Learning the command line interface (CLI)

## CSC160, Prof. Phil Lombardo

(*A special thanks to Dr. Hank Feild for designing and writing this assignment. It has been gently adapted from his original work to suit the needs of our class*)

This activity focuses on getting you familiar with the command line. While working on the command line is a great general skill to have, it will be essential for this class. To run python programs, and even interactively program, you need to know “where files live” on the computer. The command line will help you navigate to where a Python program exists on your computer and run it.

Prerequisites:

* you're running Windows, macOS, or Linux
* you've installed Visual Studio Code
* you've installed Python

Go through the steps of this activity with your assigned partner(s). Each of you should perform the steps on your own computer (this isn't the case with all activities, but it's important for this one), but keep pace with your partner(s). If you have trouble, talk it over with your partner before raising your hand for help from the instructor.

You will use Google Docs to record answers; for this particular lab, each team member should add answers to your own individual copy.

# Team members

|  |  |
| --- | --- |
| **Names:** |  |

# Learning Objectives

By the end of this part, you will be able to:

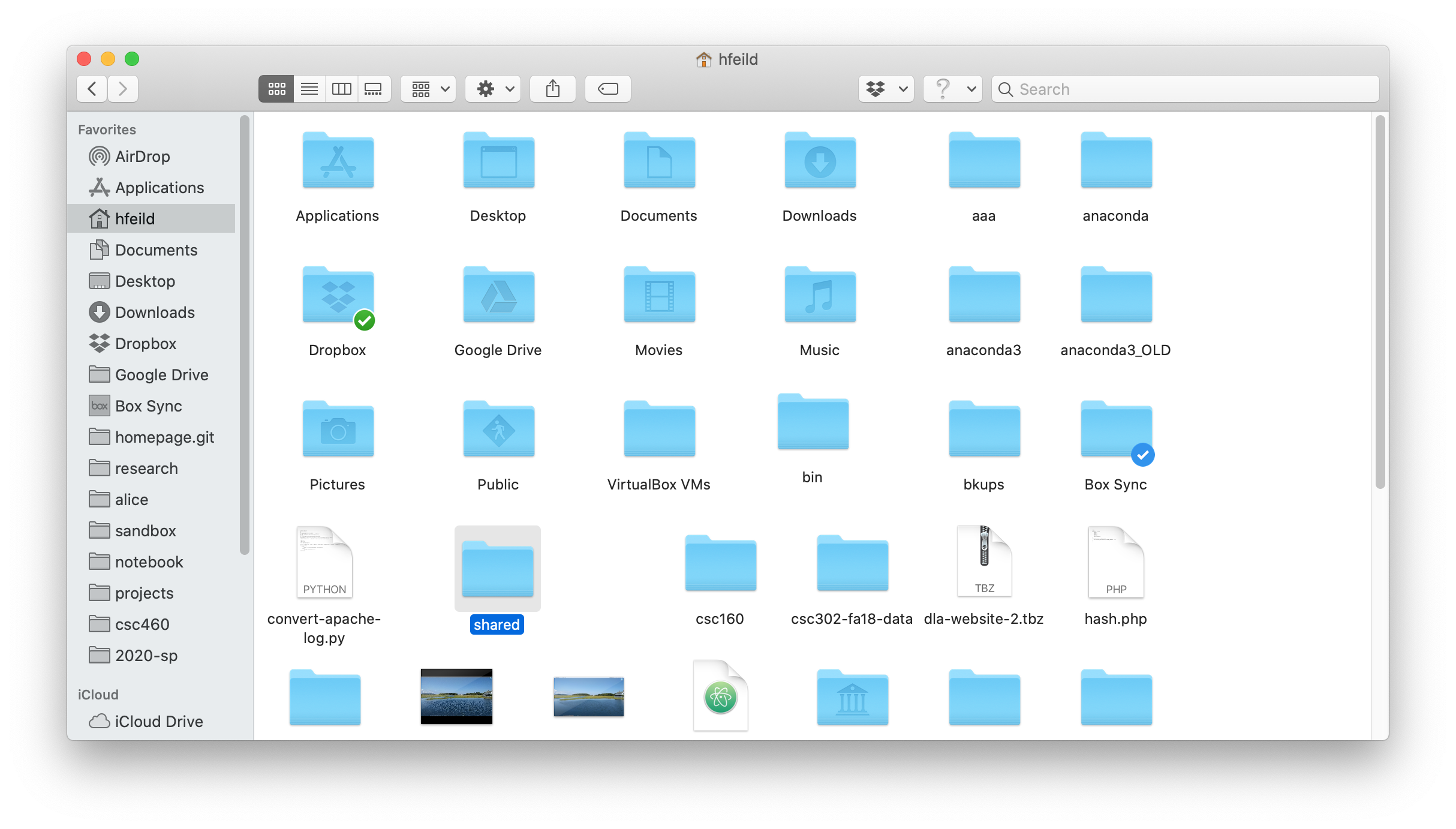
* describe how to open a terminal and navigate to a directory in the CLI
* list the contents of a directory using a CLI
* show the path to the current (working) directory in a CLI
* create a new directory from the CLI
* create an empty file from the CLI
* run the Python shell from the CLI
* run and interact with a Python program from the CLI

