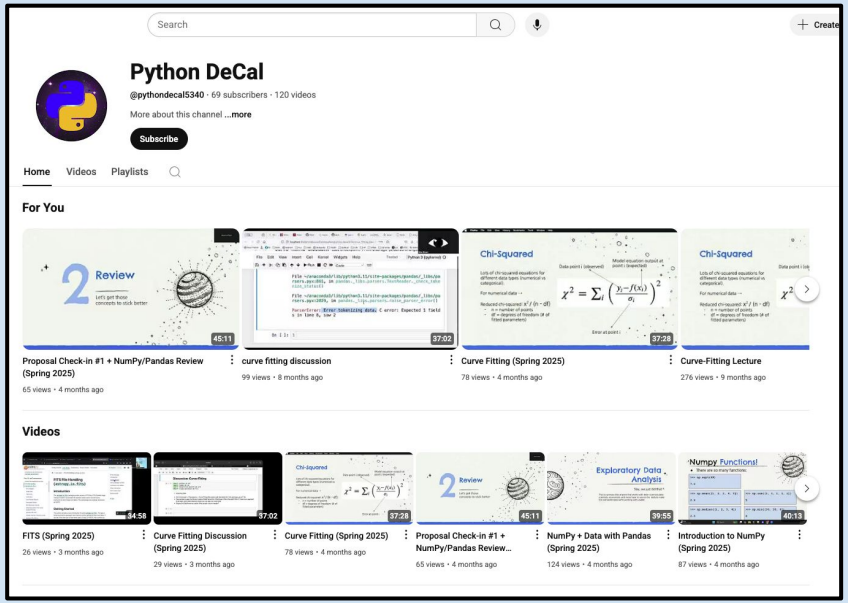
Welcome to the Python DeCal!

VS Code + Variables + Data Types + Command Line/Terminal

[Class 2] September 3rd, 2025

Lectures will be recorded

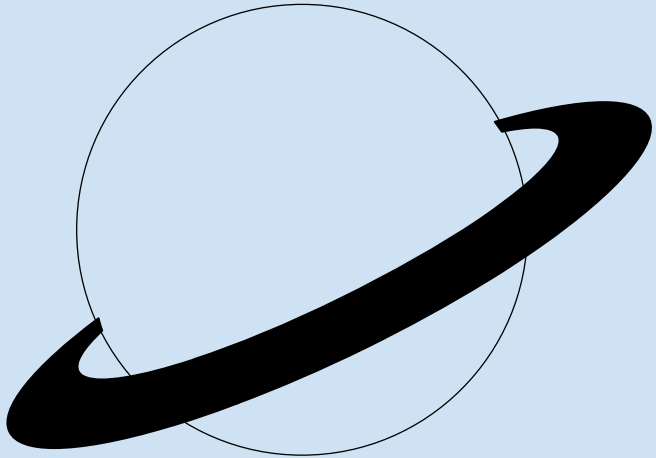


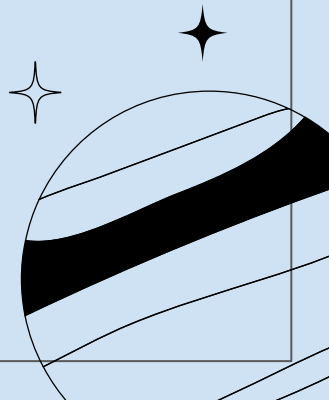
This class will be recorded and posted to our YouTube Channel!

Subscribe :)

Overview for Today!

1. Announcements
2. Super Quick Warm-Up
3. Demo
    a. VS Code (+ Comments)
    b. Variables + Data Types
    c. Practice Question?
    d. Command Line/Terminal
4. Lecture Engagement
5. Bonus Content (for homework help)





01

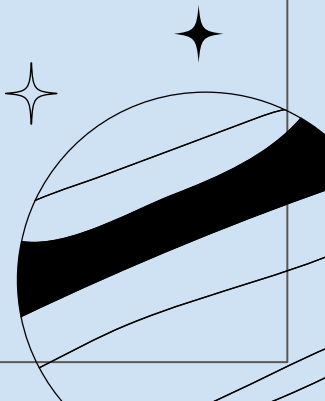
Announcements

:) :) :) :) :) :) :) :) :) :) :) :) :)

Announcements

- HW #1 was released on Monday, Jan. 26th
- HW #1 is due next Wednesday, Feb. 4th at 11:59 PM
- HW #1 late due date is Feb. 18th at 11:59 PM
  - Next, next, next Wednesday

- Office hours w2m has been posted on bCourses + Ed
  - Please fill this out by Friday EOD!
- Please finish up the installation guide if you haven't yet
  - Installation office hours throughout this week



02

Warm-Up

Let's do a little review

Super Quick Warm-up

Answer the following questions with a partner:

1) What days are homeworks released? When are they due?
2) Where are homeworks posted? Where do you turn them in?
3) When is the late due dates for homework?
4) What are lecture engagements? Where/when are they due?
5) Where are slides posted? Where are recordings posted?
6) What do you need to pass this class?
7) When are Final Project Presentations?
8) What are some ways to get help?

warm-up answers

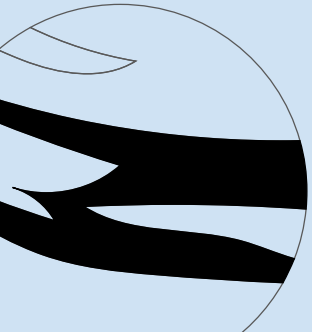
1) Released on Mondays. Due the following Wednesday
2) Posted on bCourses. Turn in on Gradescope.
3) Two Wednesdays past the due date.
4) Participation for class. Due on bCourses before the next class.
5) Slides posted on bCourses. Recordings posted on YouTube.
6) Overall grade of 70% and an attempt on all Final Project assignments.
7) Last week of instruction! (the week before dead week)
8) Office hours! Ed discussion, email, asking questions during class or after class, Google

03

Demo

Starting with VS Code

VS CODE

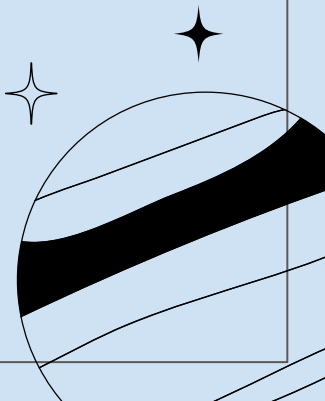


What is VS Code?

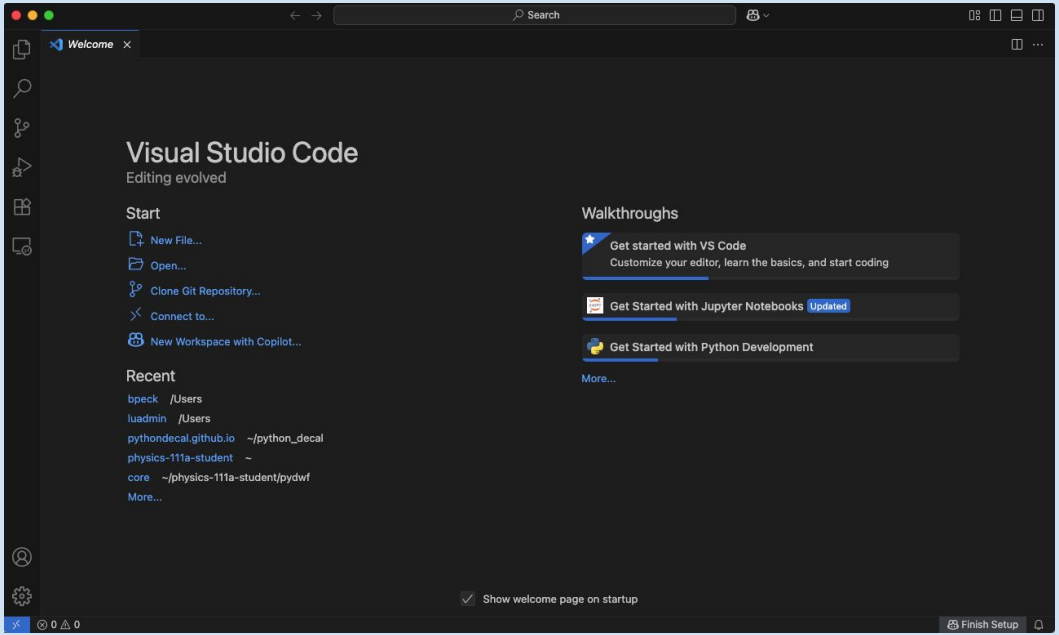
VS Code is one of the two main applications we will use to write and edit code this semester.

Why do we use it?

- It's open source (free)
- Lots of features and very customizable
- Works well with the terminal



