ENGR 298: Engineering Analysis and Decision Making – Files and Paths

How to actually find information in your computer…

Dr. Jason Forsyth

Department of Engineering

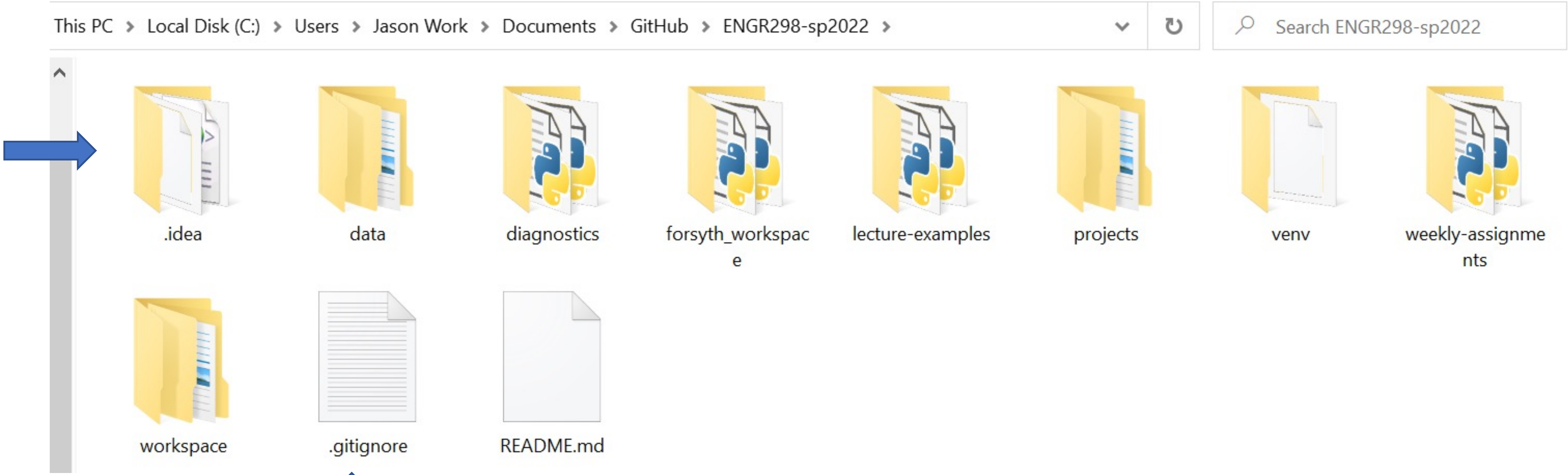
James Madison University

Two main units of organization: files and directories

- **File**: an individual object with a computer that holds information. A file may have a particular *size*, be in a certain *formats*, or have read/write *permissions* associated with it.
    - *Size*: expressed in KiloBytes (KB), MegaBytes (MB), GigaBytes (GB)…etc.
    - *Formats*: text (.txt), JSON (.json), Word (.docx), Zip (.zip)…etc.
    - *Permissions*: an application on a computer may have permission to read, write, or execute a particular file (if that file is a program)

- **Directory**: a container for files. May also contain other directories to form a hierarchy.

The ‘ENGR298-sp2022’ directory contains 9 folders and 2 files



This PC > Local Disk (C:) > Users > Jason Work > Documents > GitHub > ENGR298-sp2022 >

Search ENGR298-sp2022

.idea

data

diagnostics

forsyth_workspace

lecture-examples

projects

venv

weekly-assignments

workspace

.gitignore

README.md

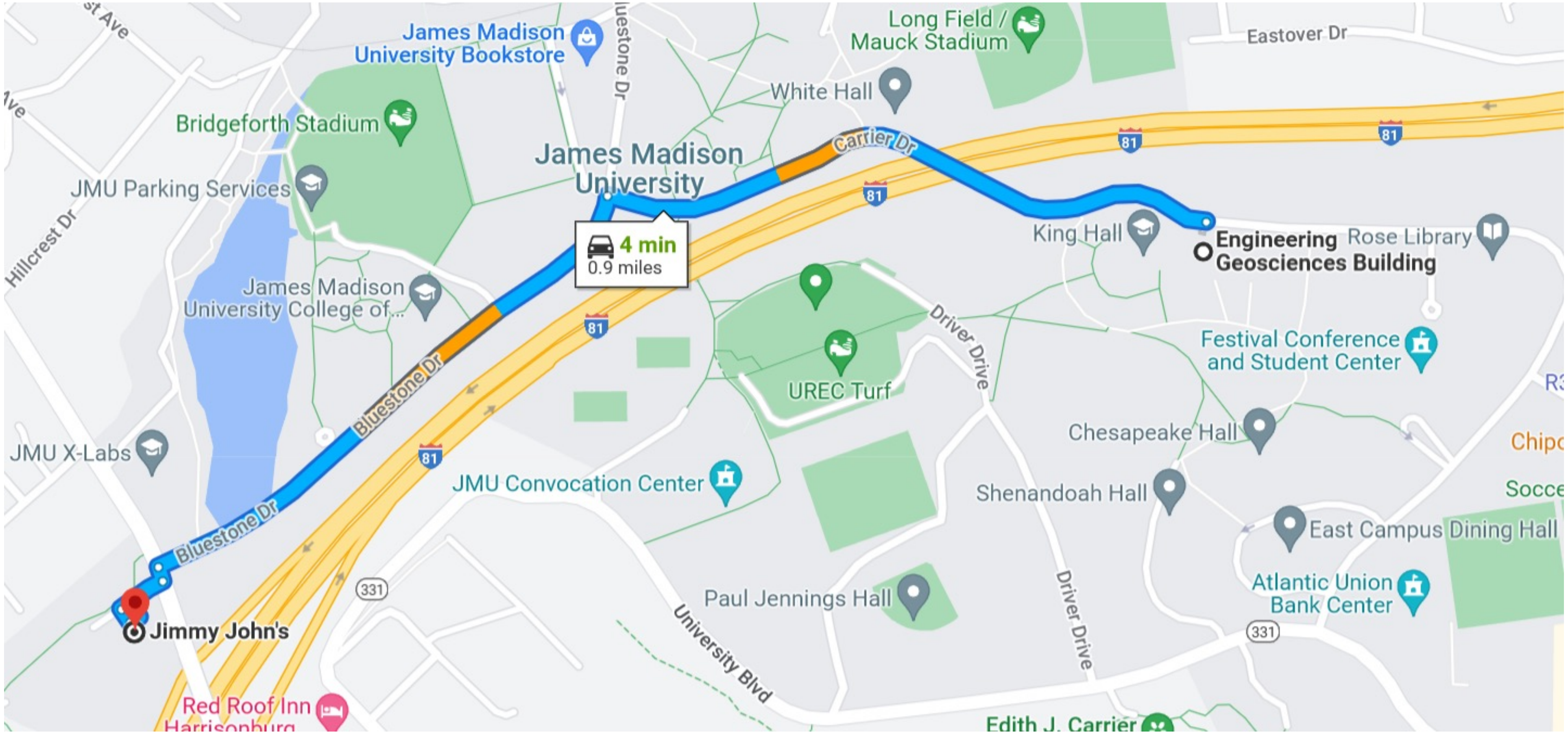
Some directories and files may be hidden in the graphical window

