Files

Fall 2019

Files

- Python can open external files to process that aren’t other Python files
  - Includes items like text files, csv files, etc
- Can open files in various different “modes” that determines what you’ll be doing with the file included
  - Cannot have a file open in more than one mode
- Files should be closed once you’re done using them; every file open should have a corresponding file close

File Modes

Mode	Description
Write (w)	Writes data to a file; overwrites any previous data the file had so it only contains the newly written data
Read (r)	Reads data from a file
Append (a)	Writes data to a file; adds on to any previous data the file contained

Files: Code Anatomy

- Open a file through built-in Python function "`open`"
  - First argument: file name
    - File name must contain entire path if it's not in the same folder as python
  - Second argument: open mode
- Files are closed through "`.close()`"
  - How you call it depends on what you named your file when it was opened

```
# opens file in read mode
nameFile = open('names.txt', 'r')

# gets first line, then prints it
line = nameFile.readline()
print(line)

# closes file out of read mode
nameFile.close()

# opens file in append mode
nameFile = open('names.txt', 'a')

# adds line, then closes file
nameFile.write("Maggie Neal\n")
nameFile.close()
```

Files: Code Anatomy

- To read a singular line from the file, you use "`.readline()`"
  - This gets the entirety of the line (including whitespace)
- To write to a file you use "`.write()`"
  - Whatever you want written should be inside the parentheses
  - You must *manually* add white space (newlines)
  - Unlike Python's print, it does not do it for you

```
# opens file in read mode
nameFile = open('names.txt', 'r')

# gets first line, then prints it
line = nameFile.readline()
print(line)

# closes file out of read mode
nameFile.close()

# opens file in append mode
nameFile = open('names.txt', 'a')

# adds line, then closes file
nameFile.write("Maggie Neal\n")
nameFile.close()
```

Files: Reading Multiple Lines

- Can loop through a file to get a dynamic (changing) number of lines
- Must get the first line of the file before the loop body
- Must get the next line of the file (and so forth) inside of the loop body

```
# reads file contents in a loop

# opens file in read mode
infile = open('names.txt', 'r')

# gets first line
line = infile.readline()

# loops until line is empty
while(line != ''):
    print(line)

    # sets line to next line in file
    line = infile.readline()

# closes file at the end
infile.close()
```

Code output:

```
Kortni Neal

Devin Neal

Maggie Neal
```

Files: Stripping Whitespace

- Uses Python built-in function .rstrip()
  - Removes all trailing whitespace: tabs, newlines, spaces, etc
  - Does not remove whitespace at the beginning of a variable
- Important to not have variables that contain unintended whitespace

Code output:

```
Kortni Neal
Devin Neal
Maggie Neal
```

```
# opens file in read mode
infile = open('names.txt', 'r')

# gets first line
line = infile.readline()
line = line.rstrip()

# loops until line is empty
while(line != ''):
    print(line)

    # sets line to next line in file
    line = infile.readline()
    line = line.rstrip()

# closes file at the end
infile.close()
```

Files: Notes

- When opening a file in “read” mode, the file *must* exist before opening, or the program will error out
- When open a file in “write” or “append” modes, if the file doesn’t exist it will be *created*
- Can also write to files in a loop

```
# writes file contents in a loop
outfile = open('numbers.txt', 'w')

# writes 0-9 to file
for i in range(10):
    line = str(i) + "\n"
    outfile.write(line)

# closes the file
outfile.close()
```

Files: Notes

- There are other ways to read/write to files
  - They mainly include lists, which haven't yet been covered

Function	Description
.readline() *	Reads in a line from the file
.write(*item*) *	Writes a specific item to the file
.read()	Reads the entire contents of the file
.readlines()	Reads all the lines of the file into a list
.writelines(*list*)	Writes items from a list to the file

Only those indicated by an * are required to know as of now, due to lists not being covered