OPEN VSCODE

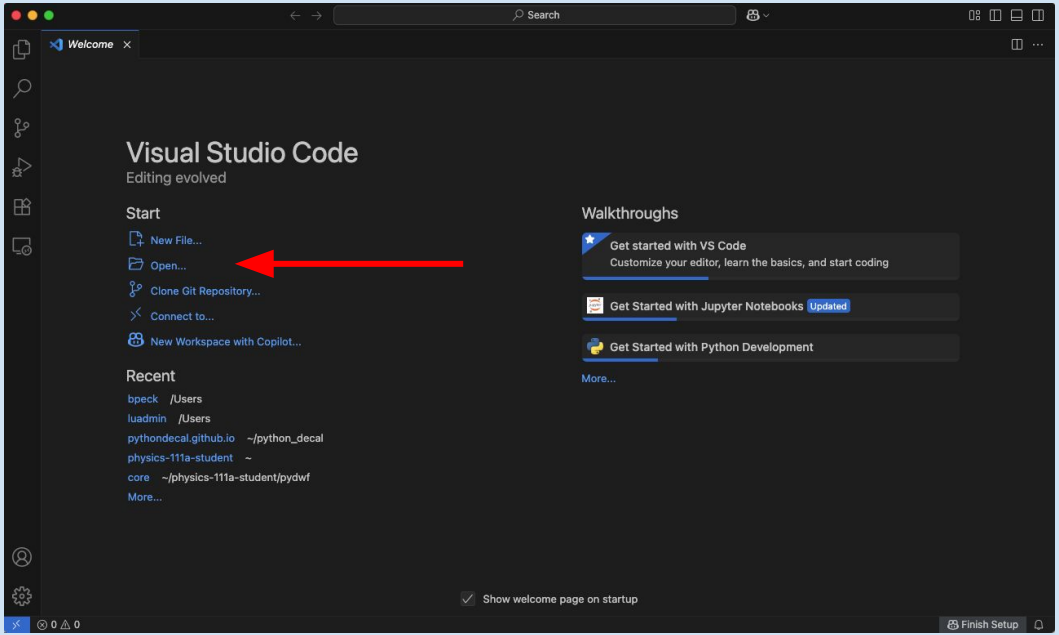


Search for VS Code under Applications or Downloads

Open VS Code

- This is what your screen should look like when its open

Open a New Folder in VS Code

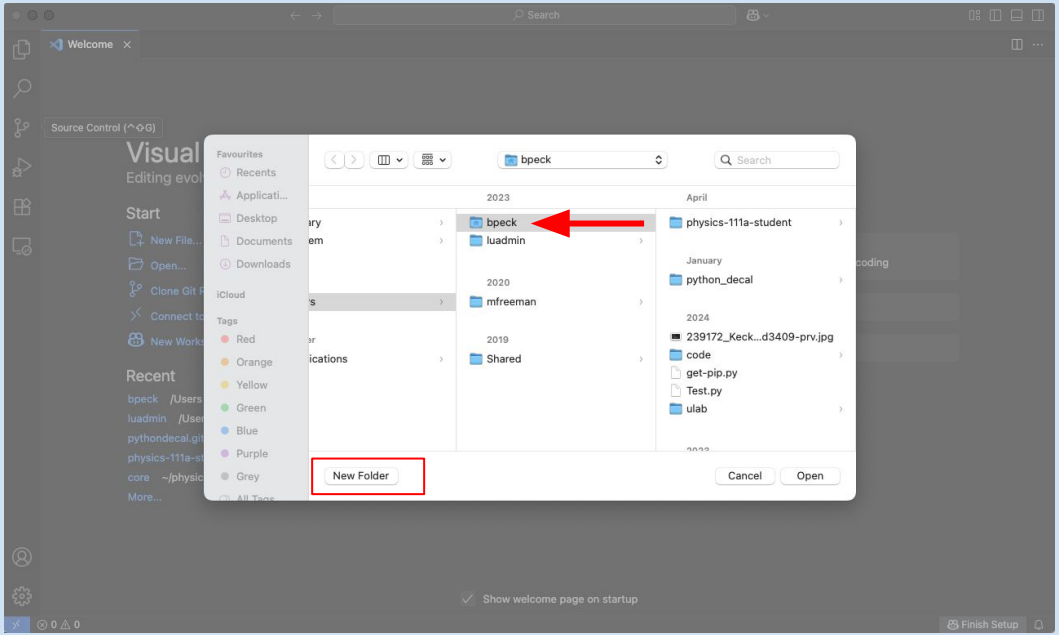


Click on Open...

We will be creating a new folder for this class

You will work primarily in this folder for the whole semester

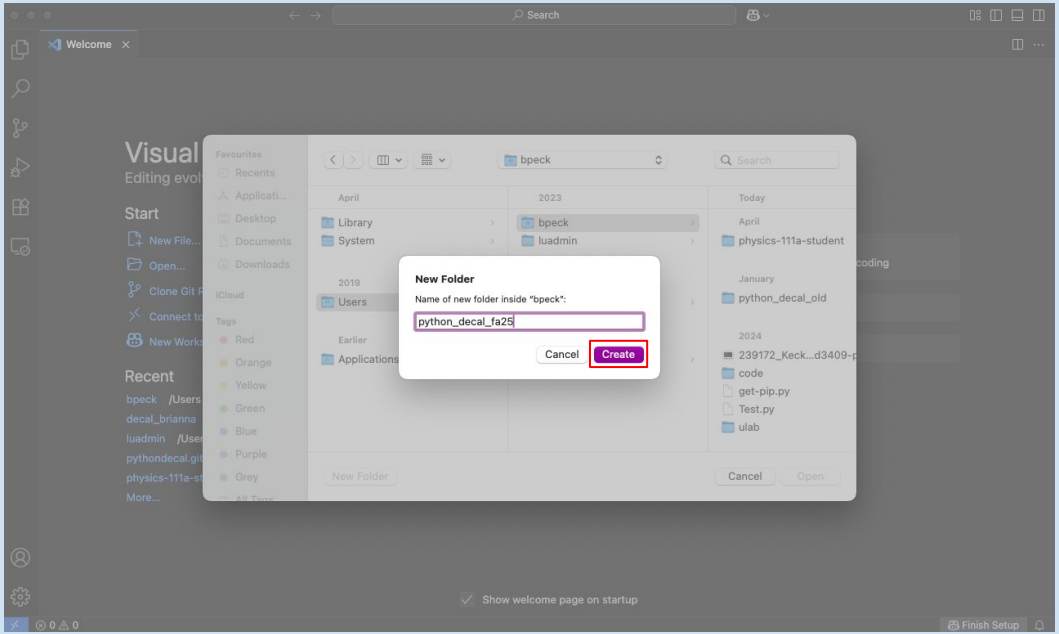
Open a New Folder (cont.)



Open a folder in the default location under your user

So I am opening a new folder under bpeck because that is my user

Open a New Folder (Cont.)

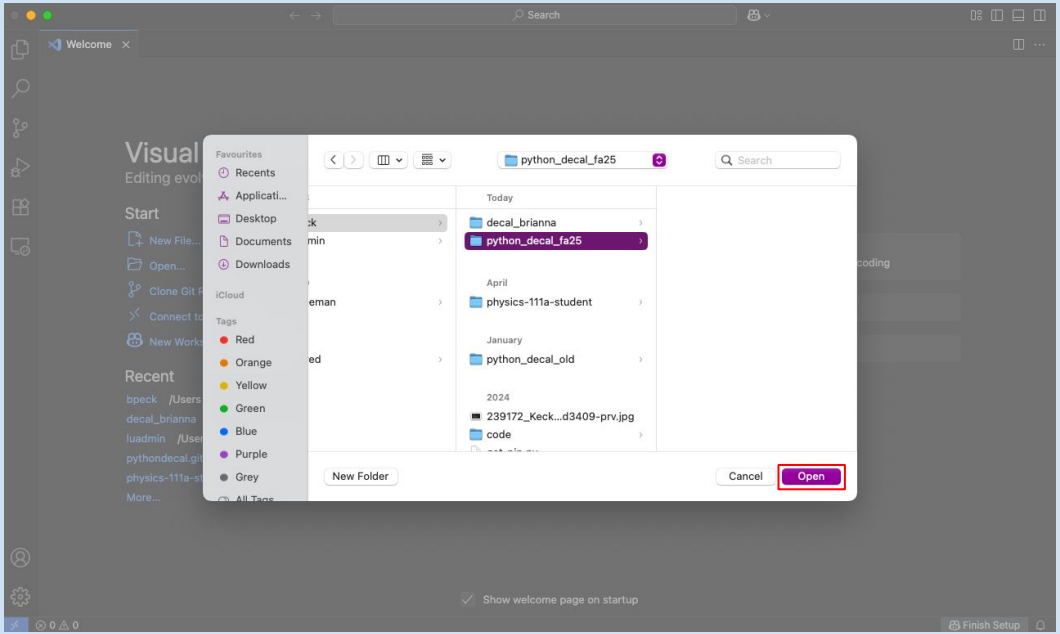


Call your new folder: python_decal_sp26

In this class, we will use folder and directory interchangeably

They mean the exact same thing

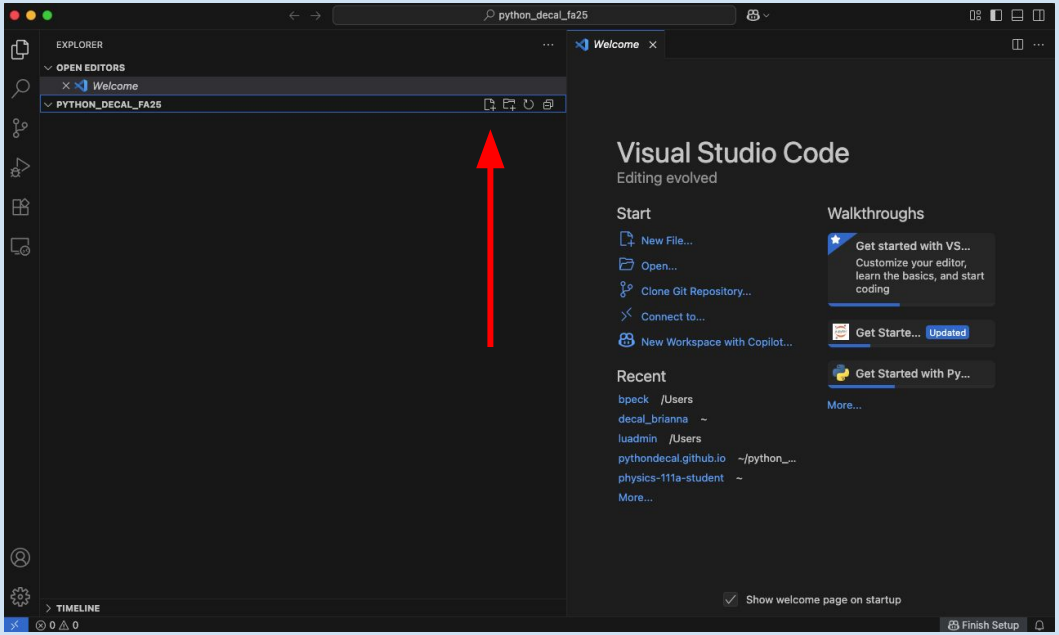
Open a New Folder (Cont.)



Open up your new python_decal_sp26 directory

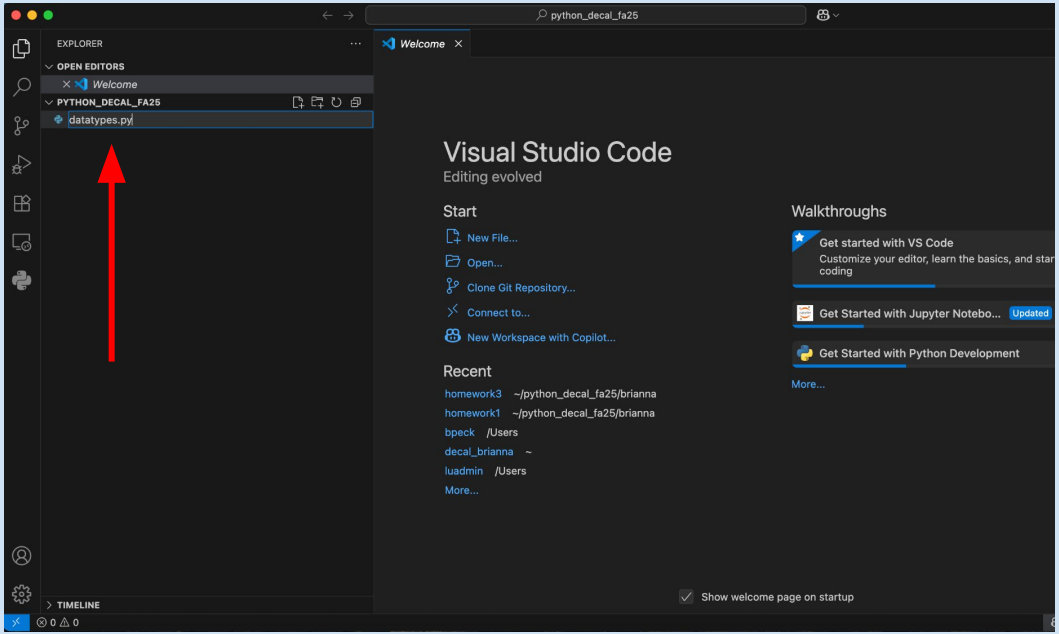
The last column should be completely empty since there is nothing in this folder yet.

