**Module 7 –** Files

Contents

[Files 1](#_Toc205449489)

[7.1 Opening Files 1](#_Toc205449490)

[7.2 Reading Files 1](#_Toc205449491)

[7.3 Writing to Files 2](#_Toc205449492)

[7.4 Using with Statement 2](#_Toc205449493)

[7.5 Iterating Over File Lines 2](#_Toc205449494)

[7.6 Checking File Existence 2](#_Toc205449495)

[7.7 Deleting Files 2](#_Toc205449496)

[7.8 File Exception Handling 3](#_Toc205449497)

[7.9 Common File Operations 3](#_Toc205449498)

## Files

File handling in Python allows you to create, read, write, and manage files. Python provides a built-in open() function along with various methods and modes to handle files efficiently.

## 7.1 Opening Files

In Python, use the [open()](https://www.w3schools.com/python/ref_func_open.asp) function to [open a file](https://www.w3schools.com/python/python_file_open.asp):

|  |
| --- |
| file = open(“example.txt”, “r”) # Open for reading (default) |

## 7.2 Reading Files

You can [read from a file](https://www.geeksforgeeks.org/python/how-to-read-from-a-file-in-python/) using different methods:

|  |
| --- |
| file = open(“example.txt”, “r”)  print(file.read()) # Read the whole file  print(file.readline()) # Read one line  print(file.readlines()) # Read all lines as a list |

## 7.3 Writing to Files

You can [write data into a file](https://learnpython.com/blog/write-to-file-python/) using the [write()](https://www.w3schools.com/python/ref_file_write.asp) function. Writing [overwrites the file](https://www.geeksforgeeks.org/python/file-mode-in-python/) if using 'w':

|  |
| --- |
| file = open(“example.txt”, “w”)  file.write(“Hello, world!”) |

Or [appends to the existing file](https://www.geeksforgeeks.org/python/file-mode-in-python/) using 'a' mode:

|  |
| --- |
| file = open(“example.txt”, “a”)  file.write(“\nAppended line.”) |

## 7.4 Using with Statement

Best practice is to use the [with](https://www.freecodecamp.org/news/with-open-in-python-with-statement-syntax-example/) statement, which automatically closes the file:

|  |
| --- |
| with open(“example.txt”, “r”) as file:  contents = file.read()  print(contents) |

## 7.5 Iterating Over File Lines

You can [iterate](https://www.cs.toronto.edu/~guerzhoy/c4m_website_archive/level3/FilesWhile.html) [line by line using a for loop](https://www.pythonmorsels.com/reading-files-line-line/):

|  |
| --- |
| with open(“example.txt”) as file:  for line in file:  print(line.strip()) |

## 7.6 Checking File Existence

Use [os.path.exists()](https://docs.python.org/3/library/os.path.html) to check if a file exists:

|  |
| --- |
| import os  if os.path.exists(“example.txt”)  print(“File exists.”)  else:  print(“File not found.”) |

## 7.7 Deleting Files

Use [os.remove()](https://www.w3schools.com/python/python_file_remove.asp) or [os.unlink()](https://www.codecademy.com/resources/docs/python/files/unlink) to delete files:

|  |
| --- |
| import os  os.remove(“example.txt”) |

## 7.8 File Exception Handling

Always wrap file operations in [try-except](https://www.geekster.in/articles/python-file-exception/) blocks to handle errors gracefully:

|  |
| --- |
| try:  with open(“missing.txt”, “r”) as file:  print(file.read())  except FileNotFoundError:  print(“File not found.”) |

## 7.9 Common File Operations

Some of the [common methods](https://www.w3schools.com/python/python_ref_file.asp) that can be used with files in python are listed here:

| **Task** | **Method** |
| --- | --- |
| Open a file | open(filename, mode) |
| Read file content | read(), readline(), readlines() |
| Write to file | write(), writelines() |
| Use safely | with open(...) as file: |
| File existence | os.path.exists() |
| Delete file | os.remove() |
| Error handling | try-except with FileNotFoundError |
|  |  |

**Common Methods:**

* **open():** The open() function opens a file and returns the file as an object. You must specify one of the filename first, followed by the mode which can be any of the following:
  + ‘r’ – Read mode
  + ‘w’ – Write mode, creates or overwrites a file
  + ‘a’ – Append mode, adds content to the end of the file
  + ‘x’ – Create mode, creates a new file or fails if the file already exists
* **read():** The read method/s simple read data from a file. There are three variations of this method:
  + **read():** reads the entire file.
  + **readline():** reads one line at a time.
  + **readlines():** reads each line as a separate entry in a list.
* **write():** These methods write data into a file. **write()** will write a single string and **writelines()** will write multiple lines (usually from a list).
* **with open() as file:** The **with** statement safely opens and closes files. This is best practice as it ensures the file will be close after the block of code finishes, even if there is an error.
* **os.path.exists():** This simply checks whether a file exists or not within the given path. Returns True or False.
* **os.remove():** This will delete the given file, but if the file doesn’t exist it will throw a FileNotFoundError.
* **try-except:** Use try-except blocks to ensure typical errors don’t impact sequential code and display a message to the user indicating what the problem is.