Paths: Relationships between files..

- To load or execute a file, a program needs to know where that file is located, relative to its own location. This information is called a **file path**.

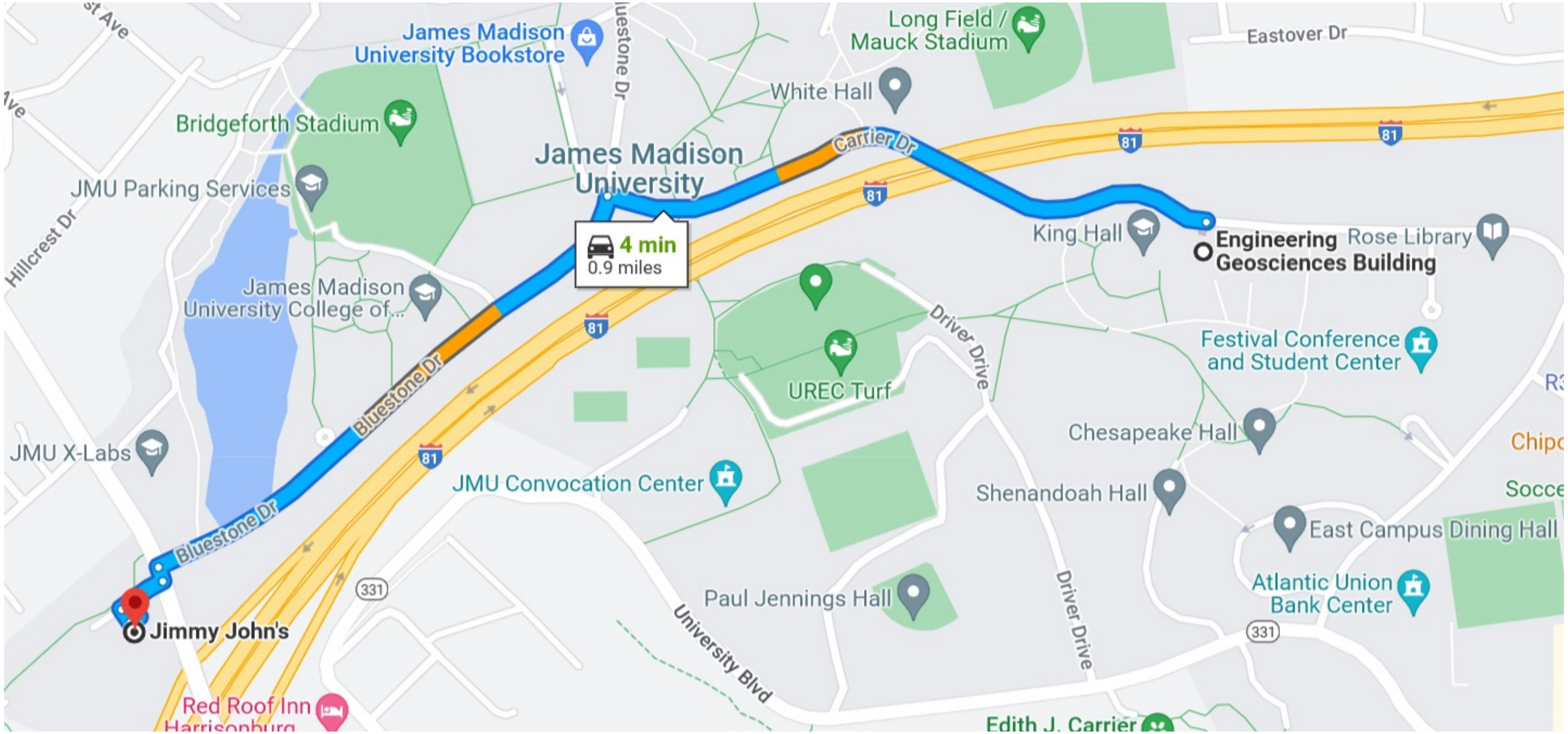
- If a program and the file are in the same directory, it is likely it can be told simply the filename: “my_data.txt”. However, if the file is in another folder, the program will need to be provided an **absolute** or **relative** path.
  - Absolute path: instructions for locating an object given some common/absolute starting location/grid.
  - Relative path: instructions relative to the current location/directory to retrieve the object.