create a new python file



To create a new file, click on the icon the red arrow is pointing to

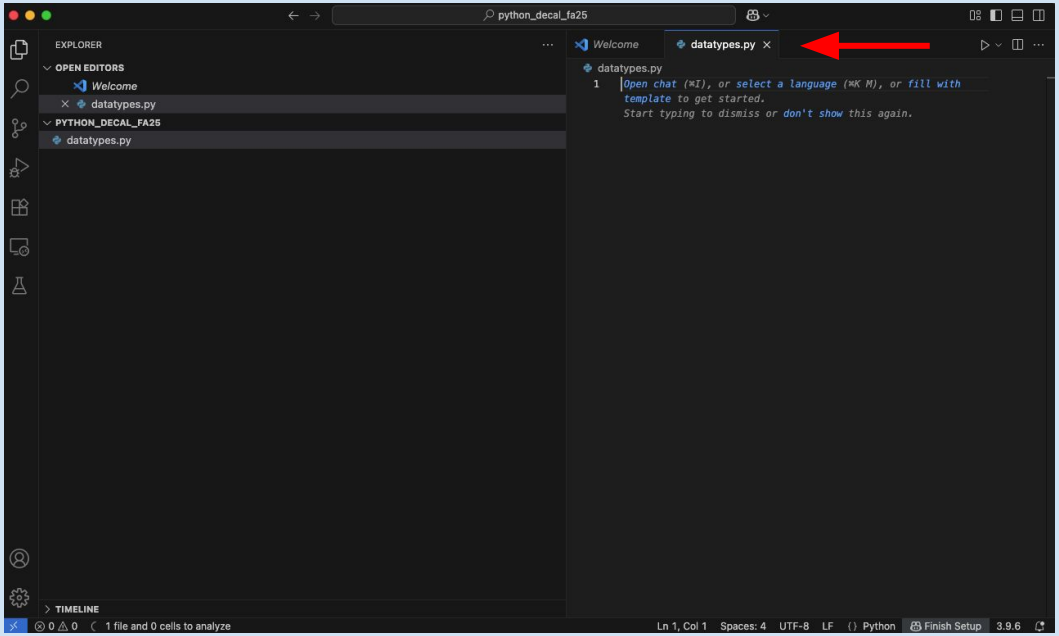
create a new file (cont.)



Under the new folder, name your new file: datatypes.py

The first topic we will cover today (besides VS Code) are data types

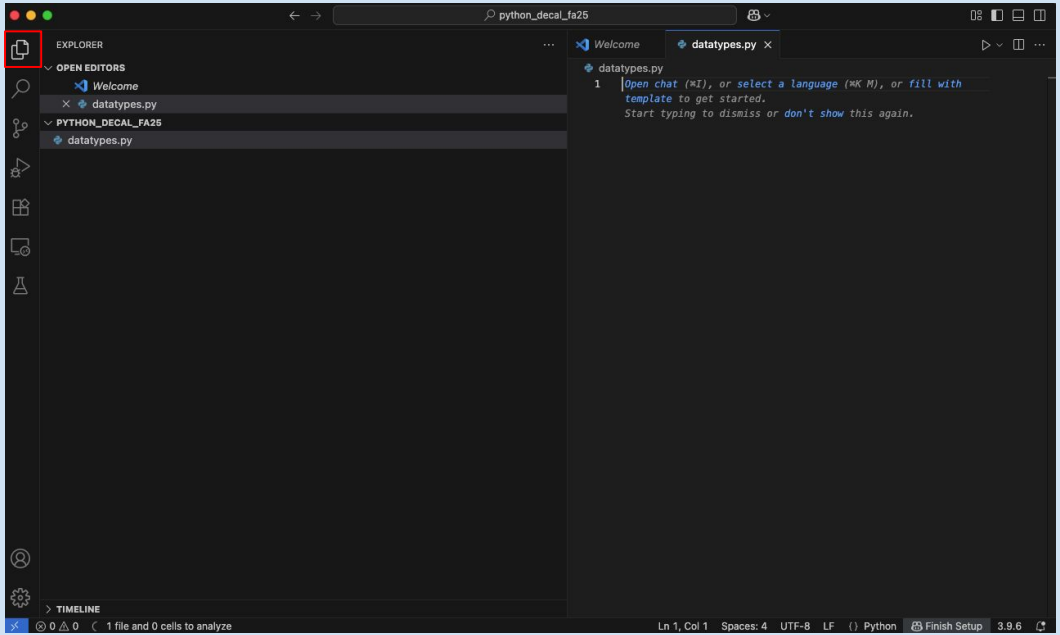
Create a new file (cont.)



On the right hand side of your window, a <u>new tab will appear</u>

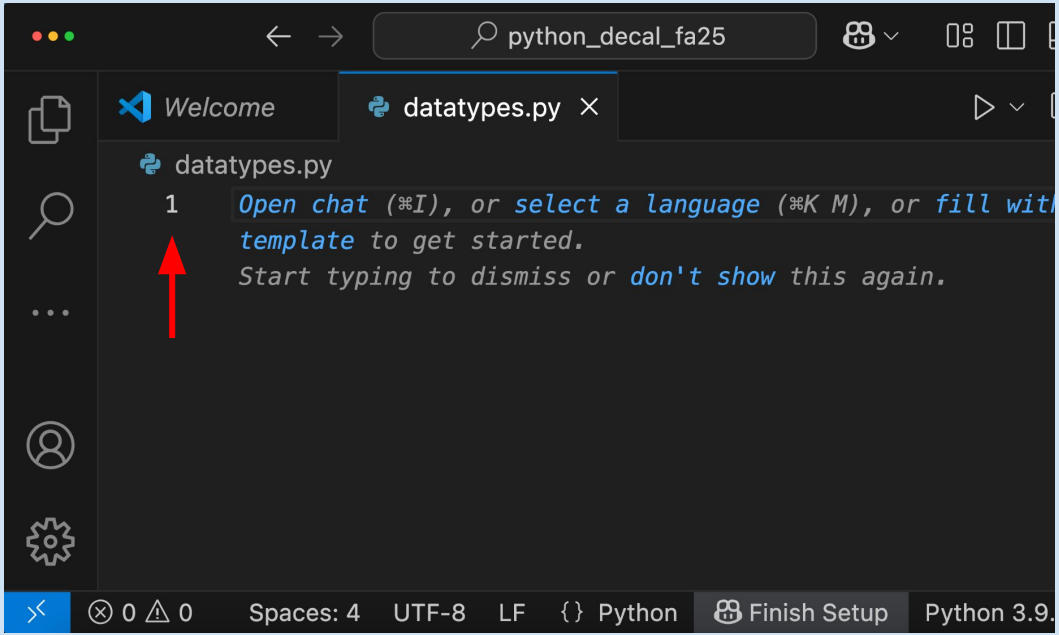
You should see the name of your file at the top of the window

create a new file (cont.)



To close the window on the left (Explorer), click the icon in the red box

write a python script

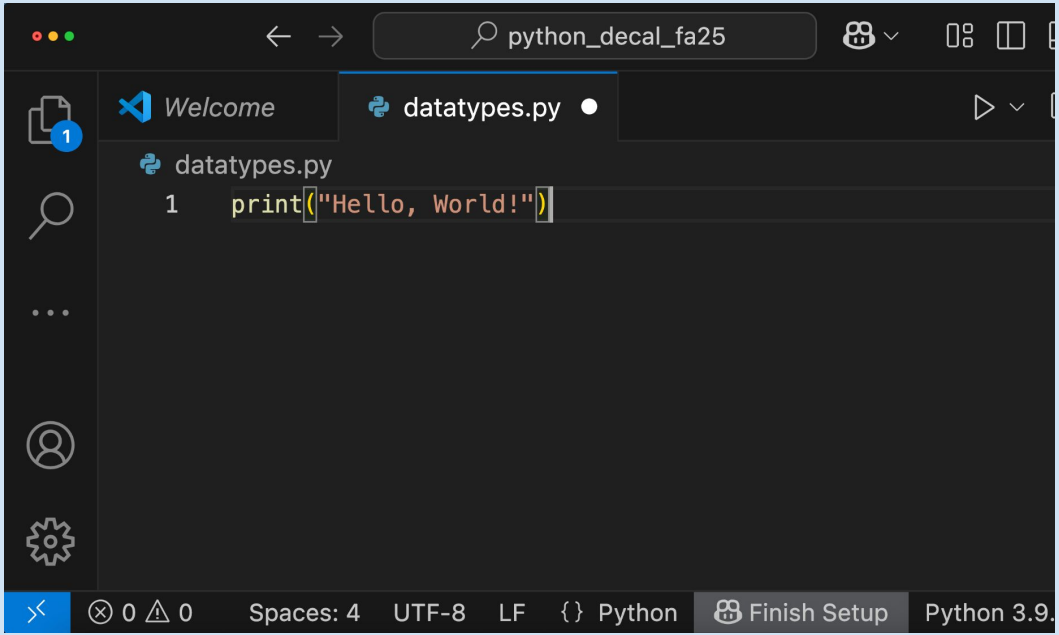


Script = a file containing python code that can perform tasks

Time to write our first python script!

Click on the line next to the number 1

Write a Python Script (Cont.)

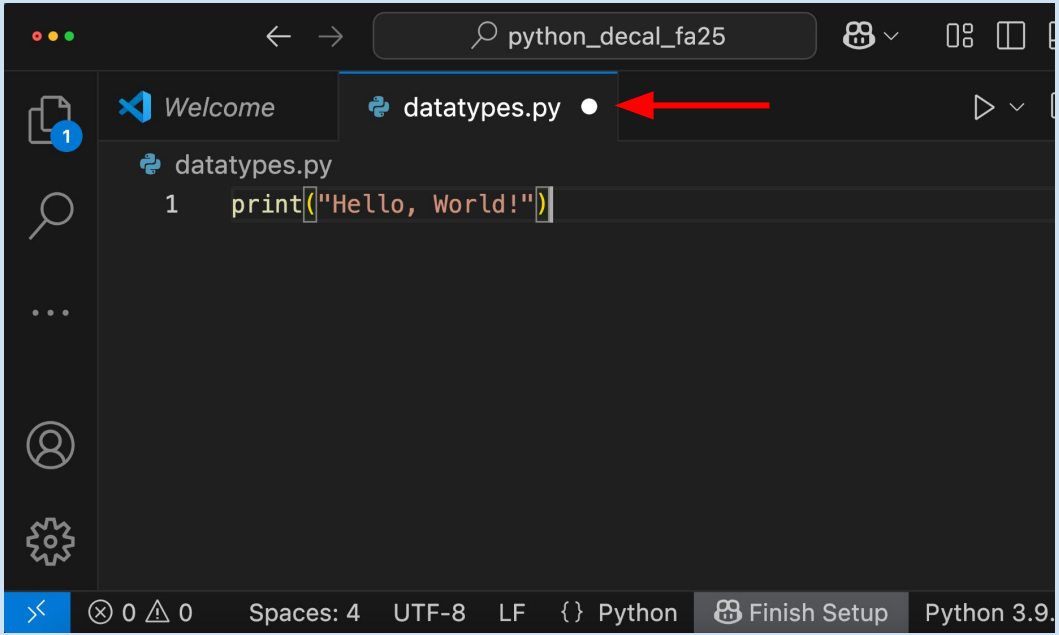


On line 1, type:
print(“Hello, World!”)

This is instructing Python to “print” the words:
Hello, World!

