

CSCI 127: Joy and Beauty of Data

Lecture 10: Files

Reese Pearsall

Summer 2021

<https://reese.github.io/classes/summer2021/127/main.html>

Announcements

Lab 6 due **Thursday** @ 11:59 PM

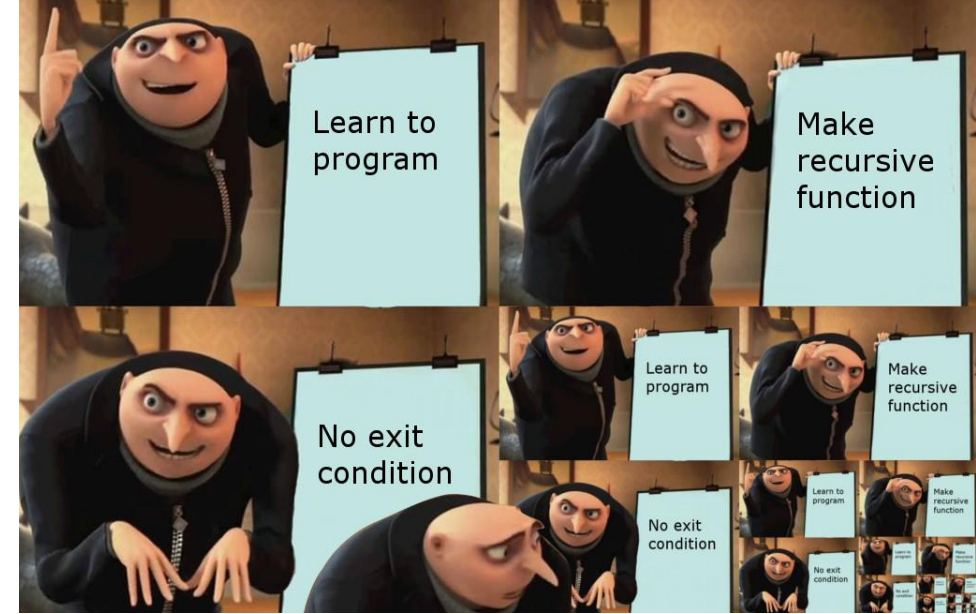
Program 3 due **Monday 6/7** @ 11:59 PM

(After today, you will be able to finish both Lab 6 and Program 3)

Currently grading the midterm exam and program 2

Gradebook on D2L has been updated to reflect different grade categories

*Remember to sign up for a 1 on 1 time w/ Reese



When your CS instructor is teaching you about recursive functions and you think you found one

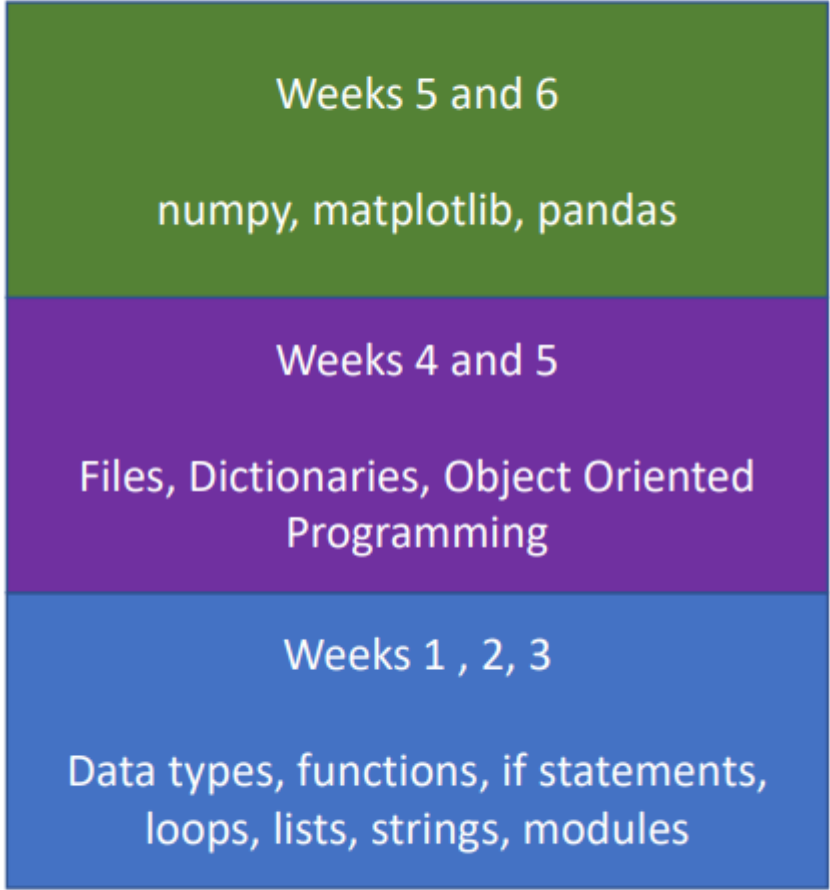


Don't forget, you can:

- Start late
- Start over
- Be unsure
- Try and fail

And still succeed





Data Science in Python

Advanced Python

Basics/Foundation of Python



Reading from files -- Motivation

So far, we've defined data *within* our program.