Relative Paths to Jimmy Johns: Only works if we know where we start



Relative path: From this room, exit turning right and walk down the main hallway. Then walk down the stairs past the James Madison statue and follow the main road. Continue until the four way stop at Godwin, turn left and continue to Port Republic. Cross Port Republic (without getting hit) and Jimmy Johns is on the left.

Absolute Paths to Jimmy Johns: Only works if we agree to a common organization system



Absolute path: Drive to 1314 Hillside Ave, Harrisonburg VA 22801

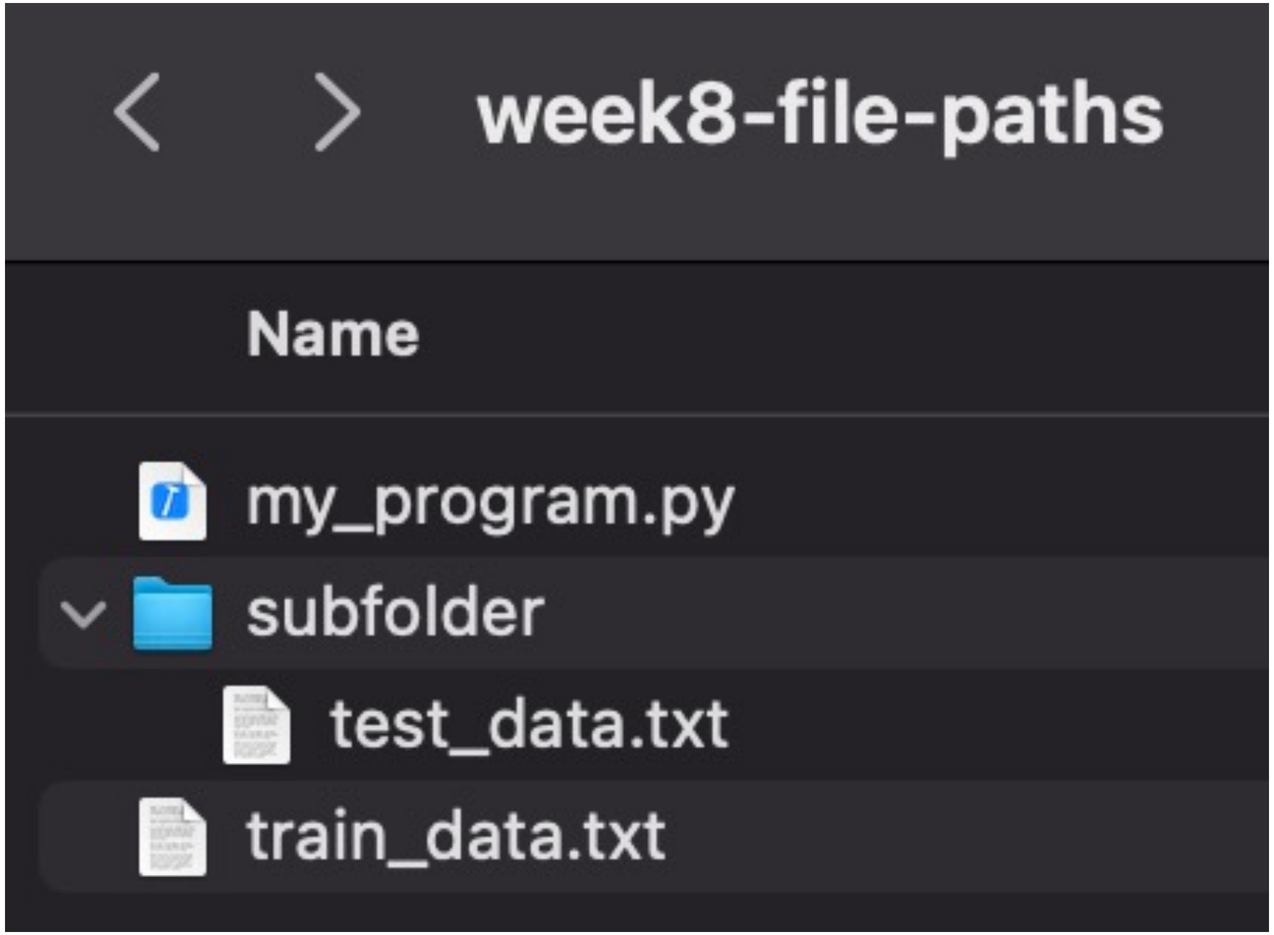
Relative and Absolute Paths in a Computer

- Both relative and absolute paths can be used to indicate where files are. Absolute paths require a “root” to establish a common structure.
- A period “.” indicates the current directory. This is where you are currently located.
- Two periods “..” indicates “up” a directory
- A slash “/” indicates the name is a directory. Descend into that directory.

A simple example

- The directory ‘week8-file-paths’ contains three items. Two files called my_program.py and train_data.txt. And another directory called subfolder.