The words need quotation marks on both sides of the phrase

Write a Python Script (Cont.)

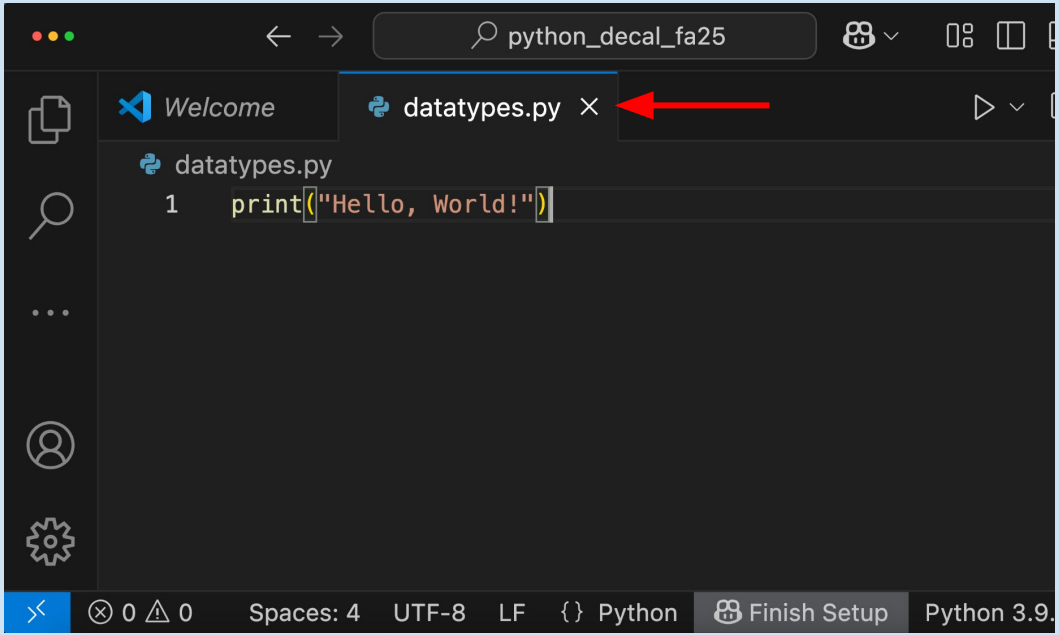


A white dot will now appear next to the top of your file

This means you have unsaved changes

Call: Ctrl + S or Cmd + S to save your changes

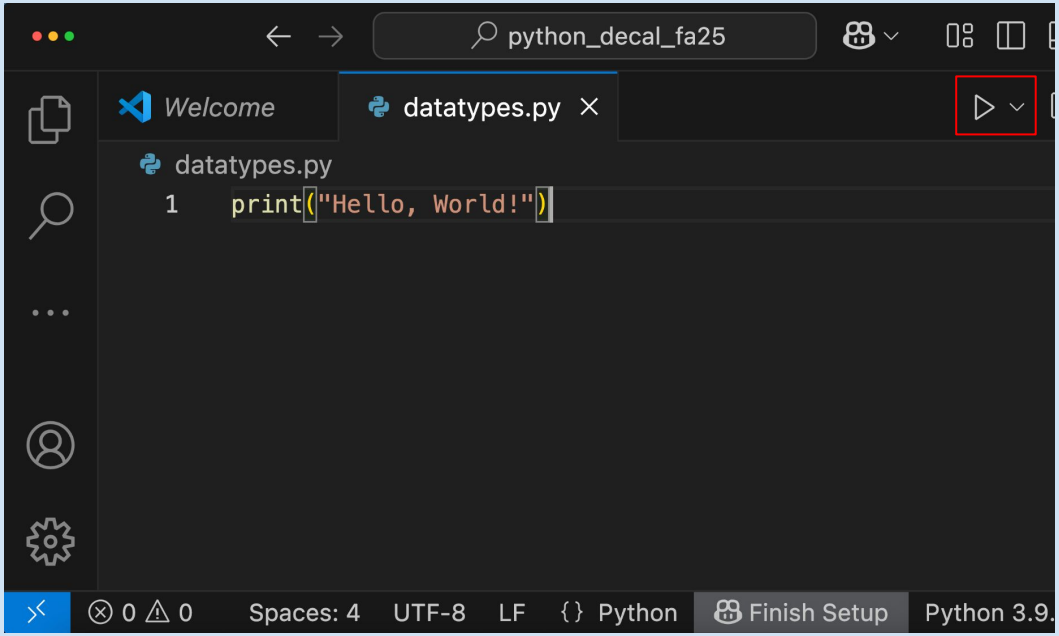
Write a Python Script (Cont.)



This is what the symbol will look like when you have saved your changes

It's always good to save your work after you make changes

Run a Python Script

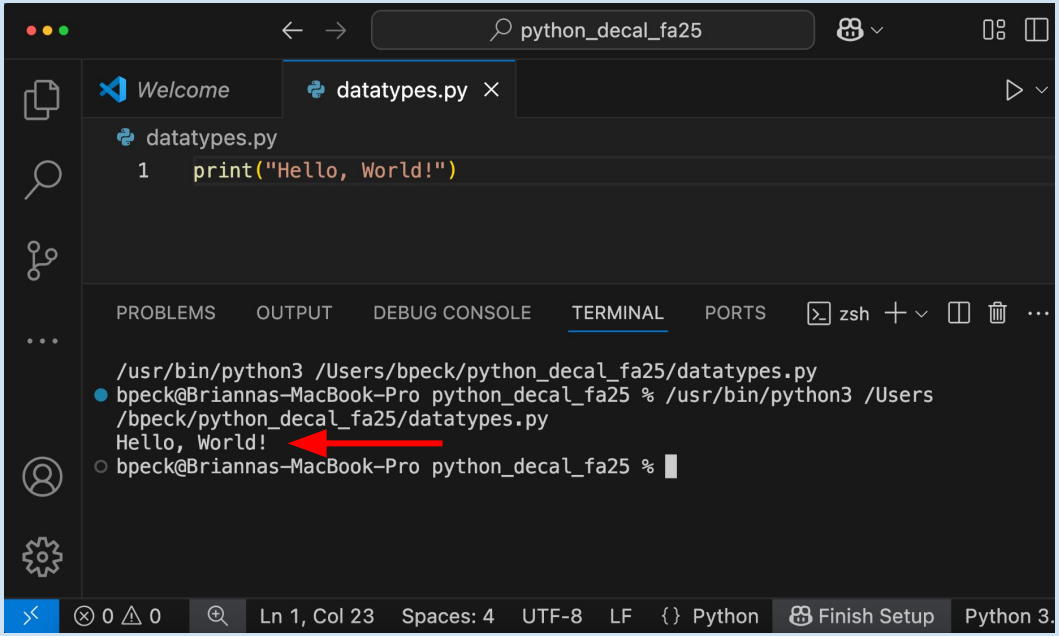


To run your python script, click on the arrow in the red box

Run a script = telling your computer to execute the program you created

Scripts go line-by-line

Run a Python Script

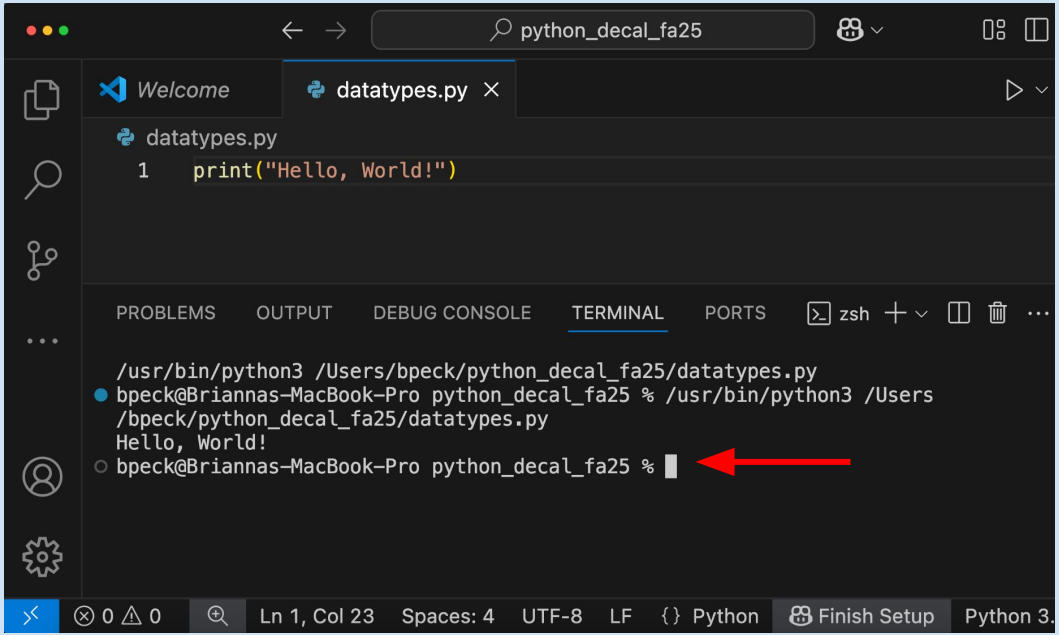


A new window will appear on the bottom of your screen

You should see the words: Hello, World!

This window is called the terminal, it shows the output of your code

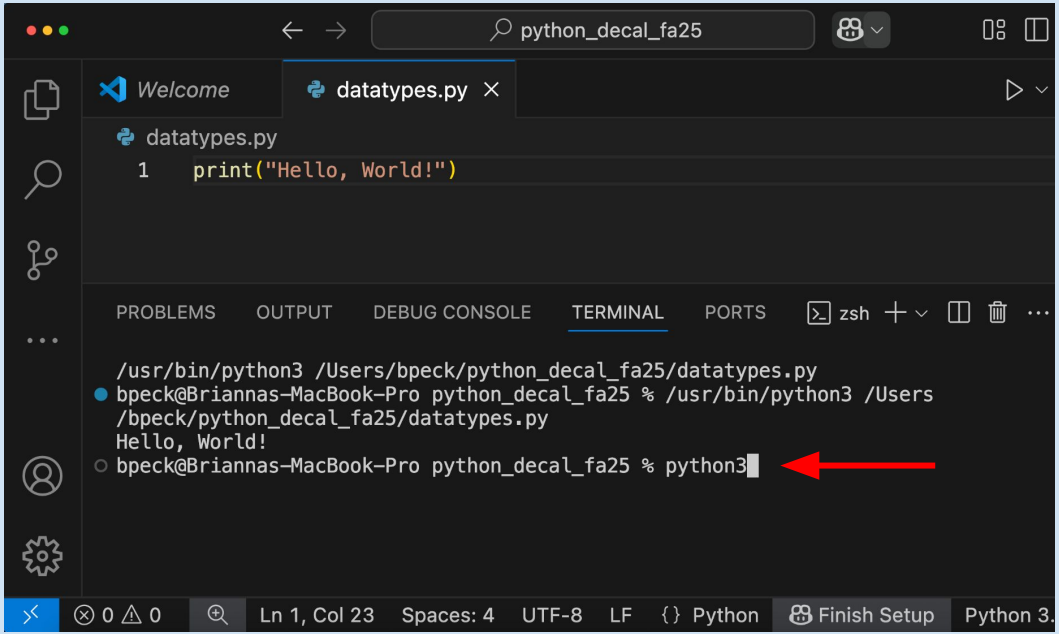
Run a Python Script



VS Code allows you to access your files and the terminal at the same time

This percentage % symbol, means that your terminal is awaiting your next command.

