times.

WIKIPEDIA

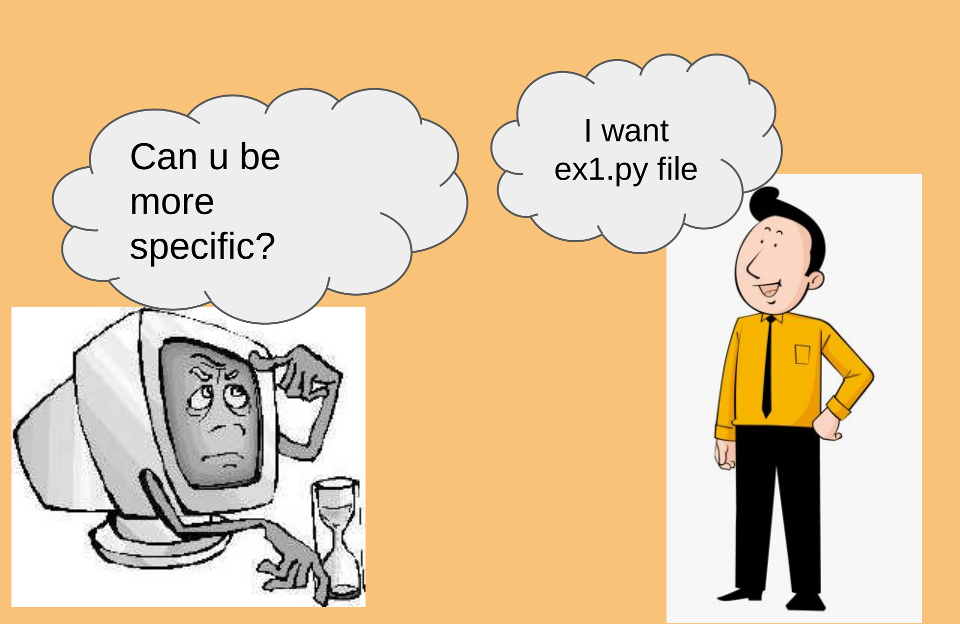
What is file path?

File path is used to access a file on an operating system.
The file path is a string that represents the location of a file. It’s broken up into 3 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 path in different OS

- **Unix**

Folder path/file_name.extention

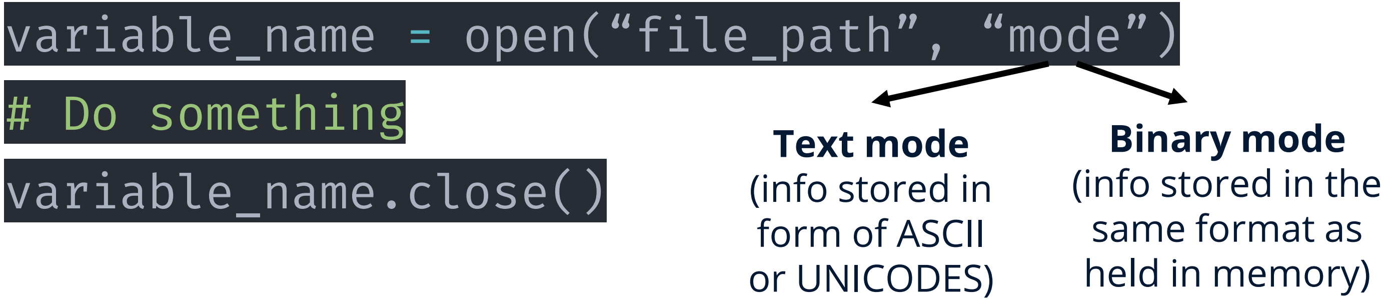
- **Window**

Folder_path\file_name.extention

/Users/mindX/Desktop/ex1.py

C:\Users\mindX\Desktop\ex1.py

Accessing file



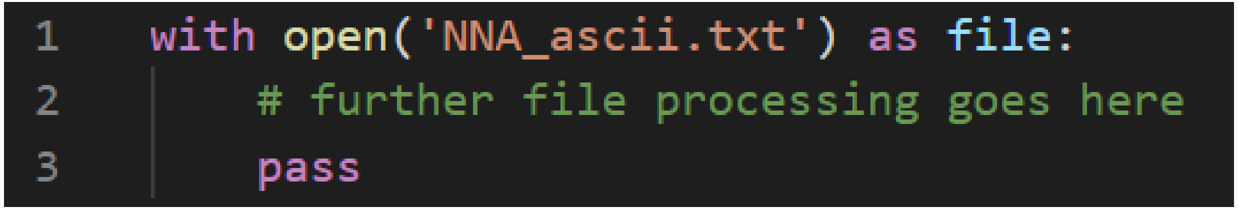
```
variable_name = open(“file_path”, “mode”)
# Do something
variable_name.close()
```

Text mode
(info stored in form of ASCII or UNICODES)

Binary mode
(info stored in the same format as held in memory)

Warning: You should *always* make sure that an open file is properly closed.