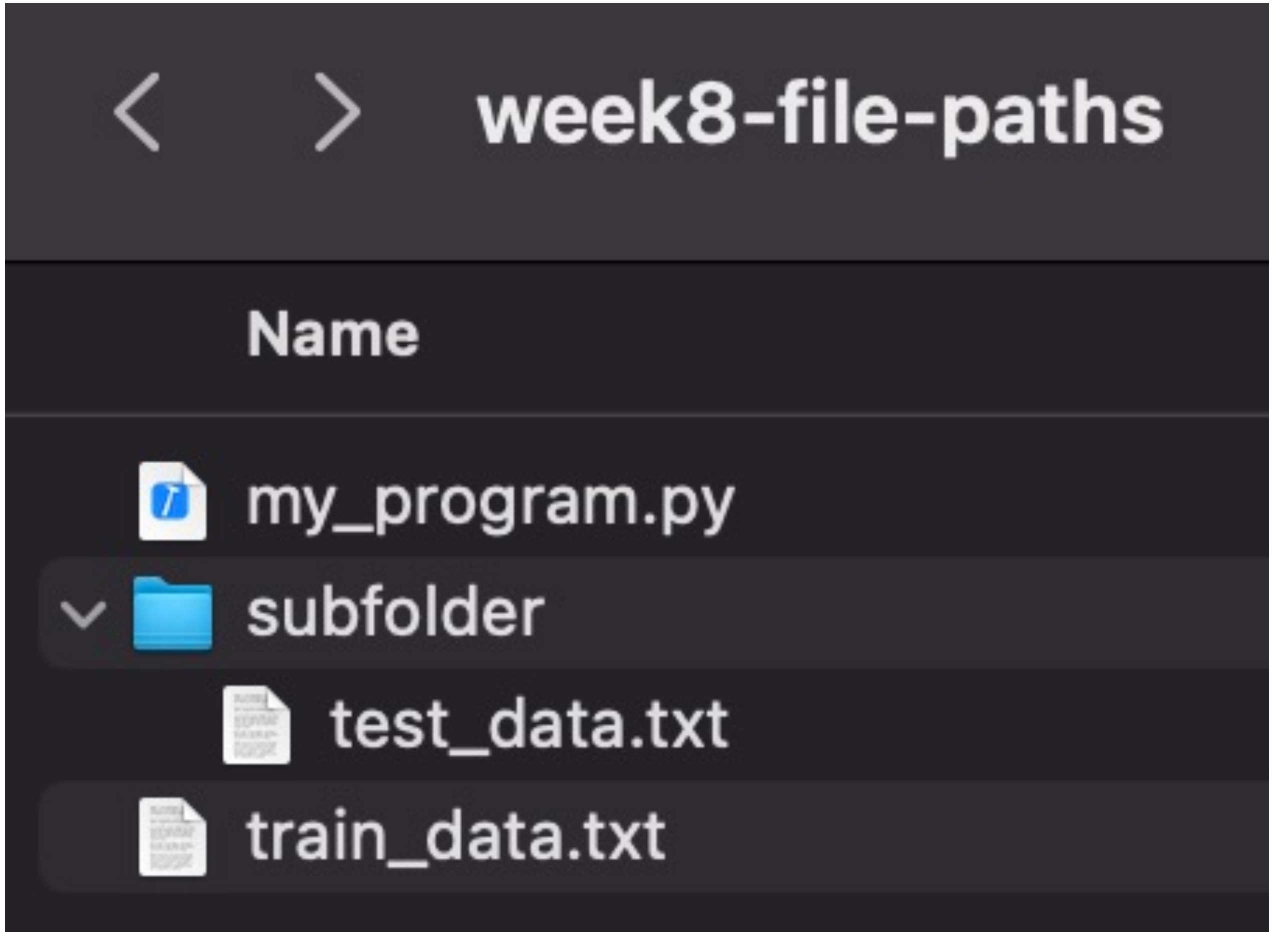
- subfolder contains another file called test_data.txt