Python Directly in Terminal

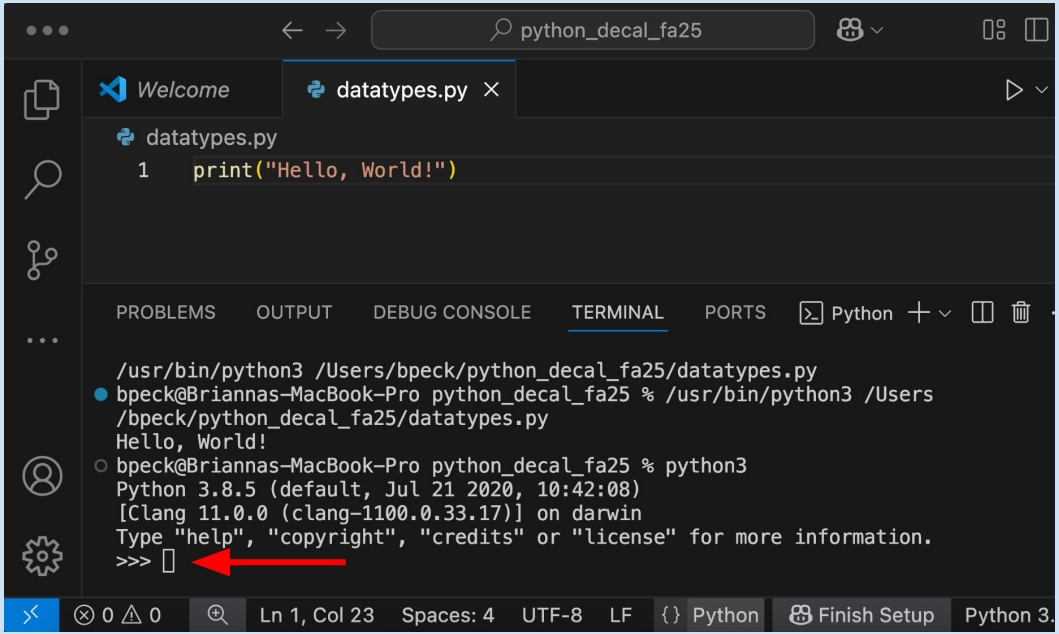


In addition to running scripts, you can type Python code directly on your terminal

Open Python by typing: python3 into your command line

Then press Return/Enter

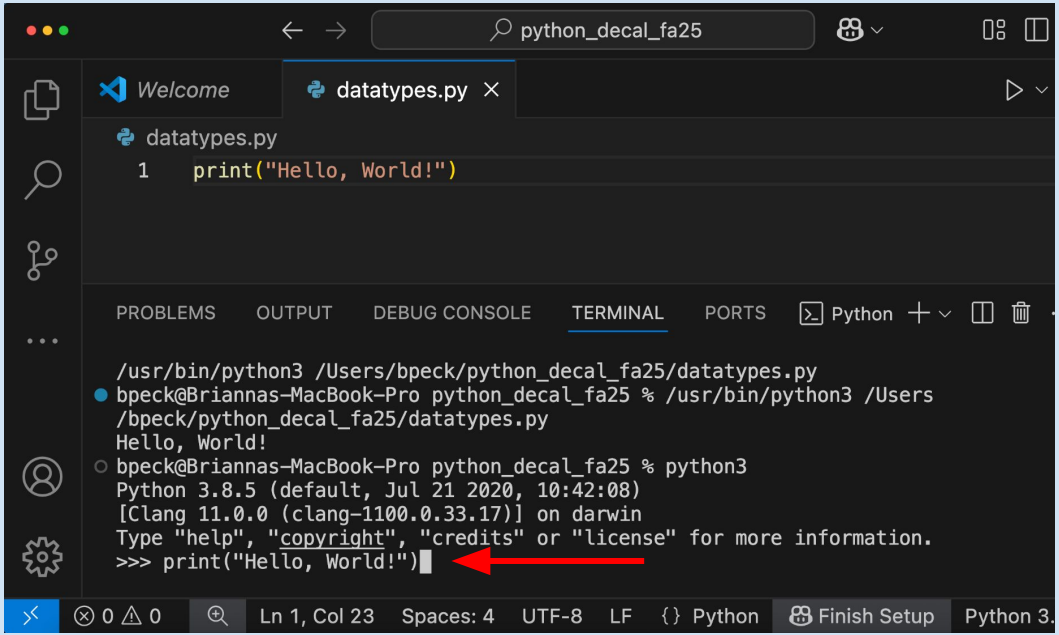
Python Directly in Terminal



Now your terminal now "speaks" in the Python language instead of zsh/bash!

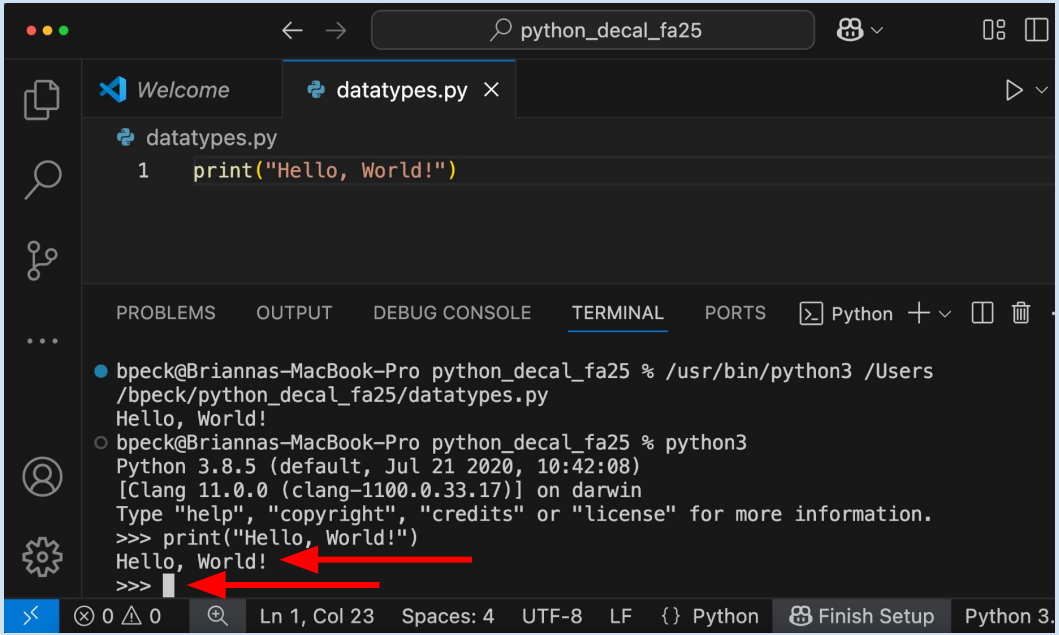
These three arrows >>>, mean the terminal is speaking in Python

Run Python in the Terminal



Next to those three little arrows, type the same statement as above: print(“Hello, World!”)

Run Python in the Terminal

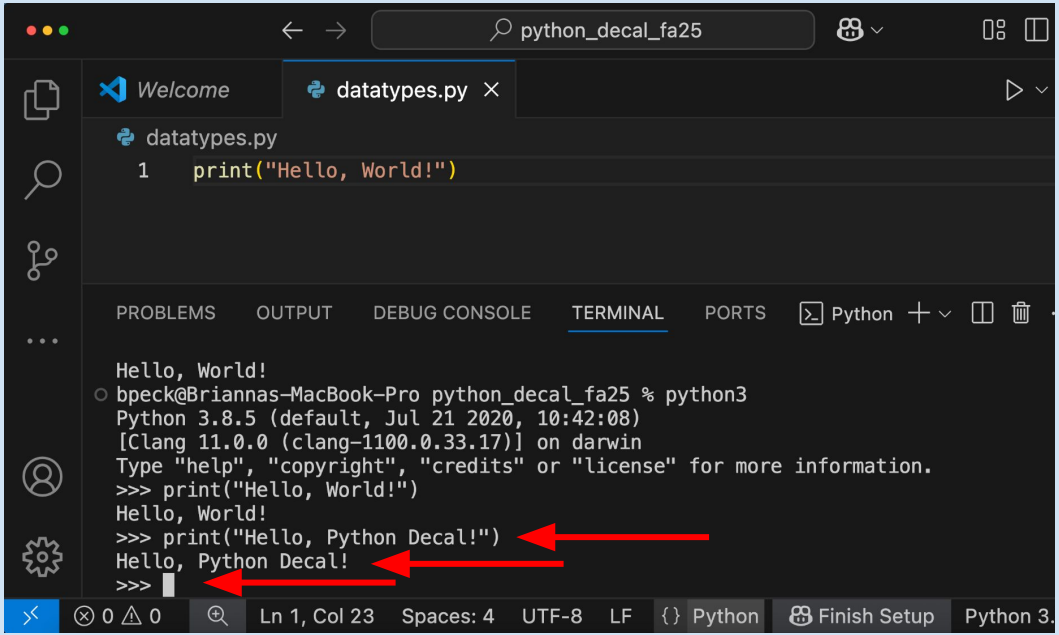


Once you hit Return/Enter, you should see another line of:

Hello, World!

Then three more arrows will appear, awaiting your next command

Run Python in the Terminal

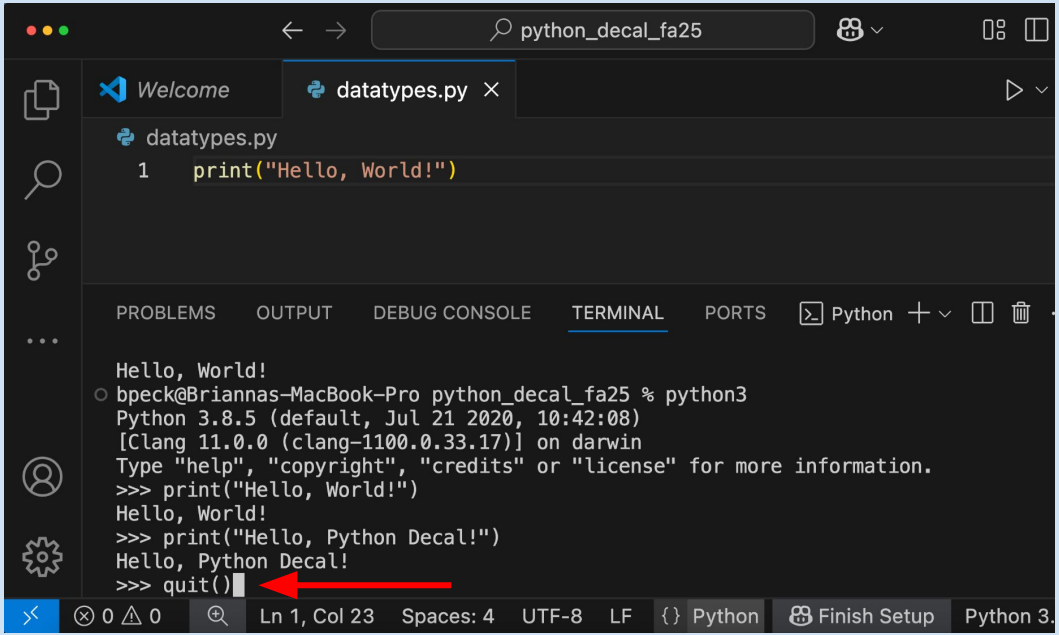


Let's practice one more time, by typing: print("Hello, Python Decal!")