Accessing file using context manager (recommended)



```python
with open('NNA_ascii.txt') as file:
    # further file processing goes here
    pass
```

Using **with** statement

Open mode

- **“r”**: Open file to read content only. Run alert if file does not exist
- **“a”**: Open file to append more content to the file. Create a new file if file does not exist
- **“w”**: Open file to overwrite new content to the file. Create a new file if the file does not exist
- **‘x’** : Creat a new file. Run alert if file exist

Open mode

- **“.name”**: return name of opened file
- **“.mode”**: return currently mode of opened file
- **“.closed”**: return a boolean value of the state of file (is the file closed or not?)
- **‘.tell()’** : return the current position of the cursor in file
- **‘.seek(position)’**: change current seek position

# If you don’t remember the syntax, use help( )

Read mode method

‘r’ (read content)

read(n)

Read all the content

readline(n)

Read single line

readlines(n)

Store every line of the file as element in a list and then read the list

Becareful!!
The parameter ‘n’ in each method is not used with the same manner
Try it yourself to figure out.

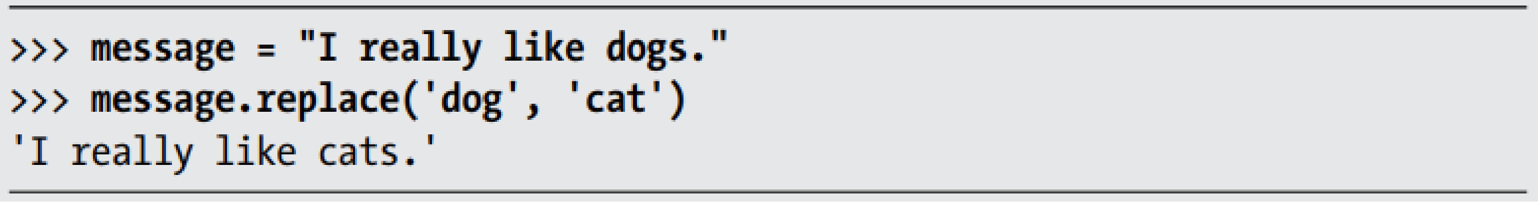
Iterating over each line in file

- Using .readline( ) method to print each line
- Using .readlines() method to create a list of lines to iterate over.
- Iterating over the file object itself.

Write a program that open file NNA_ascii.txt or any other text file. Then reads the file and prints the content inside 3 times. Print the contents once by reading in the entire file, once by looping over the file object, and once by storing the lines in a list and then working with them outside the with block.

Go to *gutenberg.org* to download the plain text version of any book (eg: Alice in Wonderland). Write a program counts number of words in the book. Update the program so it can work with multiple files at a time.

You can use the *replace( )* method to replace any word in a string with a different word. Read in each line from the file you just created and replace the word that you don’t like with any other word depends on your opinion. Print each modified line to the screen.



```
>>> message = "I really like dogs."
>>> message.replace('dog', 'cat')
'I really like cats.'
```

Write mode method

‘w’ (**overwrite** content, create a new file if it doesn’t already exist)

‘a’ (append new content at the end of the file, create a new file if it doesn’t already exist)

write()