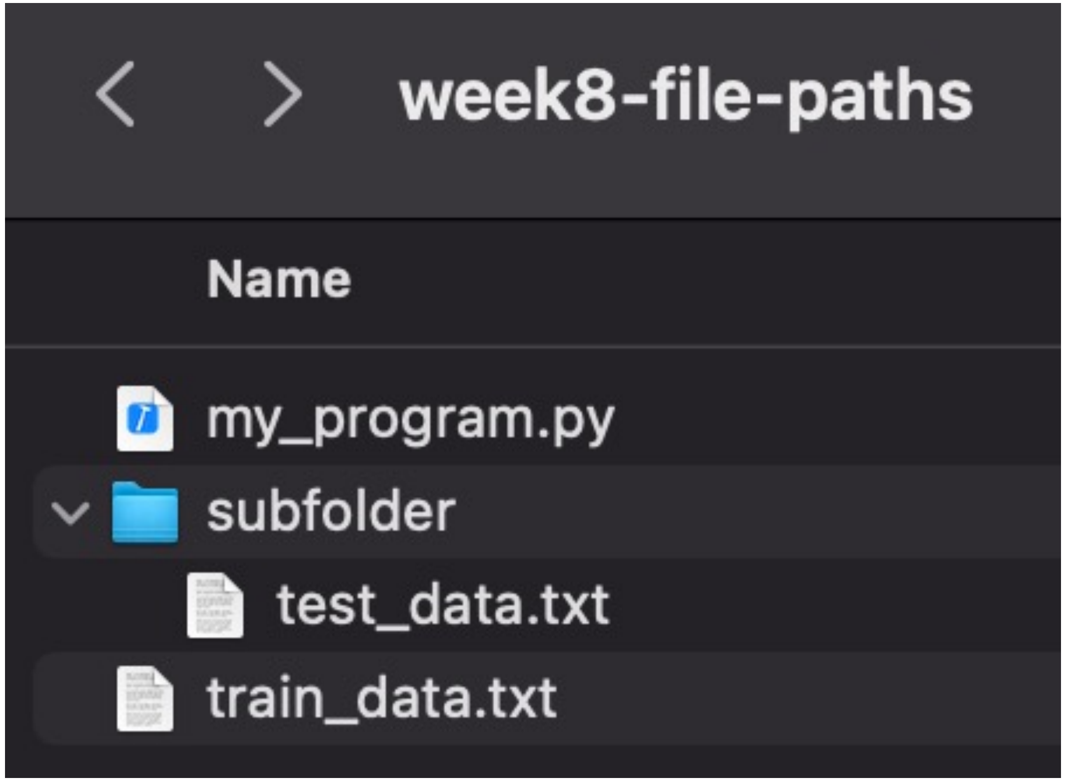


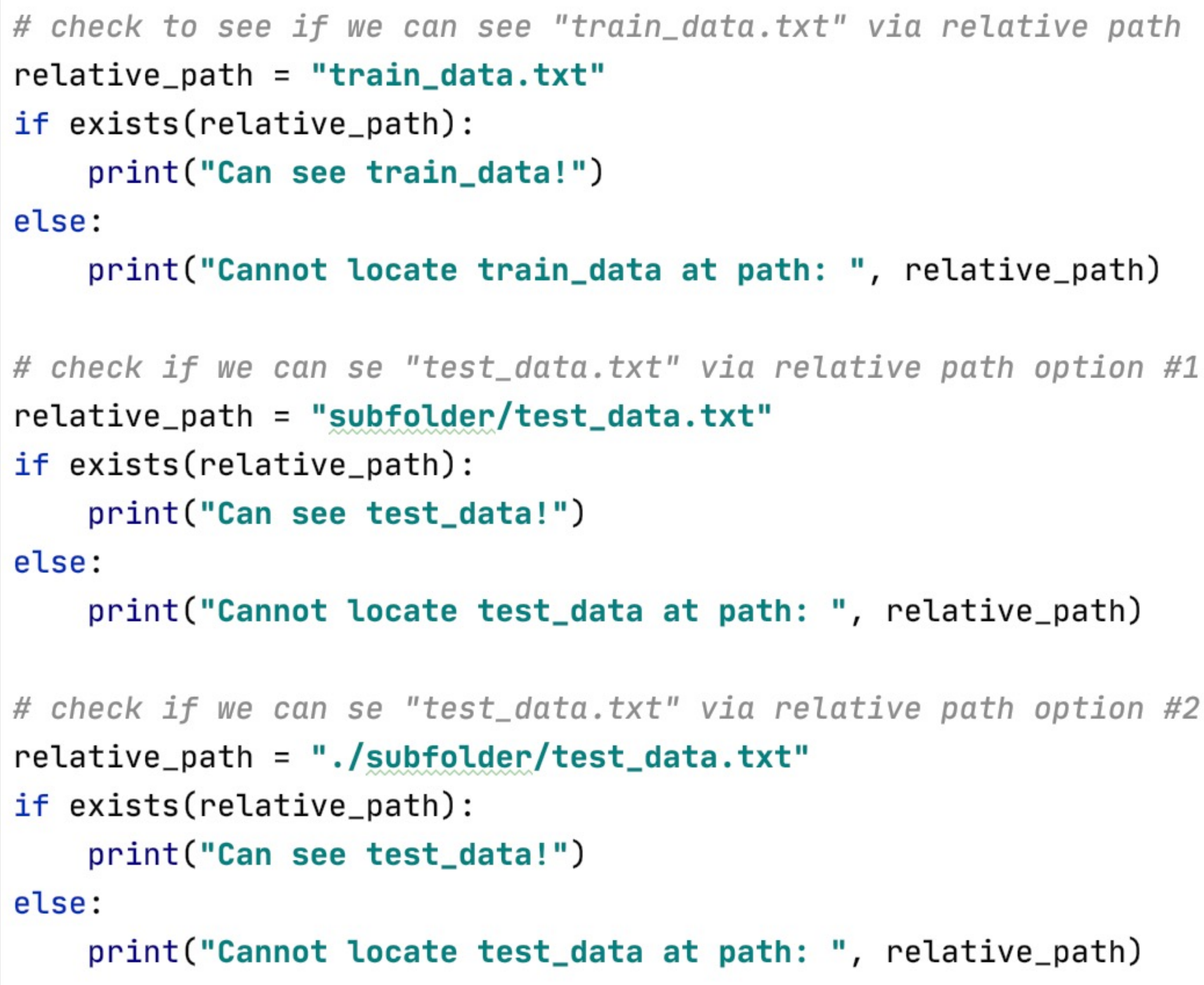
A simple example

- If my_program.py wants to load “train_data.txt” it can simply be passed “train_data.txt”

- However, if it wishes to load test_data.txt then it must be provided the relative path of “subfolder/test_data.txt” **or** “./subfolder/text_data.txt”







```python
# check to see if we can see "train_data.txt" via relative path
relative_path = "train_data.txt"
if exists(relative_path):
    print("Can see train_data!")
else:
    print("Cannot locate train_data at path: ", relative_path)

# check if we can se "test_data.txt" via relative path option #1
relative_path = "subfolder/test_data.txt"
if exists(relative_path):
    print("Can see test_data!")
else:
    print("Cannot locate test_data at path: ", relative_path)

# check if we can se "test_data.txt" via relative path option #2
relative_path = "./subfolder/test_data.txt"
if exists(relative_path):
    print("Can see test_data!")
else:
    print("Cannot locate test_data at path: ", relative_path)
```

Not so simple with absolute paths…

- We could try the same approach using absolute paths but every computer/user has a different organization. While the ENGR298 repo has the same structure for all of us, we each have different operating systems and usernames.