Then run the code!

Once your code is done, the >>> will appear again

Quit Python in the Terminal

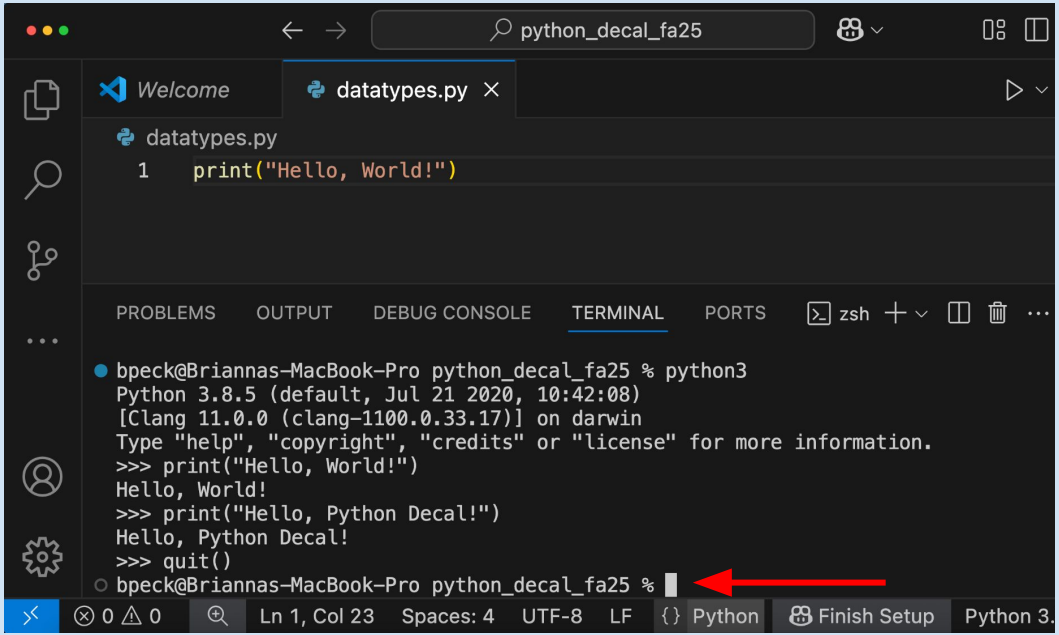


To go back to your "zsh/bash" speaking terminal, type:

quit()

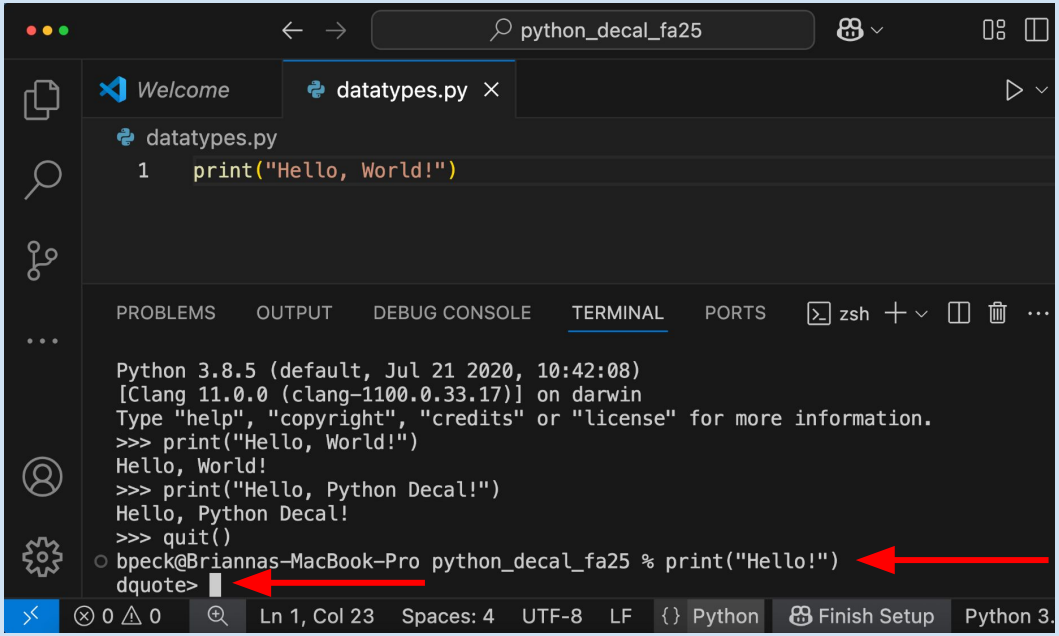
This will close your in-terminal Python session

Back to normal terminal



You know you are back to the regular terminal when you see the percentage % symbol again

Terminal no speak Python



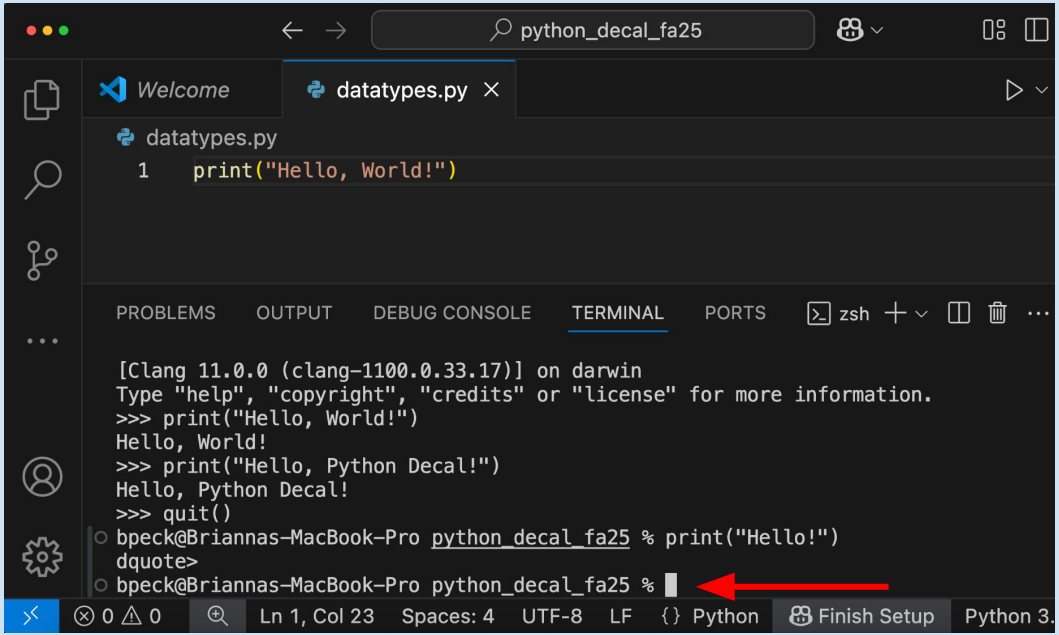
Now, if you try to type in Python, the terminal won't understand

For example:

print("Hello!")

Now you get a weird dquote>, because the terminal is confused

Terminal no speak Python



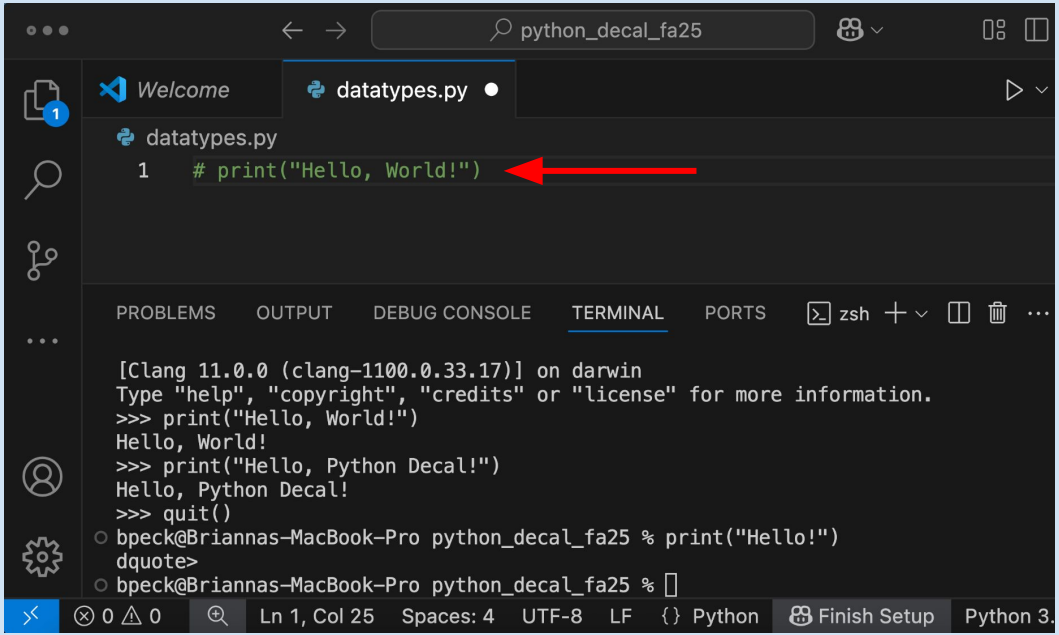
To get back to the normal terminal again, call:

Ctrl + C

This will pop back up the percentage % symbol, which awaits another command

comments

COMMENTING OUT CODE

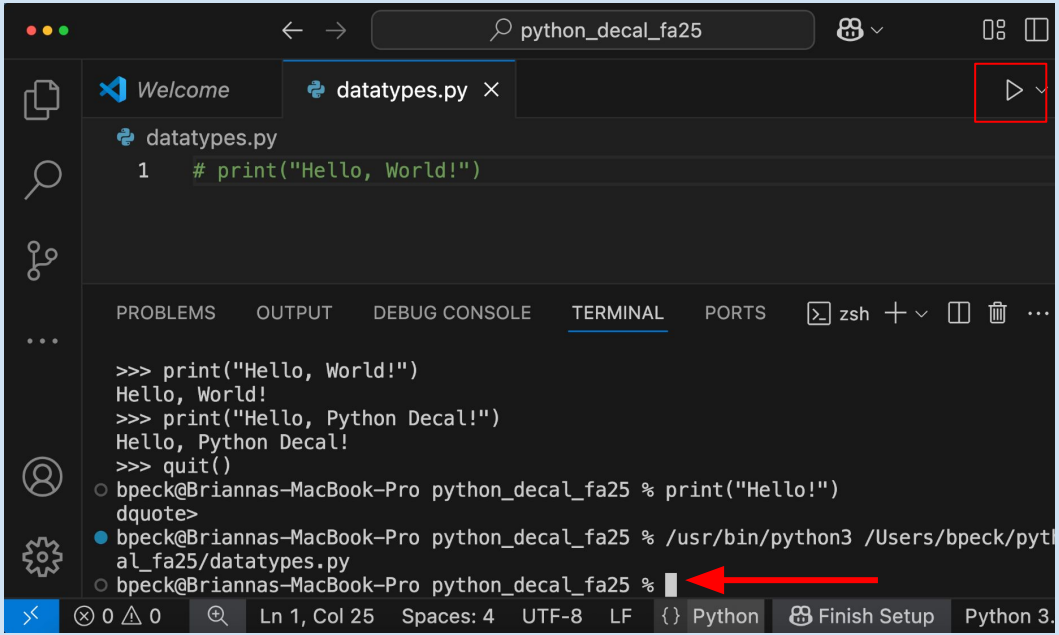


If you want Python to ignore a line of your code, you use comments

To make a comment, add a pound # symbol at the front of the code you want commented out