Programming Exercises:

1. **Read and print a File**

Write a program that **safely** opens a text file called data.txt in read mode and displays its entire content on the screen.

1. **Write a File**

Write a program that asks the user for a line of text and writes it into a file named output.txt.

If the file already exists, it should be overwritten.

1. **Append to a File**

Write a program that asks the user to enter their name, then appends it to a file called names.txt. Each name should appear on a new line.

1. **Use the with Statement**

Rewrite Exercise #1 so that it uses the with open() statement instead of open() and close(). Verify that the file automatically closes when the block ends.

1. **Count Lines in a File**

Write a program that reads a text file (notes.txt) and counts how many lines it contains. Display the total number of lines at the end.

1. **Check if a File Exists**

Write a program that asks the user to enter a filename. Check if the file exists. If it exists, display a message saying "File found." Otherwise, display "File not found."

1. **Delete a File**

Write a program that asks the user to enter a filename. If the file exists, delete it. If not, print "File not found, cannot delete."

1. **Handle FileNotFoundError**

Write a program that tries to open a file called missing.txt for reading. If the file doesn’t exist, catch the FileNotFoundError exception and display a message such as "Error: The file does not exist."

1. **Word Counter from File**

Write a program that asks the user to enter a filename. The program should check if the file exists, then open and read its contents. If the file does not exist, create a new one and write 1-10 words (using code, not manually). Count how many words are in the file and display the total word count.

1. **File Copy and Summary**

Write a program that:

* Asks the user to enter the name of an existing file to read (e.g., source.txt).
* Checks if it exists; if not, show an error message.
* Reads all the lines using a with statement.
* Creates a new file called copy.txt and writes the same lines into it.
* After copying, print: The number of lines copied, the number of characters in total, and a message: "File copy completed successfully."

**References:**

<https://www.w3schools.com/python/python_file_handling.asp>

<https://www.geeksforgeeks.org/python/python-os-path-exists-method/>

Programming Exercise Solutions

1.

|  |
| --- |
| try:  file = open("data.txt", "r")  content = file.read()  print(content)  file.close()  except FileNotFoundError:  print("Error: data.txt not found.") |

**2.**

|  |
| --- |
| text = input("Enter a line of text: ")  with open("output.txt", "w") as file:  file.write(text)  print("Text written to output.txt") |

**3.**

|  |
| --- |
| name = input("Enter your name: ")  with open("names.txt", "a") as file:  file.write(name + "\n")  print("Name added to names.txt") |

**4.**

|  |
| --- |
| try:  with open("data.txt", "r") as file:  content = file.read()  print(content)  except FileNotFoundError:  print("Error: data.txt not found.") |

**5.**

|  |
| --- |
| try:  with open("notes.txt", "r") as file:  lines = file.readlines()  print("Total number of lines:", len(lines))  except FileNotFoundError:  print("Error: notes.txt not found.") |

**6.**

|  |
| --- |
| import os  filename = input("Enter filename: ")  if os.path.exists(filename):  print("File found.")  else:  print("File not found.") |

**7.**

|  |
| --- |
| import os  filename = input("Enter filename to delete: ")  if os.path.exists(filename):  os.remove(filename)  print("File deleted successfully.")  else:  print("File not found, cannot delete.") |

**8.**

|  |
| --- |
| try:  with open("missing.txt", "r") as file:  print(file.read())  except FileNotFoundError:  print("Error: The file does not exist.") |

**9.**

|  |
| --- |
| import os  filename = input("Enter filename: ")  if not os.path.exists(filename):  with open(filename, "w") as file:  file.write("there are seven words in my text")  with open(filename, "r") as file:  words = file.read().split()  print("Total words:", len(words)) |

**10.**

|  |
| --- |
| import os  source = input("Enter the source filename: ")  if not os.path.exists(source):  print("Error: File not found.")  else:  with open(source, "r") as src, open("copy.txt", "w") as dest:  lines = src.readlines()  dest.writelines(lines)  total\_lines = len(lines)  total\_chars = sum(len(line) for line in lines)  print(f"Lines copied: {total\_lines}")  print(f"Total characters: {total\_chars}")  print("File copy completed successfully.") |