Working with Files

Camp Inc. Coding Track

Summer 2016

What Data Types Do We Know?

- Integers (eg. `19`)

What Data Types Do We Know?

- Integers (eg. `19`)
- Floating point numbers (eg. `19.5`)

What Data Types Do We Know?

- Integers (eg. `19`)
- Floating point numbers (eg. `19.5`)
- Strings (eg. `"Hello"`)

What Data Types Do We Know?

- Integers (eg. `19`)
- Floating point numbers (eg. `19.5`)
- Strings (eg. `"Hello"`)
- Booleans (`True` and `False`)

What Data Types Do We Know?

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

What Data Types Do We Know?

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

What Data Types Do We Know?

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

What Data Types Do We Know?

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

File Modes

- `"r"` will open the file for reading

File Modes

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

File Modes

- `"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

File Modes

- `"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: write the file `story.txt` in 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: write the file `story.txt` in 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.

```
>>> f.read()
''
```

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 camper...\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 you system to work with that file.

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

Using Standard Input/Output as a File

If you `import sys`, you can use the file objects `sys.stdin` and `sys.stdout` as files.

```
import sys
for line in sys.stdin:
    sys.stdout.write(line)
```

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 course website.

Don't forget the documentation!

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!