

# Working with Files

C-START Python PD Workshop

# Review: What Data Types Do We Know?

- Integers (e.g., 19)

# Review: What Data Types Do We Know?

- Integers (e.g., 19)
- Floating point numbers (e.g., 19.5)

# Review: What Data Types Do We Know?

- Integers (e.g., 19)
- Floating point numbers (e.g., 19.5)
- Strings (e.g., "Hello")

# Review: What Data Types Do We Know?

- Integers (e.g., 19)
- Floating point numbers (e.g., 19.5)
- Strings (e.g., "Hello")
- Booleans (True and False)

# Review: What Data Types Do We Know?

- Integers (e.g., 19)
- Floating point numbers (e.g., 19.5)
- Strings (e.g., "Hello")
- Booleans (True and False)
- Lists (e.g., [12, 13, ["Hello", "There"]])

# Review: What Data Types Do We Know?

- Integers (e.g., 19)
- Floating point numbers (e.g., 19.5)
- Strings (e.g., "Hello")
- Booleans (True and False)
- Lists (e.g., [12, 13, ["Hello", "There"]])
- Sets (e.g., {1, 2, 3})

# Review: What Data Types Do We Know?

- Integers (e.g., 19)
- Floating point numbers (e.g., 19.5)
- Strings (e.g., "Hello")
- Booleans (True and False)
- Lists (e.g., [12, 13, ["Hello", "There"]])
- Sets (e.g., {1, 2, 3})
- Dictionaries (e.g., {"Bill": 123, "Alice": 456})



# Review: What Data Types Do We Know?

- Integers (e.g., 19)
- Floating point numbers (e.g., 19.5)
- Strings (e.g., "Hello")
- Booleans (True and False)
- Lists (e.g., [12, 13, ["Hello", "There"]])
- Sets (e.g., {1, 2, 3})
- Dictionaries (e.g., {"Bill": 123, "Alice": 456})
- Functions

# The File Object

A **file object** is a data type we use to read and write to files. We can create a file object using the `open` function.

`open(filename, mode)` will open the file with name `filename` in mode `mode`, and return a file object corresponding to it.

Here is an example:

```
f = open("myfile", "r")
```

- "r" will open the file for reading

- "r" will open the file for reading
- "w" will open the file for writing, removing anything from the file that already exists

- "r" will open the file for reading
- "w" will open the file for writing, removing anything from the file that already exists
- "a" will open the file for appending, keeping the file contents and adding to the end

- `"r"` will open the file for reading
- `"w"` will open the file for writing, removing anything from the file that already exists
- `"a"` will open the file for appending, keeping the file contents and adding to the end
- `"r+"` will open for read and write

# Reading Files

Calling `f.read()` on a file object `f` will read the entire file and return a string with the contents.

**Try it:** type “This is my short story.” into a file named `story.txt` using your text editor with a short story in it, then try and read it from the interactive interpreter:

```
>>> f = open("story.txt", "r")
>>> f.read()
"This is my short story.\n"
```

# Reading Files

Calling `f.read()` on a file object `f` will read the entire file and return a string with the contents.

**Try it:** type “This is my short story.” into a file named `story.txt` using your text editor with a short story in it, then try and read it from the interactive interpreter:

```
>>> f = open("story.txt", "r")
>>> f.read()
"This is my short story.\n"
```

What happens if we call `f.read()` again?

We will be at end of file, and subsequent calls to read will return an empty string.



# Reading a File Line-by-Line

Iterating over a file object will iterate over each line.

```
f = open("story.txt", "r")
for line in f:
    words = line.split()
    for word in words:
        print(word)
```

# Writing to Files

When a file object `f` was opened in either write or append mode, `f.write(some_string)` will write the string `some_string` to the file.

**Try it:** Write another story, but this time from the interactive interpreter.

```
>>> f = open("story2.txt", "w")
>>> f.write("There once was a teacher...\n")
```

# Closing Files

When you are finished with a file, you should close it with `f.close()`. Doing so will free up system resources and allow other processes on your system to work with that file.

```
f = open("my_file.txt", "r")
contents = f.read()
f.close()           # don't forget to close
```

# Saving Phonebook Program

Now that you have the knowledge of File I/O, you can create a phone book program which saves and loads from a file. Try to modify our original program, and if you need help, the solution is on the workshop website.

# Don't forget the documentation!

Python also includes a data type for sets. A set is an unordered collection of unique elements, and eliminating duplicate entries. Set objects also support mathematical operations like union and difference.

The *Input and Output* page in the official Python documentation has excellent information and examples on using files.

These slides are nowhere near complete! Go forth and read the docs!

```
1 # Create a set with unique elements
2 s = {'orange', 'banana', 'apple', 'apple'}
3
4 # Test membership testing
5 'orange' in s
6 True
7
8 'mango' in s
9 False
10
11 # Demonstrate set operations on unique letters from two words
```