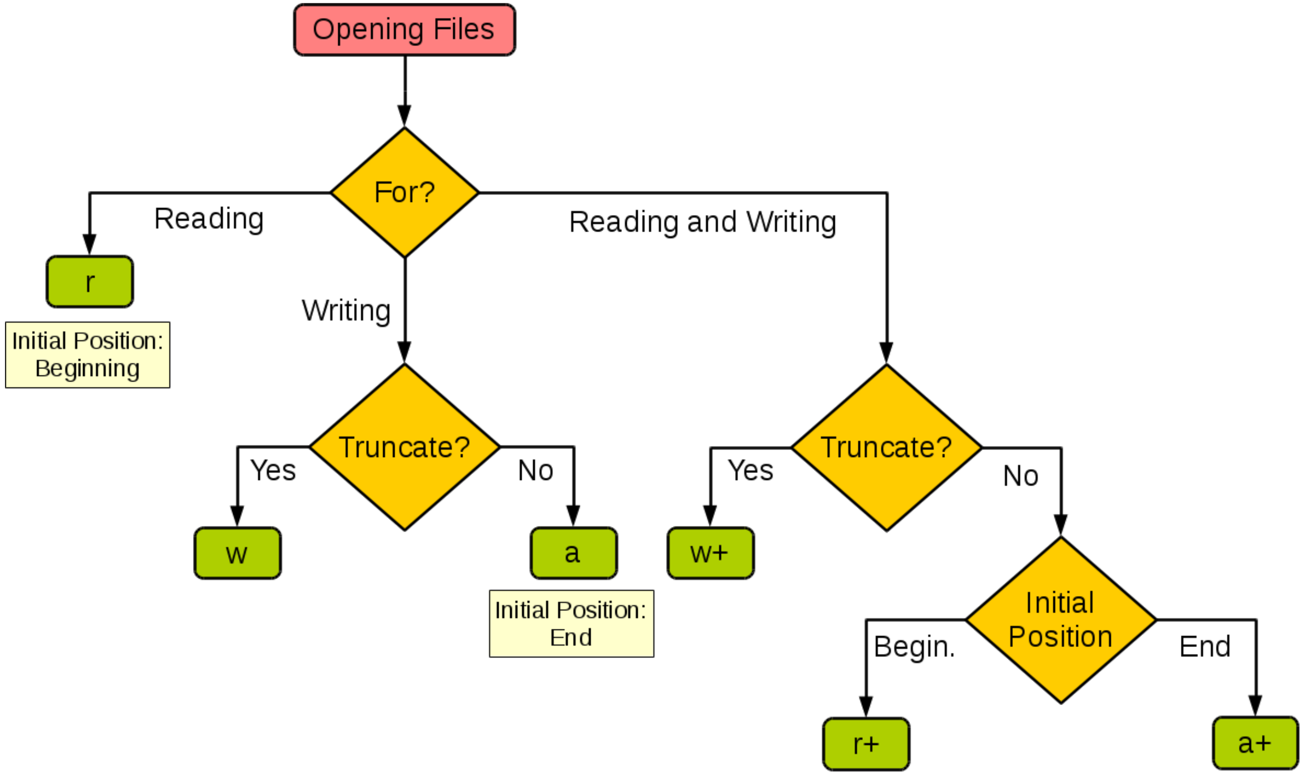
Write data to the file, only strings are allowed to be written in a text file.

writelines()

Write content of a list to the file (it DOES NOT add a newline character to the end of string)

How to create a copy of a file?

Decision branch



Delete file, folder

Import **os**

Import **shutil**

remove(file)

Delete a file

rmdir(dir)

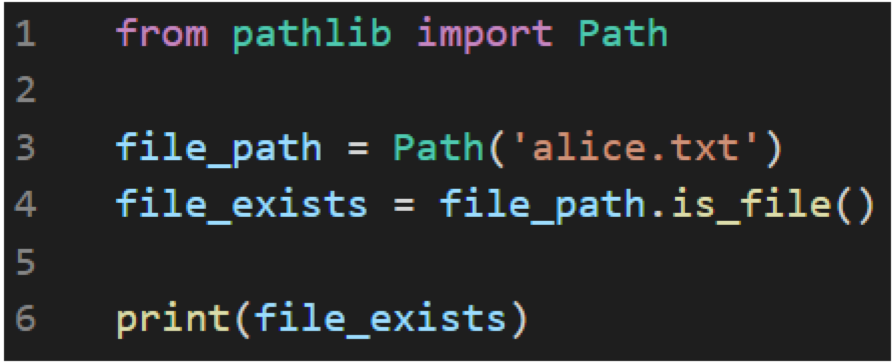
Delete an empty directory

rmtree(dir)