# Grading

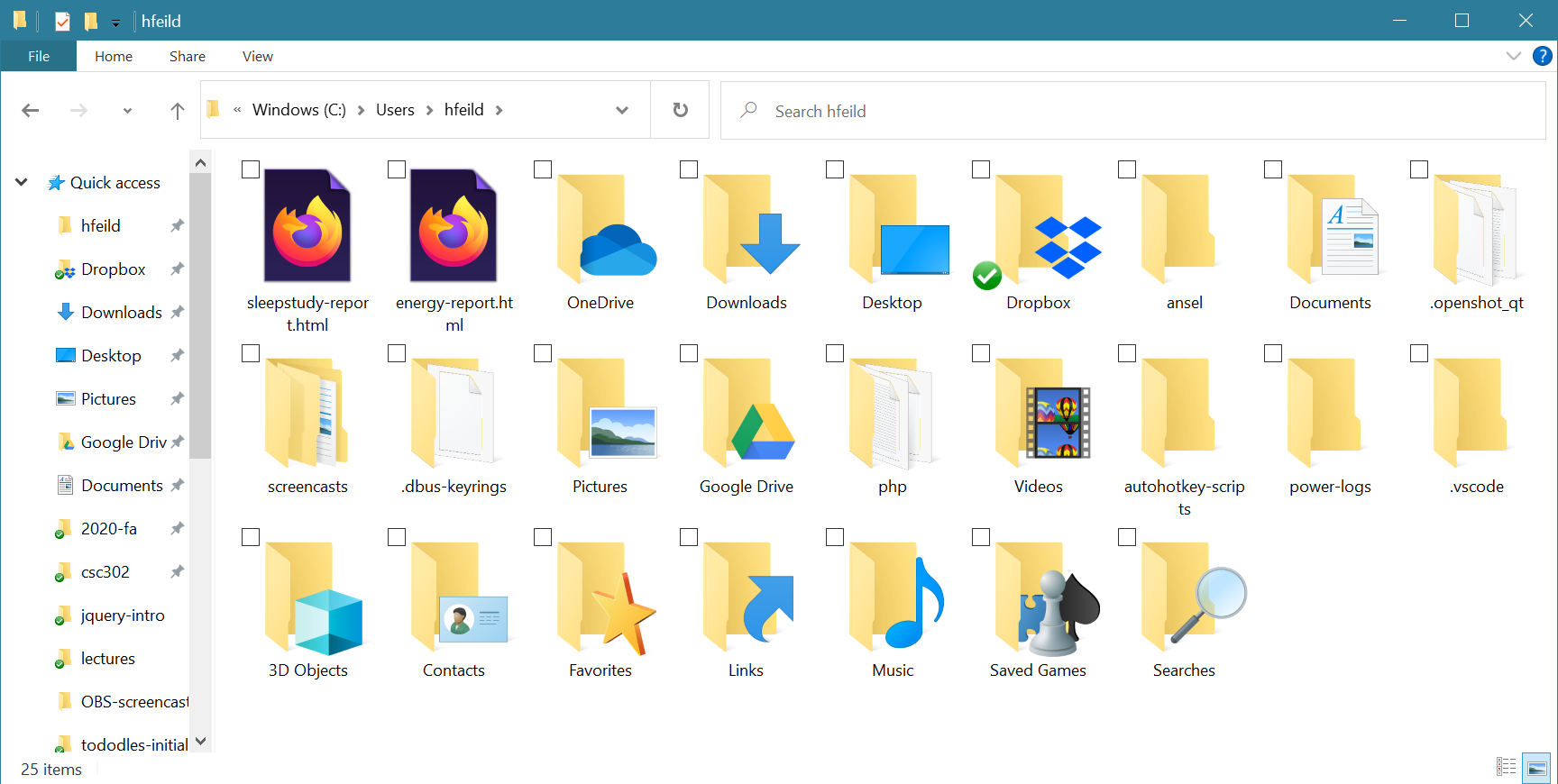
The following rubric will be used to grade your work on this lab part. See the syllabus for information about redo tokens for increasing your grade.

