Working With Files in Python

UNIVERSITY of HOUSTON
DIVISION OF RESEARCH
HEWLETT PACKARD ENTERPRISE DATA SCIENCE INSTITUTE

Reading files in Python

- Python uses a file handle to manipulate files
- For example:
- ```
  fileHandle = open('myfile.txt')
  ```
- Opens the file 'myfile.txt' and returns a file handle for that file
  ```

Reading files in Python

- *#reading the content in TomSawyer file*
```
fileHandle = open('TomSawyer.txt')
content = fileHandle.read()
print(content)
fileHandle.close()
```

- *#we can also use the readlines method*
```
fileHandle = open('TomSawyer.txt')
lines = fileHandle.readlines()
for line in lines:
    print(line)
fileHandle.close()
```

Reading files in Python

- Pay attention to:
- If the file doesn’t exist or is not in the path specified python will throw an error
- Content from the file is read as String use type conversion when needed
- Remember to close the file once you’re done!

Writing to a file

- To write data to a file you need to:
  1. Open a file in writing mode:

     ```
     fileHandle = open('myfile.txt', 'w')
     ```

  2. Use the write method to send data to the file:

     ```
     fileHandle.write("Stuff needs to be stored permanently")
     ```

  3. Close file when you're done!

     ```
     fileHandle.close()
     ```

Writing to a file

- Pay attention to:
- If the file already exists, it will be rewritten, if it doesn’t it will be created
- The write method takes only strings as arguments, so when needed use type conversion
- Remember to close the file when done!

Open file options

Mode	Description	Read?	Write?	Overwrite?	Create missing file?
‘r’	Read from file	Yes	No	No	No
‘w’	Write to file	No	Yes	Yes	Yes
‘a’	Append content to file	No	Yes	No	Yes

We can also combine these modes with ‘+’ to open a file for both reading and writing

More details on writing to files

- Python uses buffering when writing to files
- This means that text is not immediately written to the file when the interpreter finds a write() statement
- Data is written to the file when a newline character ('\n') is found
- We don’t need to fiddle with the buffering process for now, just know that you can
- We can call the flush() function to force the interpreter to send all data in the buffer to the file
- Closing the file also causes the buffer to be flushed

Safe file handling: with statement

- It's a good practice to use the with statement to manipulate files in python
- it automatically closes files when we're done
- it also makes an efficient use of memory for reading large files

```
with open ('myfile_d.txt', 'w') as f:
    a,b=4,6
    f.write(str(a)+"\n")
    f.write(str(b)+"\n")
#no need to call f.close() - f closes automatically
# once the with block is exited
```

- In above example, objects **a** and **b** are only created if we can open the file for writing successfully.

Overview of Read Methods

Method Name	Use	Explanation
`write`	`filevar.write(astring)`	Add astring to the end of the file. filevar must refer to a file that has been opened for writing.
`read(n)`	`filevar.read()`	Reads and returns a string of `n` characters, or the entire file as a single string if n is not provided.
`readline(n)`	`filevar.readline()`	Returns the next line of the file with all text up to and including the newline character. If n is provided as a parameter than only n characters will be returned if the line is longer than `n`.
`readlines(n)`	`filevar.readlines()`	Returns a list of strings, each representing a single line of the file. If n is not provided then all lines of the file are returned. If n is provided then n characters are read but n is rounded up so that an entire line is returned.

Assignment

- Write a python program that will find the longest word in a file. The program should print the word and the number of characters in that word.

- The file for this exercise is posted on Materials folder. Download file ‘TomSawyer.txt’