Remove a non-empty directory

Check if file exists

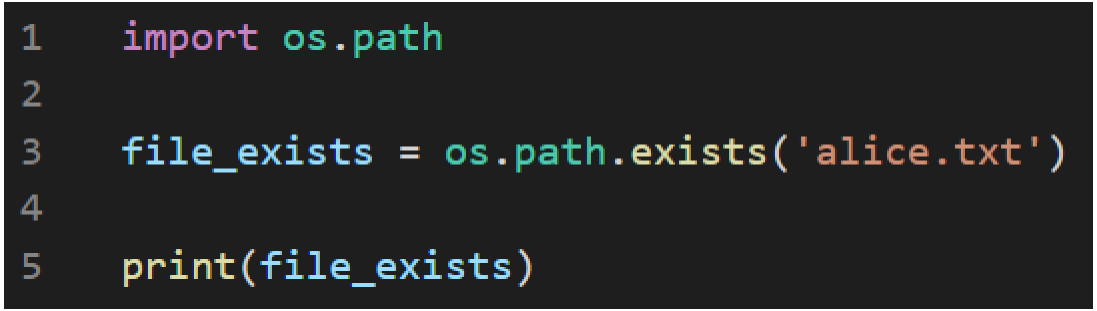
Both ways return **True** if file exists, **False** otherwise



```python
from pathlib import Path

file_path = Path('alice.txt')
file_exists = file_path.is_file()

print(file_exists)
```



```python
import os.path

file_exists = os.path.exists('alice.txt')

print(file_exists)
```

