

File Input/Output

Most data is stored in files, not input by the user every time. In this activity, you'll learn the basics of reading and writing text files.

Before starting

Create a directory name *FileIO* in your *CS149* directory. Download the two files below into this directory.

Source and data file: [write.py](#) [names.txt](#)

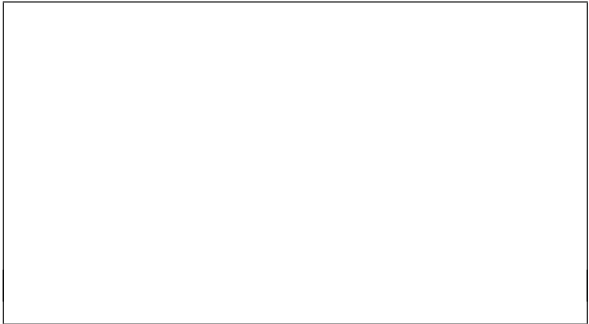
Note1: the write method returns the number of characters written. Depending on your environment, you might not see the output file contents until after calling the close or flush method.

Unit 1 Writing to a File

The following code creates a new file (in the current/default folder) named `out.txt` and writes several lines of output. Run the code, and write the contents of `out.txt` in the space provided. The code below is in [write.py](#).

```
1 outfile = open("out.txt", "w")
2 outfile.write("Example ")
3 outfile.write("output ")
4 outfile.write("text file\n")
5 outfile.write("xyz Coordinates\n")
6 outfile.write("MODEL\n")
7 outfile.write(f"ATOM {1:3d}")
8 seq = f"n {0:5.1f}{1:5.1f}{2:5.1f}"
9 outfile.write(seq)
10 outfile.write("\n")
11 outfile.close()
```

out.txt



Questions

1. Based on the Python code:

- How many arguments are passed to `open`? What are their types?
- What variable stores the *file object* returned by the `open` function?
- Identify the names of all methods used on this file object in the code.
- What type of data does the write method require for its argument?

2. Based on the `out.txt` file:

- a) How many times was the `write` method called to create the first line of text?
- b) How many times was the `write` method called to create the second line of text?
- c) What does the `"\n"` character do when writing to the file?
- d) How is the `write` method different from the `print` function?

3. Write a program that creates a file named `lines.txt` and writes 100 lines like this:

```
Line #1
Line #2
Line #3
...
```

Unit 2 Appending to a File

The second argument of `open` specifies the *mode* in which the file is opened. When writing output to a file, there are two basic modes:

- The write mode (`"w"`) will overwrite/replace the file contents.
- The append mode (`"a"`) will add new data to the end of the file.

Run the following lines in a Python Shell:

Python code	Shell output
<code>afile.write("new line\n")</code>	
<code>afile = open("out.txt", "a")</code>	
<code>afile.write("new line\n")</code>	
<code>afile.write(2.0)</code>	
<code>afile.write("2.0")</code>	
<code>afile.close()</code>	
<code>afile.write("new line\n")</code>	

Questions

4. Explain what happens as a result of the line: `afile = open("out.txt", "a")`

5. How do the arguments passed to the `open` function differ for writing a new file in comparison to appending an existing file?
6. What does the `write` method return? Run `help(afile.write)` to check your answer.
7. Explain the reason for the error observed after entering:
 - a) the first line of code: `afile.write("new line\n")`
 - b) the last line of code: `afile.write("new line\n")`
 - c) the statement: `afile.write(2.0)`

Unit 3 Reading from a File

Here is a version of `out.txt` that you should have generated from the two previous activities. Run the following lines in a Python Shell:

Python code	Shell output
<code>infile = open("out.txt", "r")</code>	
<code>infile.readline()</code>	
<code>infile.readline()</code>	
<code>infile.readlines()</code>	
<code>infile.readline()</code>	
<code>infile.close()</code>	
<code>infile = open("out.txt", "r")</code>	
<code>for line in infile:</code> <code> print(line)</code>	
<code>infile.close()</code>	
<code>infile = open("out.txt", "r")</code>	
<code>for i in range(3):</code> <code> infile.readline()</code>	
<code>line = infile.readline()</code>	
<code>infile.close()</code>	
<code>line</code>	
<code>line[0]</code>	
<code>line[0:5]</code>	
<code>words = line.split()</code>	
<code>words</code>	
<code>words[0]</code>	

Questions

8. Based on the output above:

- What type of data does the `readline` method return?
- What type of data does the `readlines` method return?

9. Why did the `readline` method return different values each time?
10. What happens if you try to read past the end of the file? Justify your answer.
11. What is the difference between the two `for` loops in Unit 3?
12. Consider the output of the first `for` loop:
- a) Why does the program display the file as if it were double spaced?
 - b) How would you change the code to avoid printing extra blank lines?
13. Based on the second half of Unit 3:
- a) Why was it necessary to open the file again?
 - b) Write code that would output 1.0 using `line`
 - c) Write code that would output 1.0 using `words`
14. Consider a file `names.txt` that contains first and last names of 100 people, with one name per line (e.g., "Anita Borg"). Write a program that prints all the last names (the second word of each line) in the file.