The text turns green

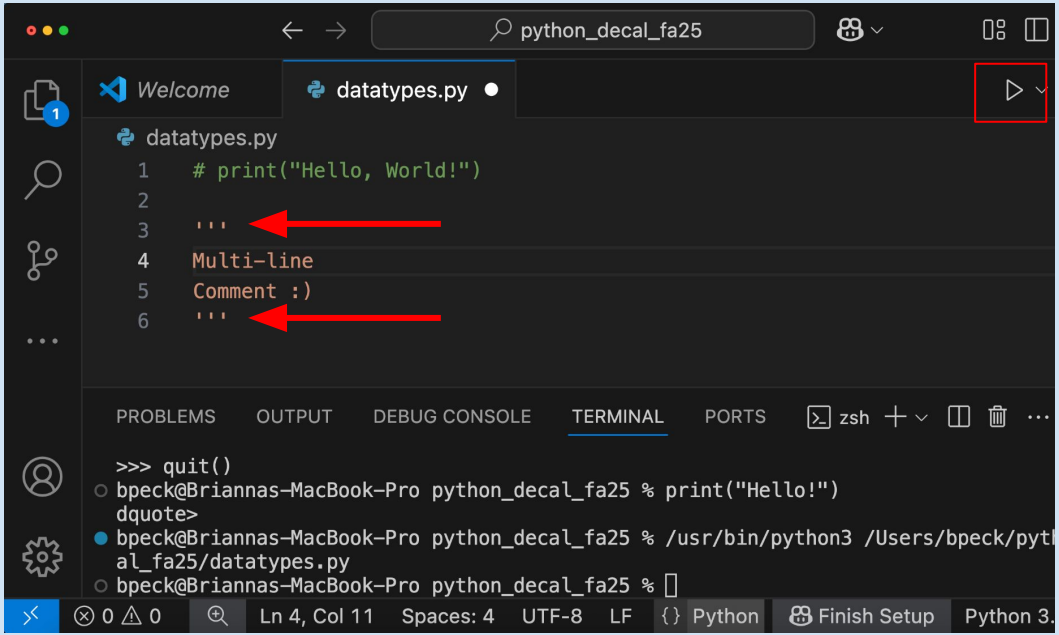
COMMENTING OUT CODE



If we press the arrow at the top of the window, now the Terminal will not return anything

Comments are for human eyes only, they are extremely useful! More on that later

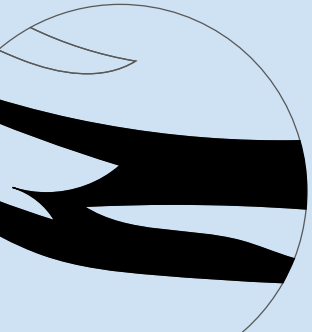
Multi-line comments



If you want to make a really large comment.

Surround your code with ‘‘‘ at the top of the code and ’’’ at the bottom

variables + data types

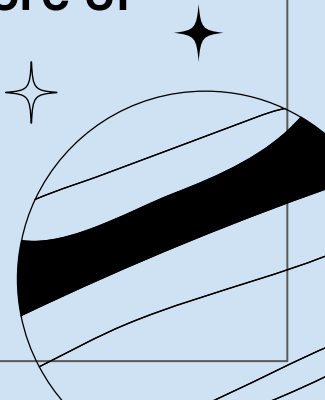


Data Types Overview

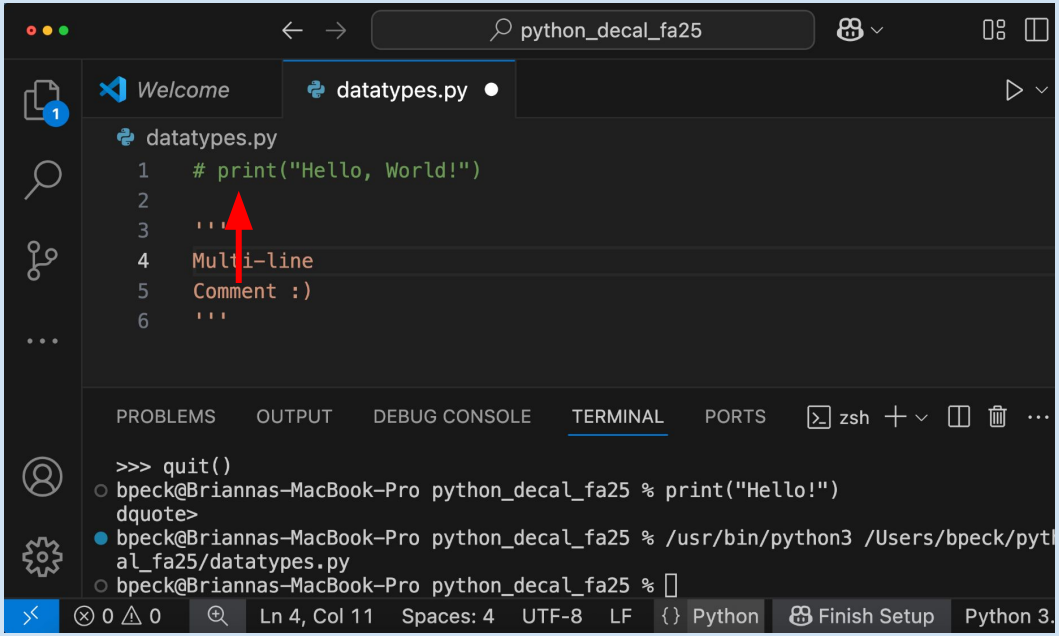
In the demo, we will cover three different data types today:

- Strings = words or text
- Integers = whole numbers
- Floats = fractional numbers or numbers with decimals

There are a ton of data types in Python, but you discover more of them in this week's homework.



Python Syntax

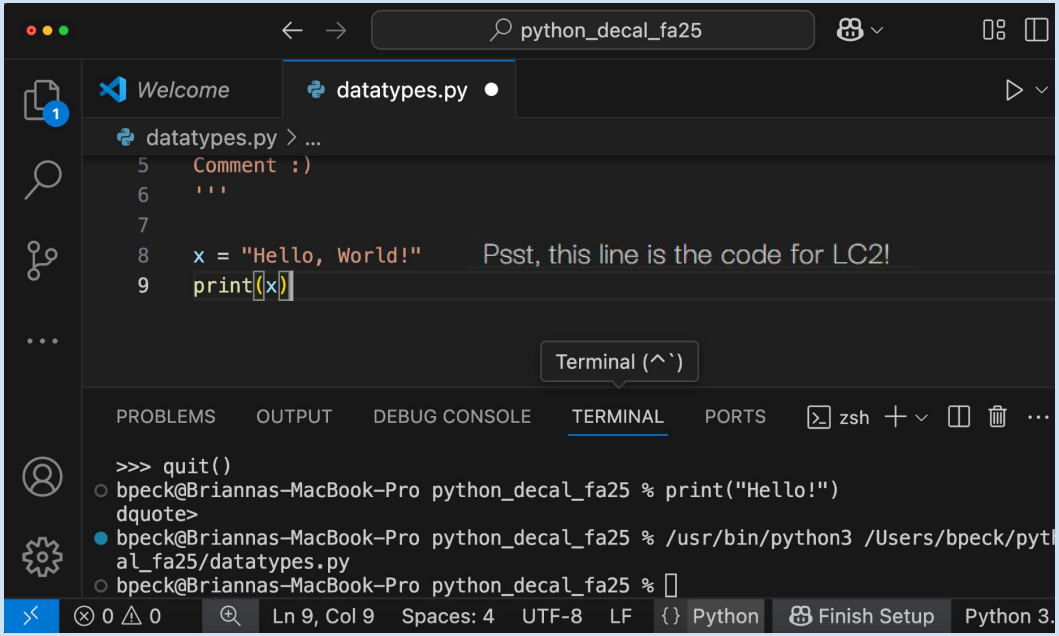


All you have learned about Python so far is just the print() function

Function = reusable code

Calling the built-in print() function tells Python to output the phrase you wrote

creating a variable



To create a variable in python, use the equals = symbol

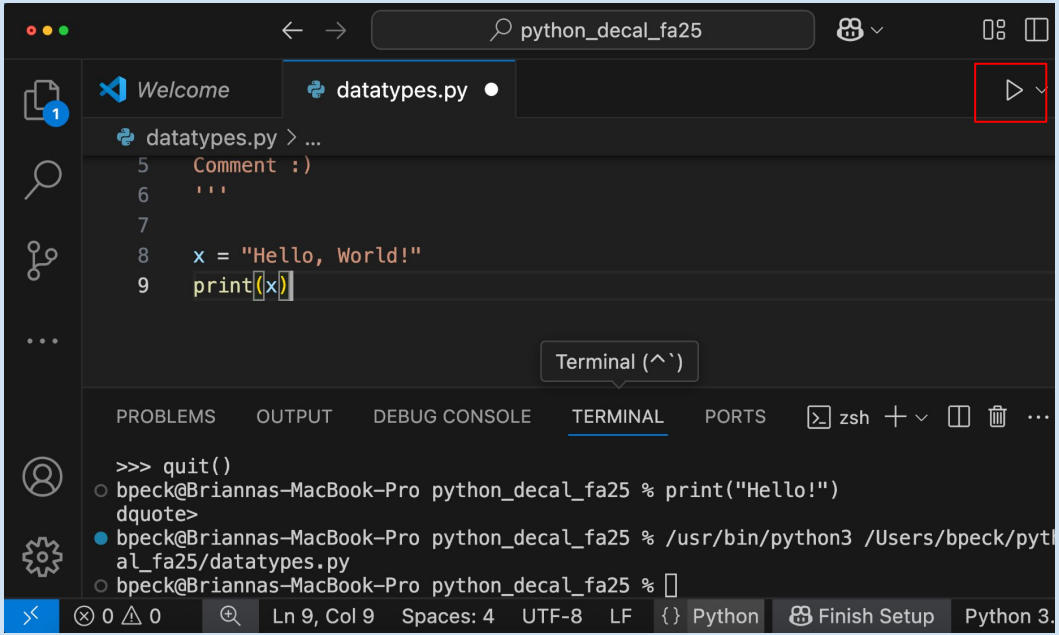
Variable = holds information in Python

Type:

x = “Hello, World!”