|  |  |  |
| --- | --- | --- |
|  | **Yes** | **No** |
| You worked diligently during the lab period and worked well with your partner. | 5 pts | 0 pts |

# Model



This is a Finder window (macOS only) opened up in my home folder (/Users/hfeild).



This is a File Explorer window (Windows only) opened up in my home folder C:\Users\hfeild.

# Steps & Questions

*Please place your answers in the purple boxes.*

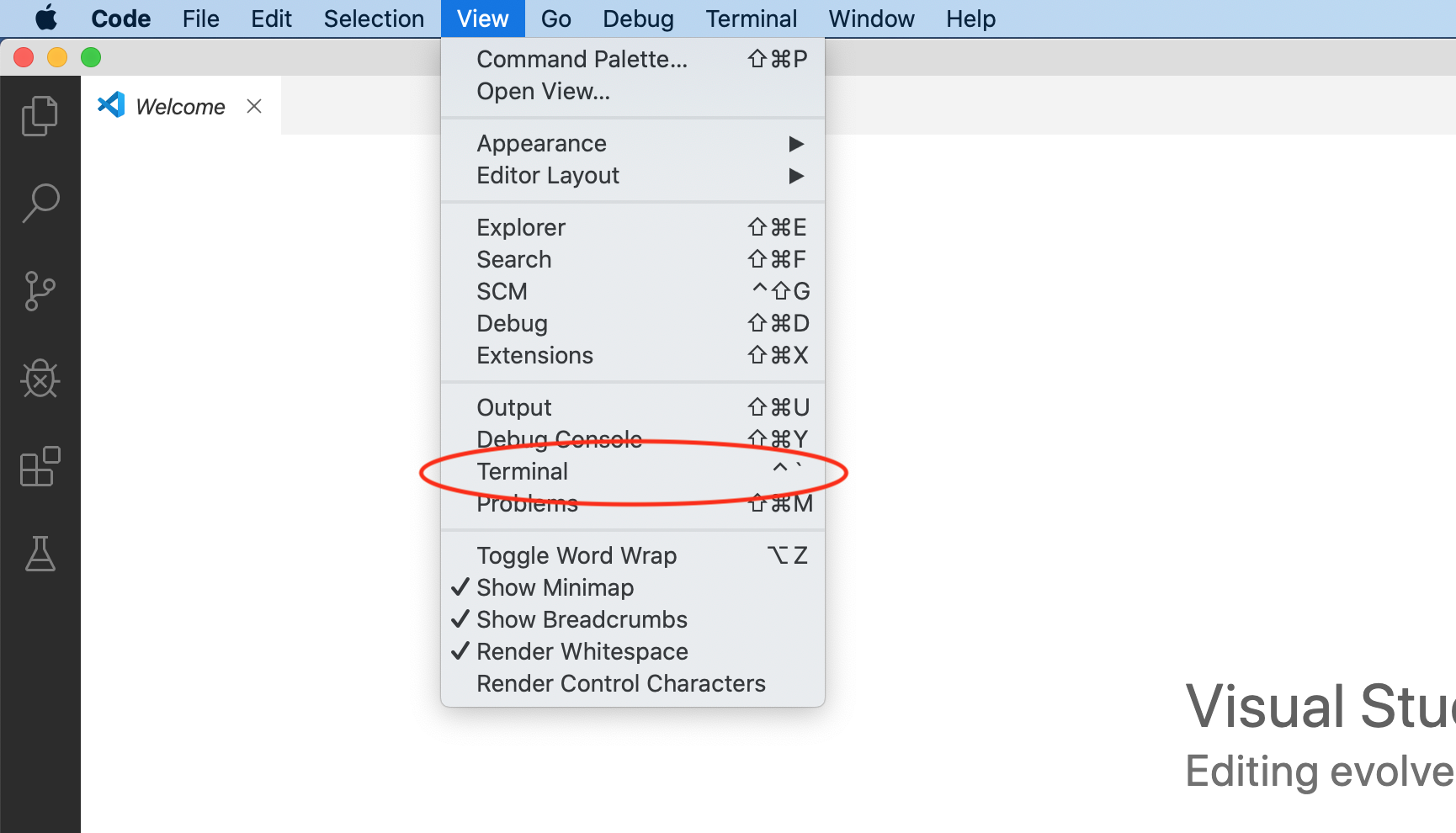
1. Generally speaking, what do applications such as Finder (on macOS) and File Explorer (on Windows) show us when we open a folder (said differently, what do the icons represent)?