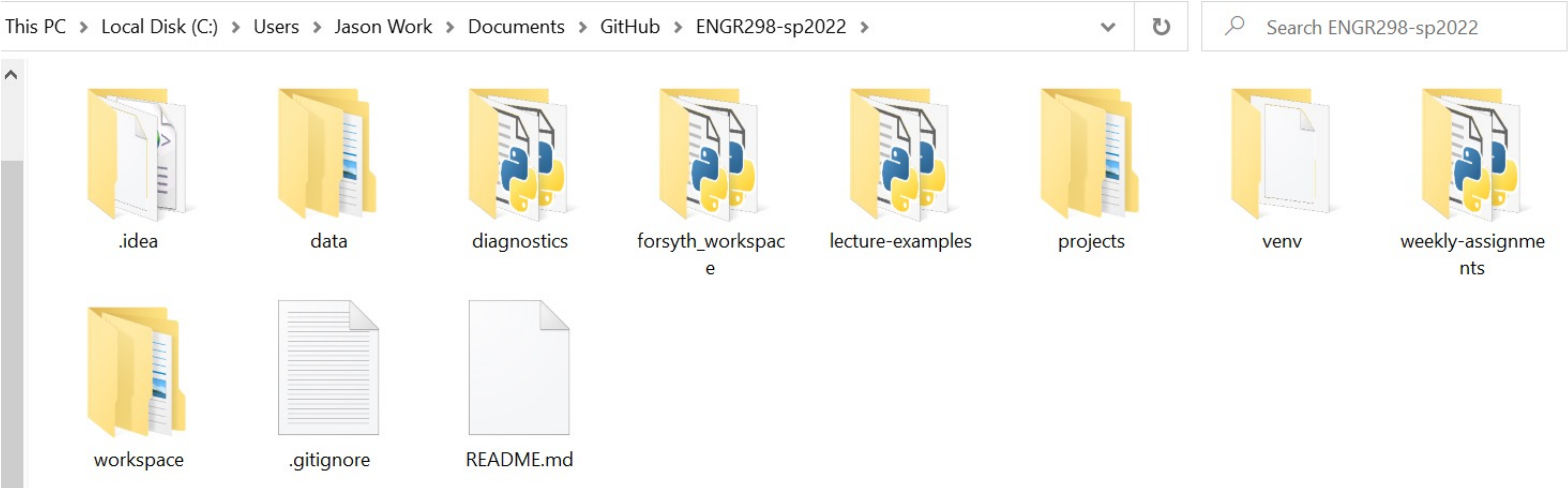
- For my desktop the absolute path is: *C:\Users\Jason Work\Documents\GitHub\ENGR298-2022-Private\lecture-examples\week8-file-paths\my_program.py*

- For my macbook the absolute path is: */Users/jforsyth/GitHub/ENGR298-2022-Private/lecture-examples/week8-file-paths/my_program.py*

- Each operating system has its own root (c:\ vs /) and directory indicator (/ vs \ ) ☹☹

Consider how our course repository looks in two different operating systems…

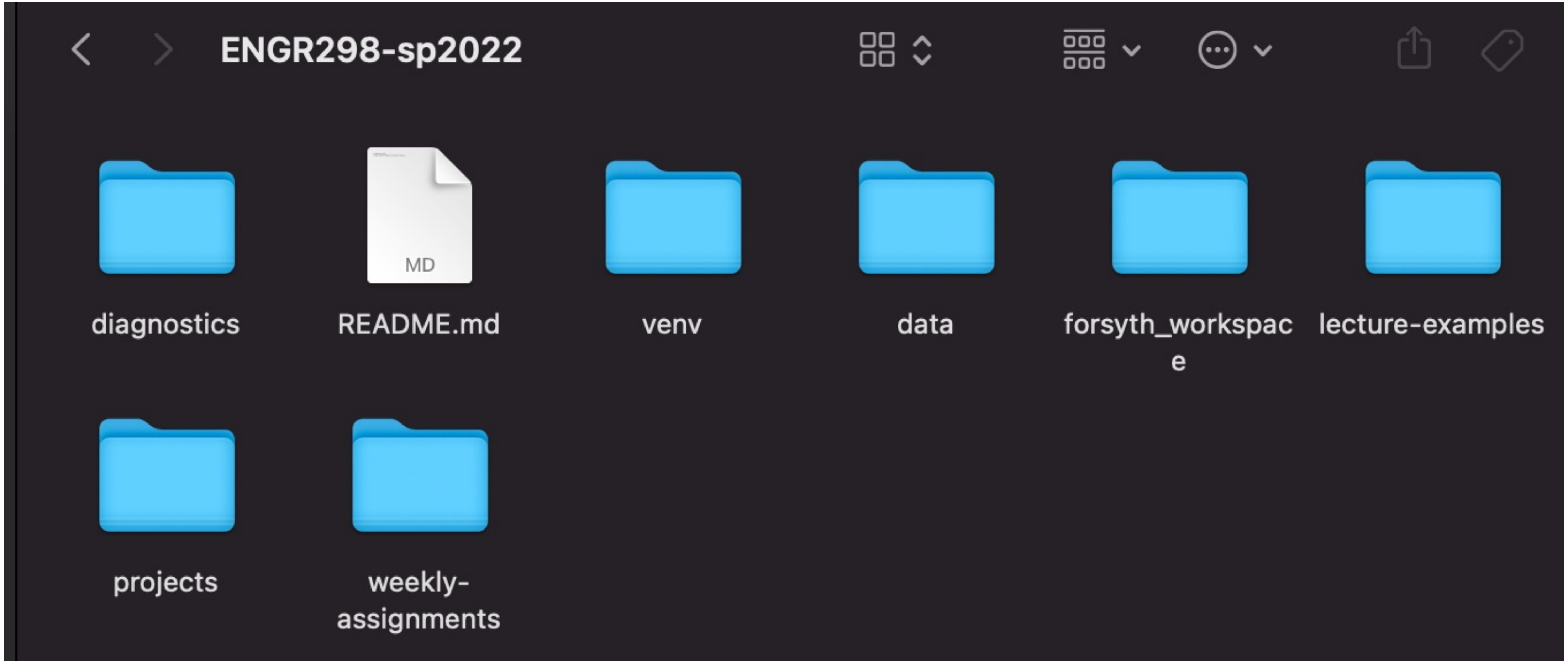
Course repo within Windows



This PC > Local Disk (C:) > Users > Jason Work > Documents > GitHub > ENGR298-sp2022 >