Exception handling

try - except **blocks**

any error

DivisionError exception

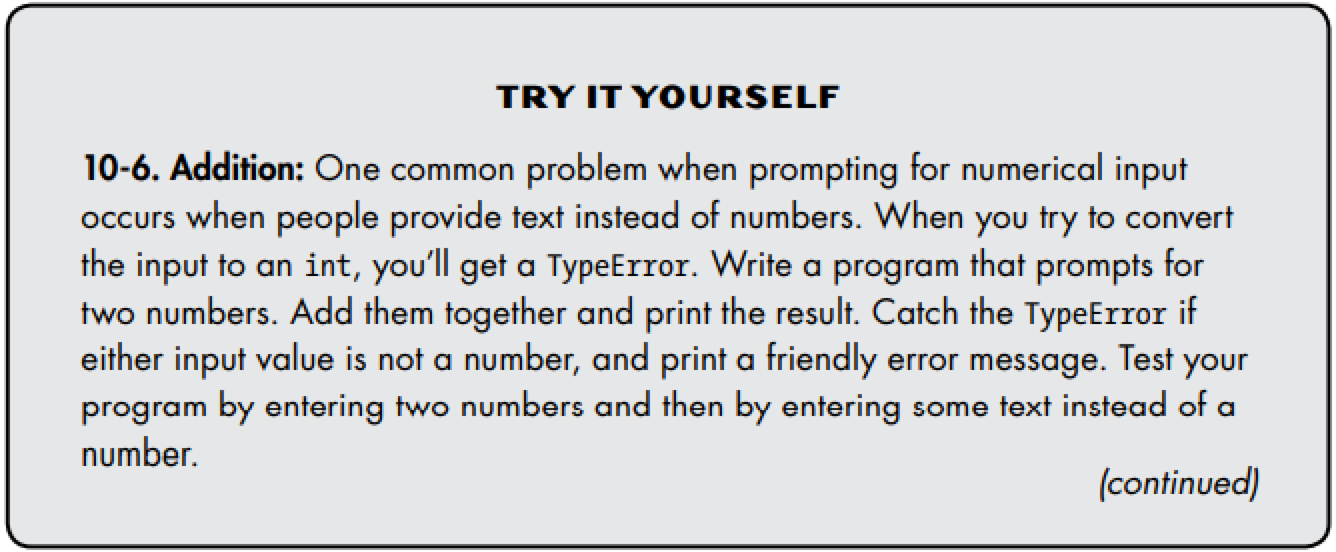
try - except – els

otFoundError exception

ilently

try - except – finally **blocks**

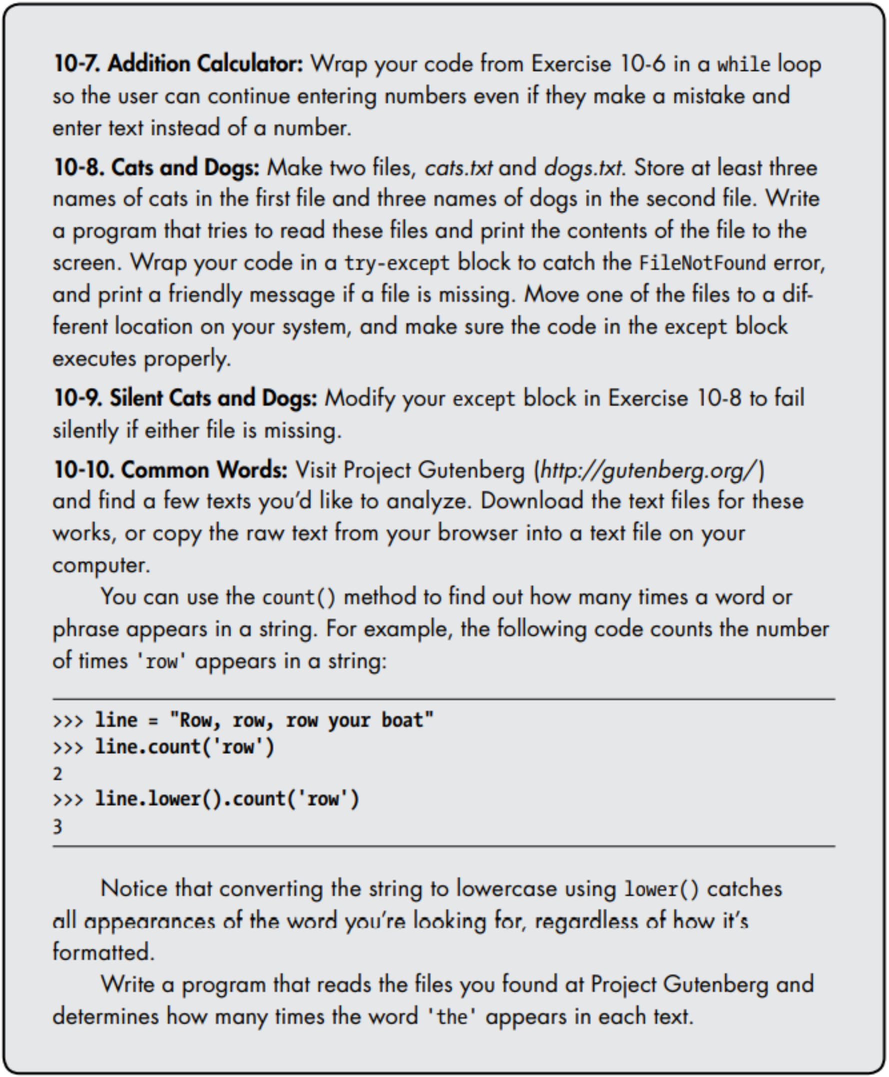
How to check if file exists using Exception handling?



TRY IT YOURSELF

10-6. Addition: One common problem when prompting for numerical input occurs when people provide text instead of numbers. When you try to convert the input to an `int`, you'll get a `TypeError`. Write a program that prompts for two numbers. Add them together and print the result. Catch the `TypeError` if either input value is not a number, and print a friendly error message. Test your program by entering two numbers and then by entering some text instead of a number.

(continued)



10-7. Addition Calculator: Wrap your code from Exercise 10-6 in a `while` loop so the user can continue entering numbers even if they make a mistake and enter text instead of a number.

10-8. Cats and Dogs: Make two files, *cats.txt* and *dogs.txt*. Store at least three names of cats in the first file and three names of dogs in the second file. Write a program that tries to read these files and print the contents of the file to the screen. Wrap your code in a `try-except` block to catch the `FileNotFound` error, and print a friendly message if a file is missing. Move one of the files to a different location on your system, and make sure the code in the `except` block executes properly.

10-9. Silent Cats and Dogs: Modify your `except` block in Exercise 10-8 to fail silently if either file is missing.

10-10. Common Words: Visit Project Gutenberg (*http://gutenberg.org/*) and find a few texts you'd like to analyze. Download the text files for these works, or copy the raw text from your browser into a text file on your computer.