|  |
| --- |
|  |

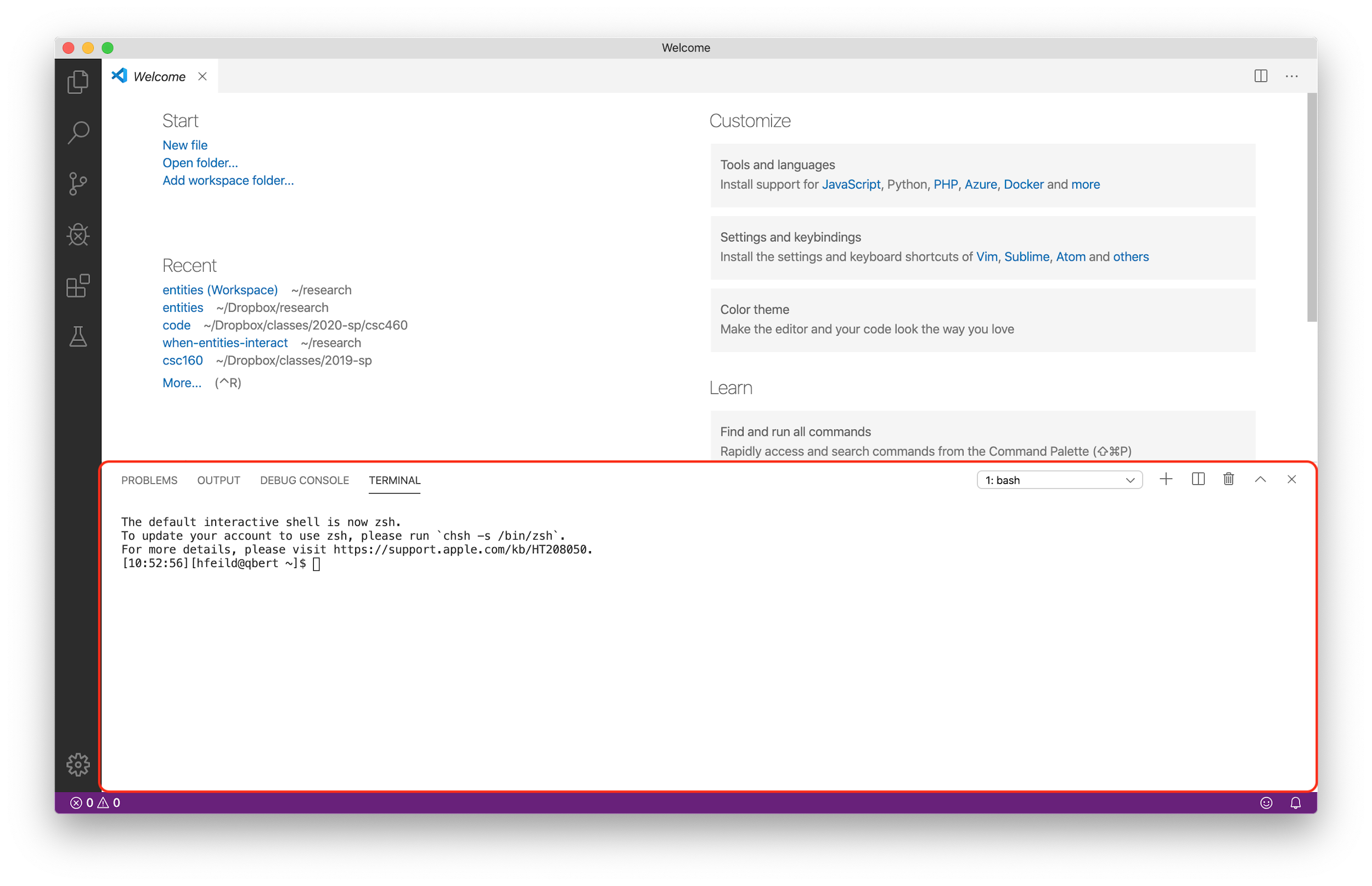
2. Suppose that I am using Finder/File explorer and I am currently looking at the contents of my home folder (as pictured above). I have a folder named "Documents", and inside of that, a folder named "csc160". What would I need to do to navigate into the "csc160" folder in Finder or File Explorer?