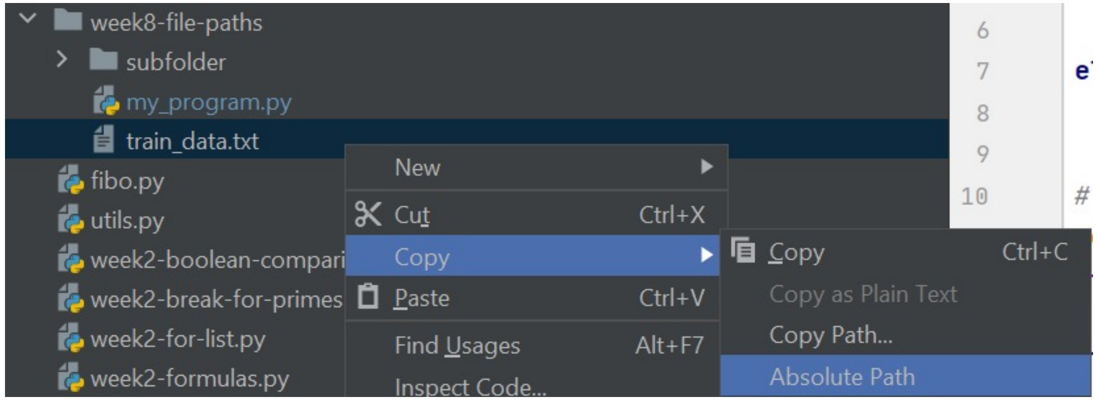
C:\Users\Jason Work\Documents\GitHub\ENGR298-sp2022

Course repo within Mac OS



```
/Users/jforsyth/Documents/GitHub/ENGR298-sp2022
```

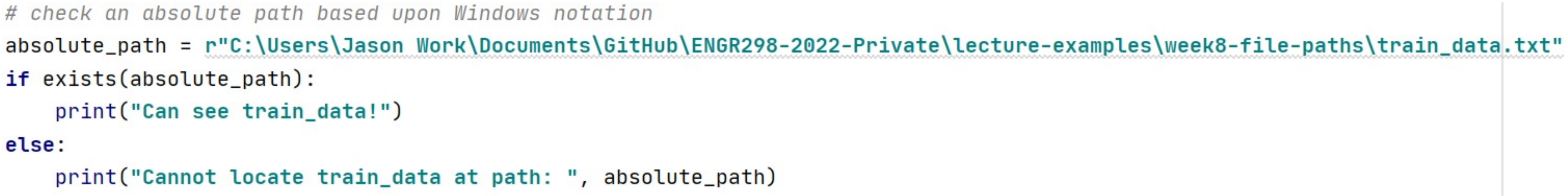
You can grab an absolute path for your file system…



- Right-clicking on a particular file will provide an absolute path for an individual file.
- Useful if you’re really struggling to have a program load a file.
- Will only work on that machine though…

This code only runs on my desktop....

Need to add ‘r’ to indicate it is a raw string. Otherwise Python will interpret \ as escape character and not as folder



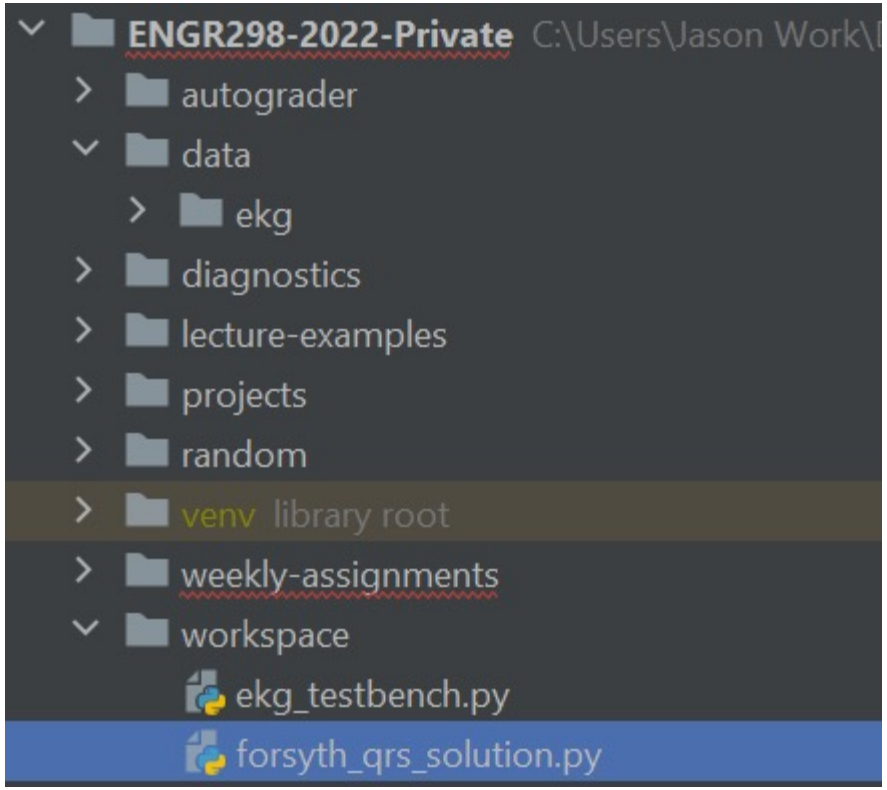
```
# check an absolute path based upon Windows notation
absolute_path = r"C:\Users\Jason Work\Documents\GitHub\ENGR298-2022-Private\lecture-examples\week8-file-paths\train_data.txt"
if exists(absolute_path):
    print("Can see train_data!")
else:
    print("Cannot locate train_data at path: ", absolute_path)
```