If we have a lot of data, then our program can become very convoluted and messy

Sometimes, it is easier to have our data exist in some *external* file that we can read from

The files we will be working with will be *thousands* of lines long

Quick Review - .split()

`list_name = string_name.split(delimiter)` is how we can divide up a string (`string_name`) based off a character (`delimiter`) and convert it into a list (`list_name`)

Quick Review - .split()

`list_name = string_name.split(delimiter)` is how we can divide up a string (`string_name`) based off a character (`delimiter`) and convert it into a list (`list_name`)

`greeting = "Hello. My name is Reese. Welcome to CSCI 127. I hope you enjoy."`

`my_list = greeting.split(".")`

Quick Review - .split()

`list_name = string_name.split(delimiter)` is how we can divide up a string (`string_name`) based off a character (`delimiter`) and convert it into a list (`list_name`)

`greeting = "Hello | My name is Reese | Welcome to CSCI 127 | I hope you enjoy"`

`my_list = greeting.split("|")`

Quick Review - .split()

`list_name = string_name.split(delimiter)` is how we can divide up a string (`string_name`) based off a character (`delimiter`) and convert it into a list (`list_name`)

```
greeting = "Hello My name is Reese Welcome to CSCI 127 I hope you enjoy"
```

```
my_list = greeting.split(" ")
```








```
print(my_list) → ['Hello', ' My name is Reese', ' Welcome to CSCI 127', ' I hope you enjoy', '']
```

0 1 2 3 4

Reading from files

Steps for reading a file

STEP 0: Make sure your .py file and file you are reading from are in the same folder/directory !!

 book_file	12/14/2020 9:57 AM	Python File
 calculate_football_wins	12/10/2020 6:27 AM	Python File
 calculate_mode	12/9/2020 5:51 AM	Python File
 celsius	12/1/2020 5:31 AM	Python File
 change	12/3/2020 6:07 AM	Python File
 change_updated	12/3/2020 7:26 AM	Python File
 classics	12/14/2020 8:28 AM	Microsoft Excel C...

Reading from files

Steps for reading a file

Name of file. Could be
a .txt, .in, etc

"r" means you reading
from the file

```
file = open("my_file.csv", "r")
```

← 1. Open the file

```
file.readline()
```

← 2. Discard the column headers (if necessary)

```
for each_line in file:
```

← 3. Do a for loop through each line in the file

```
    row = each_line.split(",")
```

← 4. Convert each line of the file (string)
to a list using **.split()**. If you reading
from a .csv file, you will always split on
a comma (","), otherwise you will need
to figure out what to split on.

```
    ## do stuff here with row
```

```
file.close()
```

← Close the file

Using classics.csv (<https://corgis-edu.github.io/corgis/csv/>)

Write a function that will:

- Calculate the book with most sentences

- Given an author name, calculate the number of books published by them

- Given a year, calculate number of books published that year

Announcements (Wednesday):

Lab 6 due **Thursday** @ 11:59 PM

Program 3 due **Monday 6/7** @ 11:59 PM



Today: More on reading from files, writing to files

Writing to Files

Name of file that will be created

"w" for write

```
file = open("out.txt", "w")
```

1. Open the file

```
for i in range(10):  
    file.write("hello x" +str(i))
```

2. Write to file using **.write()**
Note: You cannot use commas like you can with print()
Note2: Everything you write must be a string

```
file.close()
```

3. Close the file
IMPORTANT

Example: Arcade High Score Table

Write a Python Program that will read in a series of 10 arcade game scores along with a player name. These scores are sorted from greatest to least

When the program is run, it should prompt the user for a new score and name. If the score the user inputs is the highest score so far, it should get placed at the very top of the scores. Otherwise, it should get placed in the appropriate spot in the top 10 (if it made it within the top 10)

The program should then write out to the same file it read from with the updated high score list. The high score list should ONLY have the top 10 scores and player names