|  |
| --- |
|  |

Open Visual Studio Code. Once opened, click on "View" at the top and select "Terminal", as shown below.



That opens a terminal window with a command line interface (CLI) panel at the bottom of Visual Studio Code, outlined in red below.



3. In the CLI, type pwd and press the Enter (or Return) key. What is displayed? Copy and paste the output below.

|  |
| --- |
|  |

The pwd command lists the **p**ath to the **w**orking **d**irectory. A path is a sequence of folders that lead to a location on your computer's file system. *Directory* is another word for *folder*. A working directory is another way of saying "currently opened folder".

4. In the CLI, type cd ~/Downloads and press the Enter (or Return) key. What is displayed?

|  |
| --- |
|  |

5. Now enter the pwd command again. It should be different from the last time you entered it. Why do you think that is?

|  |
| --- |
|  |

The cd command **c**hanges the working **d**irectory to the given directory.

~ (the *tilde* symbol, and pronounced *til-da*) in a path is treated special and it serves as an abbreviation of the absolute path to your home directory. For example, entering

cd ~/Downloads

is the same as entering the following on macOS:

cd /Users/yourusername/Downloads

or this on Windows:

cd C:\Users\yourusername\Downloads

(where "yourusername" is whatever your username is; e.g., mine is hfeild).

6. What action is issuing the cd command equivalent to in the Finder/Windows Explorer example? That is, how do you change directories in Finder/Windows Explorer?

|  |
| --- |
|  |

7. Enter cd ~, then enter pwd again. What is the output?

|  |
| --- |
|  |

Entering cd ~ will change the working directory to your home directory. If you ever find yourself lost in the CLI, enter cd ~ to return to your home directory.

8. Enter cd Downloads and then enter ls. What is displayed (describe the output, don't list it)?

|  |
| --- |
|  |

The ls command **l**i**s**ts the contents (files and sub-directories) of the working directory. You can also provide ls a folder path and it will list the contents of that folder.

9. Now enter cd .. (that's cd followed by a space then two periods). Then enter pwd. Enter

cd .. again, followed by pwd. Looking at the output of pwd, what do you think the cd .. command does?

|  |
| --- |
|  |

Every directory has two special subdirectories (they are actually links, or shortcuts): . (single period) is a link to the current directory and .. (two periods) is a link to the parent directory. So cd .. changes to the parent directory of the current directory (equivalent to the "Up one folder" button in File Explorer (macOS doesn't have a similar button).