You can use the `count()` method to find out how many times a word or phrase appears in a string. For example, the following code counts the number of times `'row'` appears in a string:

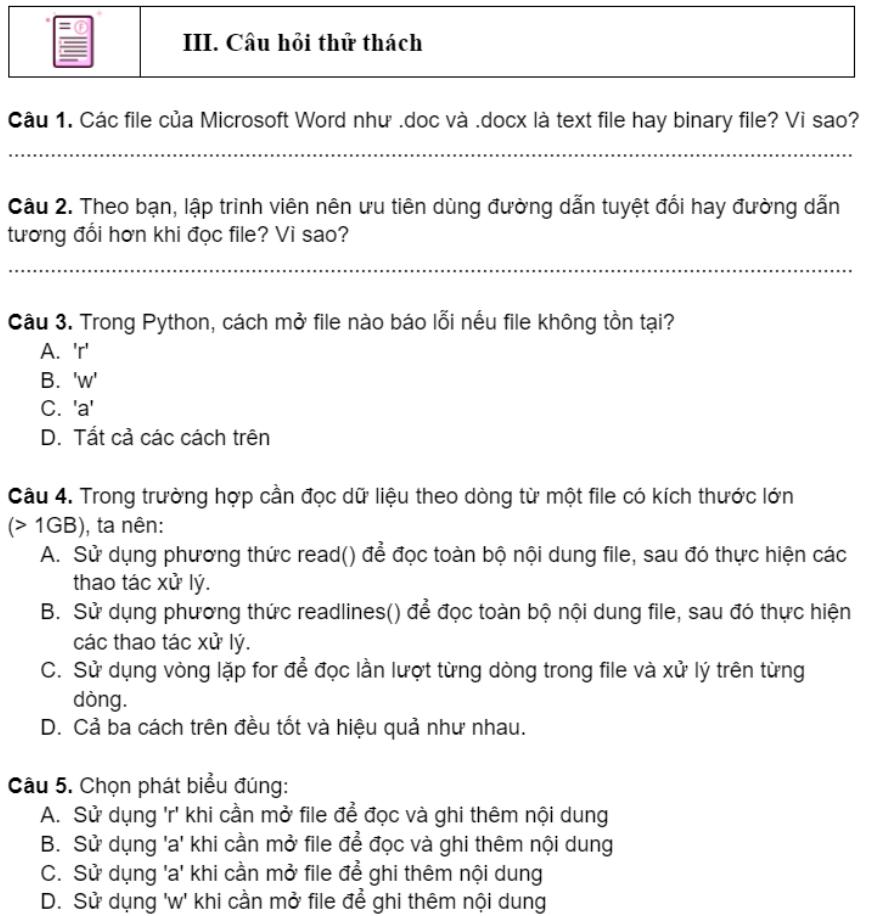
```
>>> line = "Row, row, row your boat"
>>> line.count('row')
2
>>> line.lower().count('row')
3
```

Notice that converting the string to lowercase using `lower()` catches all appearances of the word you're looking for, regardless of how it's formatted.

Write a program that reads the files you found at Project Gutenberg and determines how many times the word `'the'` appears in each text.

Challenge questions

Lab 12. Xử Lý File - Google Tài liệu



III. Câu hỏi thử thách

Câu 1. Các file của Microsoft Word như .doc và .docx là text file hay binary file? Vì sao?

....

Câu 2. Theo bạn, lập trình viên nên ưu tiên dùng đường dẫn tuyệt đối hay đường dẫn tương đối hơn khi đọc file? Vì sao?

....

Câu 3. Trong Python, cách mở file nào báo lỗi nếu file không tồn tại?

A. 'r'
B. 'w'
C. 'a'
D. Tất cả các cách trên

Câu 4. Trong trường hợp cần đọc dữ liệu theo dòng từ một file có kích thước lớn (> 1GB), ta nên:

A. Sử dụng phương thức read() để đọc toàn bộ nội dung file, sau đó thực hiện các thao tác xử lý.
B. Sử dụng phương thức readlines() để đọc toàn bộ nội dung file, sau đó thực hiện các thao tác xử lý.
C. Sử dụng vòng lặp for để đọc lần lượt từng dòng trong file và xử lý trên từng dòng.
D. Cả ba cách trên đều tốt và hiệu quả như nhau.

Câu 5. Chọn phát biểu đúng:

A. Sử dụng 'r' khi cần mở file để đọc và ghi thêm nội dung
B. Sử dụng 'a' khi cần mở file để đọc và ghi thêm nội dung
C. Sử dụng 'a' khi cần mở file để ghi thêm nội dung
D. Sử dụng 'w' khi cần mở file để ghi thêm nội dung

mindX
Technology & Startup School

THANKS!

See you in the next lesson!

CREDITS: This presentation template was created by **Slidesgo**, including icons by **Flaticon**, and infographics & images by **Freepik**.

Please keep this slide for attribution.