How will we resolve this…

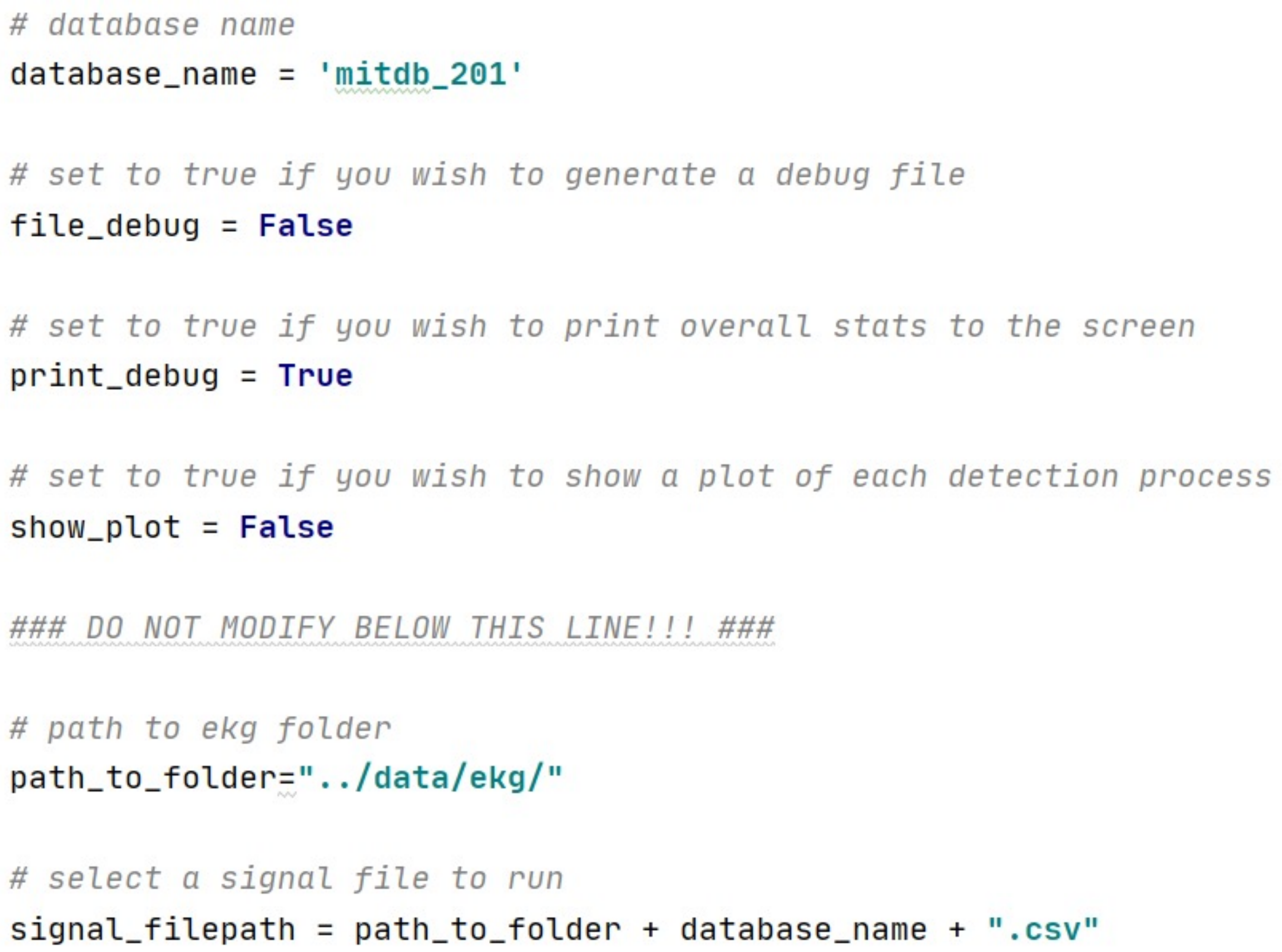
- Generally the course will use relative file paths to avoid conflicts between operating systems and individual user names.

- This is why the repo structure is important and files (data and programs) should not be moved around => the relative file paths will break.

- If you receive a “file not found” error take a look at what is reported. You can likely fix it by moving up or down directories. Always consider where your executable program is located to the “here” or “.” You can create paths from that point.



```
ENGR298-2022-Private  C:\Users\Jason Work\
  autograder
  data
    ekg
  diagnostics
  lecture-examples
  projects
  random
  venv  library root
  weekly-assignments
  workspace
    ekg_testbench.py
    forsyth_qrs_solution.py
```



```
# database name
database_name = 'mitdb_201'

# set to true if you wish to generate a debug file
file_debug = False

# set to true if you wish to print overall stats to the screen
print_debug = True

# set to true if you wish to show a plot of each detection process
show_plot = False

### DO NOT MODIFY BELOW THIS LINE!!! ###

# path to ekg folder
path_to_folder="../data/ekg/"

# select a signal file to run
signal_filepath = path_to_folder + database_name + ".csv"
```

Your QRS solution MUST be located in a folder at the same “level” as the data folder. Ekg_testbench must be located in that same folder as well. If you want to access challenge data in ekg, change to “../data/ekg/challenge”