From the command line, navigate to your Documents folder using the cd command. From File Explorer or Finder, also navigate to your Documents folder. Type the following from the command line in Visual Studio Code:

mkdir csc160

10. Do you see a new folder appear in File Explorer or Finder after entering the command above? What is it called?

|  |
| --- |
|  |

Now change directories into csc160 in both the command line interface as well as File Explorer or Finder. If you are on **macOS**, enter the following command from the command line interface:

touch lab1.py

If you are on **Windows**, enter this command instead:

New-Item lab1.py

11. What do you see in File Explorer/Finder in the csc160 folder?

|  |
| --- |
|  |

The mkdir command creates a new directory with the given name; this is equivalent to creating a new file from Finder or File Explorer. The touch/New-Item commands create an empty file, much like you can do (but probably don't often) from Finder or File Explorer using the "New file" option, or creating a new file from Visual Studio Code and then saving it.

Changing gears, let's see about running Python. From the terminal at the bottom of Visual Studio Code, type:

python3

As long as you've installed Python correctly and all your system paths are up to date, you should see the interactive Python shell start up...something like this should appear:

[06:24:48][hfeild@virginia22 csc160]$ python3

Python 3.10.1 (tags/v3.10.1:2cd268a, Dec 6 2021, 19:10:37) [MSC v.1929 64 bit (AMD64)] on win32

Type "help", "copyright", "credits" or "license" for more information.

>>>

The ">>> " is the prompt that the interactive Python shell uses to tell you that it is waiting for you to enter the next bit of Python code to execute.

12. To the right of the ">>> " prompt, type the following Python *statement* and press enter:

print('Hello, world!')

What happens?

|  |
| --- |
|  |

13. Now try entering the following *expression*:

((10\*354.5) / 6.2) \*\* 2

What happens?

|  |
| --- |
|  |

14. Write a Python expression that calculates the area of a square with width 15.6. Write the expression below along with the output from the Python shell:

|  |
| --- |
|  |

Exit the Python shell and return to the system command line by calling the exit function:

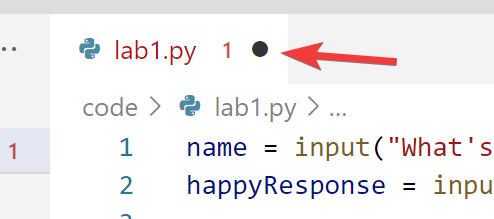
exit()

Be sure to include the parentheses directly after the word "exit" or it will not work.

In Visual Studio Code, open the file we "touched" earlier (**lab1.py**). Copy and paste (or re-type) the following Python code—**don't** **save** quite yet, though!

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | name = input("What's your name? ")  print("Hi, "+ name +"!")  happyResponse = input("Are you feeling happy today? (yes|no): ")  if happyResponse == 'yes':  print("Great! Me, too.")  else:  print("I'm sorry to hear that :( I hope your day turns around.") |

Take a look at the file tab (near the top of Visual Studio Code). Do you see the filled in circle?



15. In the terminal, run the program by typing the following command:

python3 lab1.py

What happens, if anything?

|  |
| --- |
|  |

16. Now save your file by either going to "File" -> "Save", or by pressing ctrl-s (Windows) or cmd-s (macOS). What happened to the filled in circle?

|  |
| --- |
|  |

17. Trying running the program again from the terminal. (Pro-tip, instead of re-typing the command, you can use the up and down arrow keys to go back through a history of recent commands.)

What happens when you run the program now?

|  |
| --- |
|  |

The input function will display the text provided in the parentheses, then waits for the user (whoever is running the program from the command line) to enter a response. Go ahead and enter your name, then when asked, reply to the next question with yes or no. When the program is finished running, it will return you back to your command line prompt (the thing that starts with $).

18. Copy and paste your interaction with **lab1.py** on the command line and paste it below.

|  |
| --- |
|  |

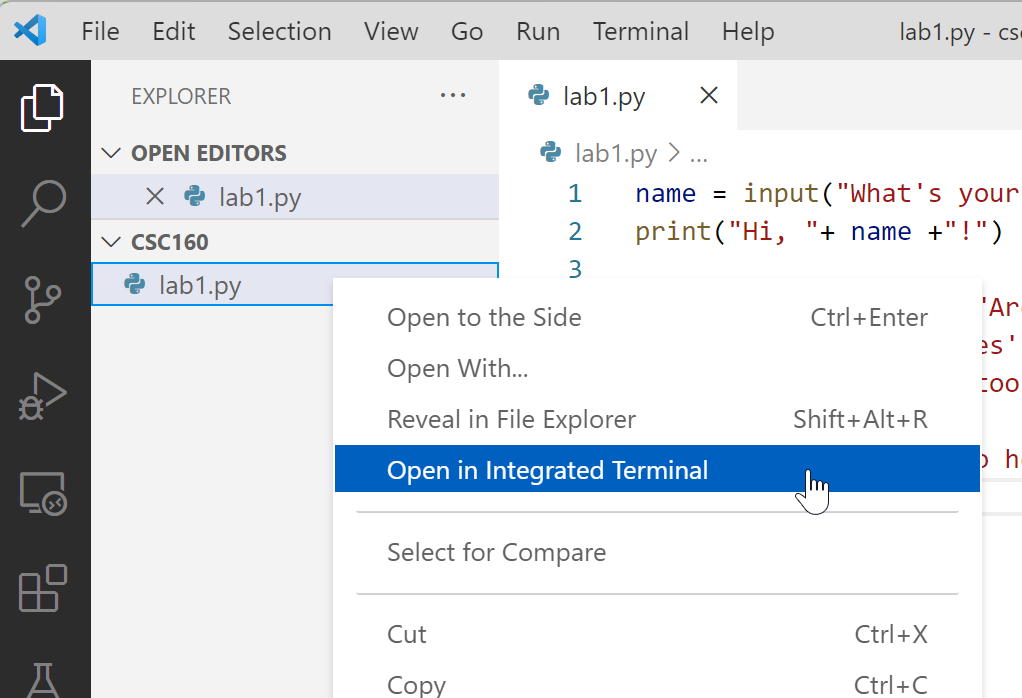
19. Now run the program again, but this time, don't enter anything when it asks for your name. Instead, press the key combo: ctrl-c (if you're on macOS, make sure you're pressing ctrl and not cmd).

Describe what happens when you do that.

|  |
| --- |
|  |

In a terminal, when you type ctrl-c, it sends a signal to the terminal to force exit whatever program (if any) is running. This is helpful if you are running a program that hangs (e.g., due to buggy code) or if you for some reason want to exit your program without going through the whole thing (e.g., if you have a program that asks for lots of inputs, but you don't want to have to go through all of them).

20. In the VS Code terminal, enter cd ~ to change your working directory to your home directory. Enter pwd to confirm you are in your home directory. On the left side of VS Code, find your Python source code file (**lab1.py**). Right click and select "Open in integrated terminal".



In the terminal that opens, enter the command pwd once again. What is the path of your working directory when you do that?

|  |
| --- |
|  |

It can be overwhelming and/or tedious to use cd to navigate to the directory that holds your code. So use the "Open in integrated terminal" shortcut to quickly navigate to your directory in VS Code. (You are still responsible for knowing how to use cd and the other commands.)

21. Fill in the chart below with descriptions of each of the commands.

|  |  |
| --- | --- |
| **Command** | **Description of what the command does** |
| **pwd** |  |
| **cd *path/to/folder*** |  |
| **cd ~** |  |
| **cd ..** |  |
| **ls** |  |
| **mkdir** |  |
| **touch / New-Item** |  |
| **python3 *filename*** |  |

See the bottom of the Software page on Canvas for links to videos and other resources for learning about the command line.

When you have finished this step, please call me over so I can see your progress and be prepared to demonstrate using the commands in the Visual Studio Code terminal.

# Reflection

Please answer the following questions individually, though you may discuss your answers with your group.

1. What confused you most about this activity?

|  |
| --- |
|  |

2. What questions do you have either about the activity directly or about related topics that came to mind while you did the activity?

|  |
| --- |
|  |

3. Do you feel like you achieved all of the learning objectives listed at the top of each part that you completed? If you feel uncertain about any of them, please list them here.

|  |
| --- |
|  |