print(x)

creating a variable

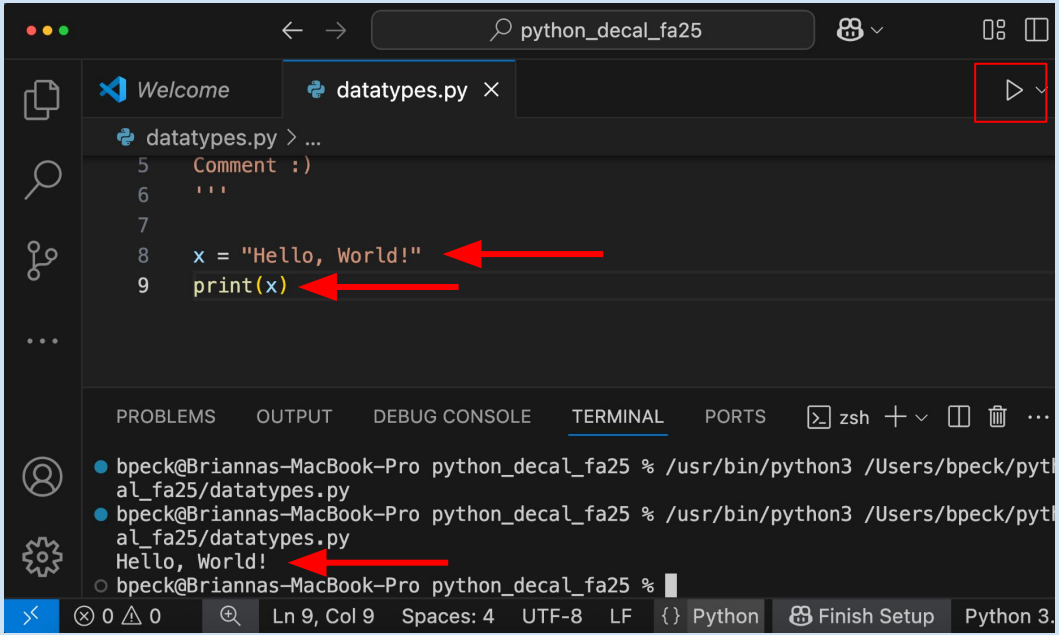


Overview: we assigned the variable x the words: Hello, World!

Then we are telling Python to print our variable

Click the arrow to run

creating a variable

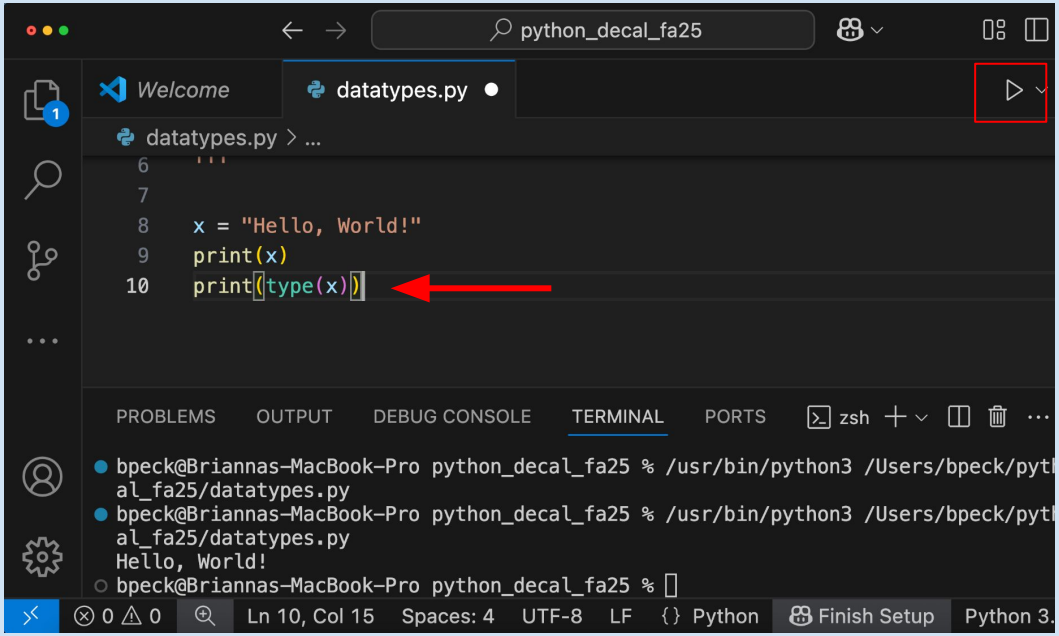


Overview: We assigned the variable x to the words: Hello, World!

Then we are telling Python to print our variable

Click the red arrow to run

Data Type of a Variable



To check the data type:

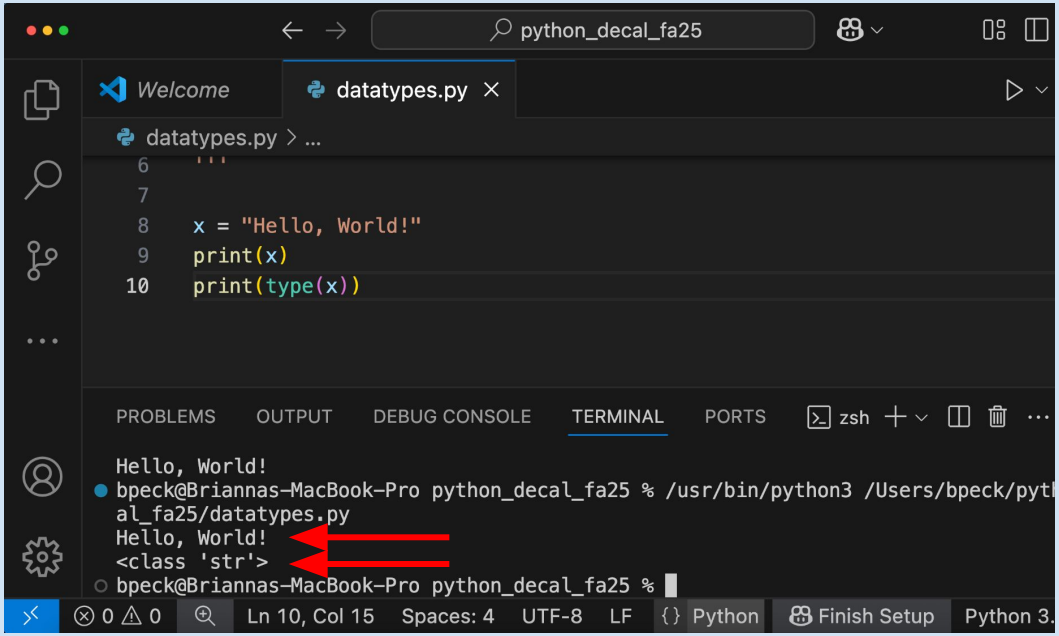
print(type(x))

Now we have covered two built-in functions:

- print() = shows output
- type() = shows type

Click the arrow in the boc

Data Type: String



The terminal will now output two different lines

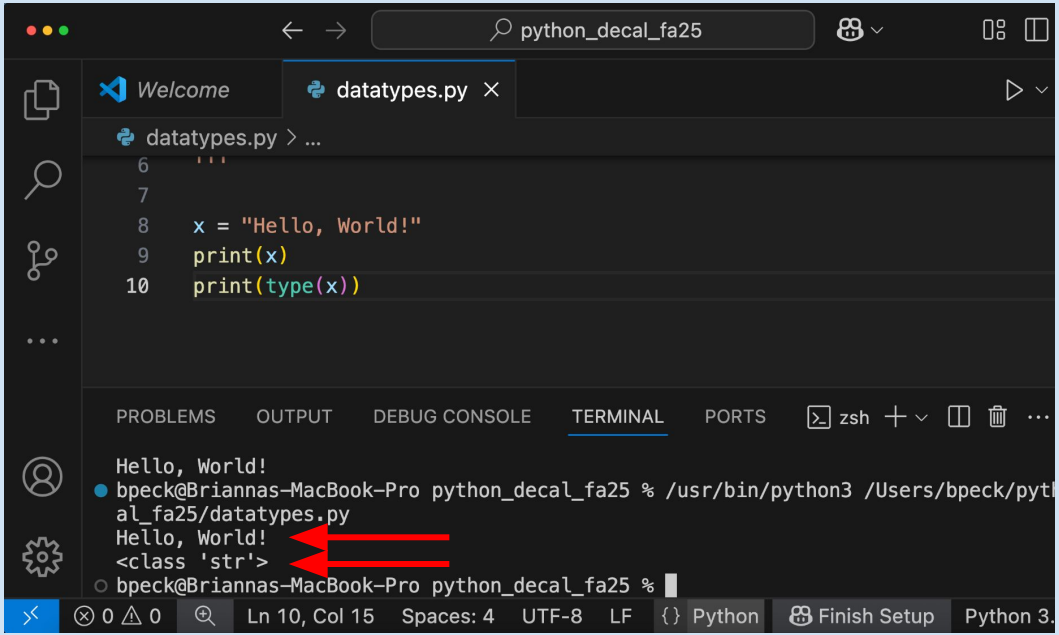
1st: “Hello, World!”

- Original statement

2nd: <class ‘str’>

- str, stands for string
- Ignore the word “class” for now

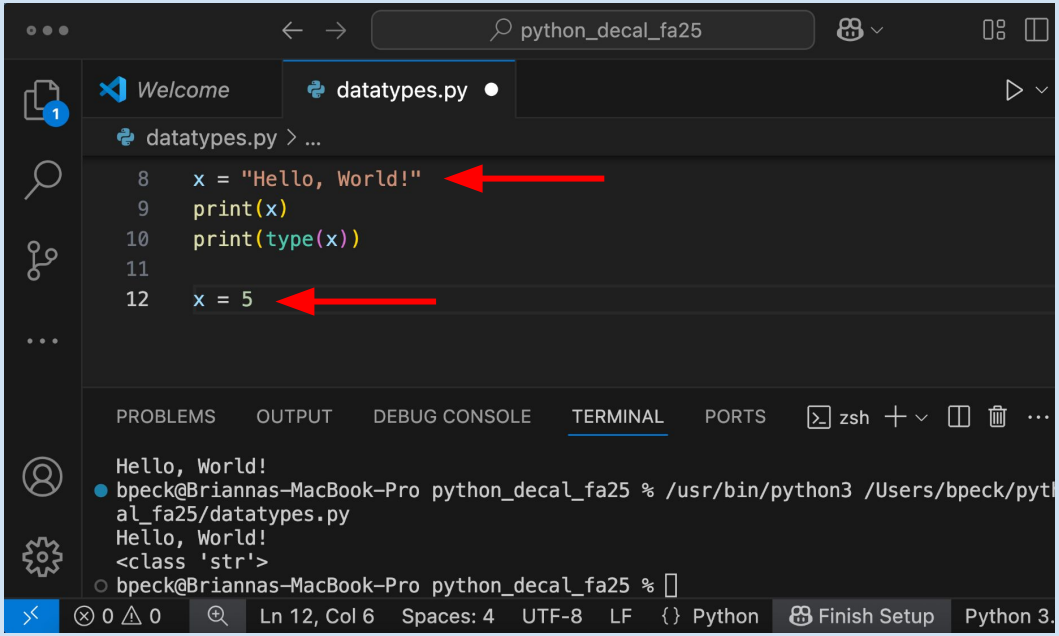
Data Type: String



String = a data type that holds text. That's it!

You can't do a lot of operations on strings, but they are very helpful for figuring out what your program is doing

Use Different Names



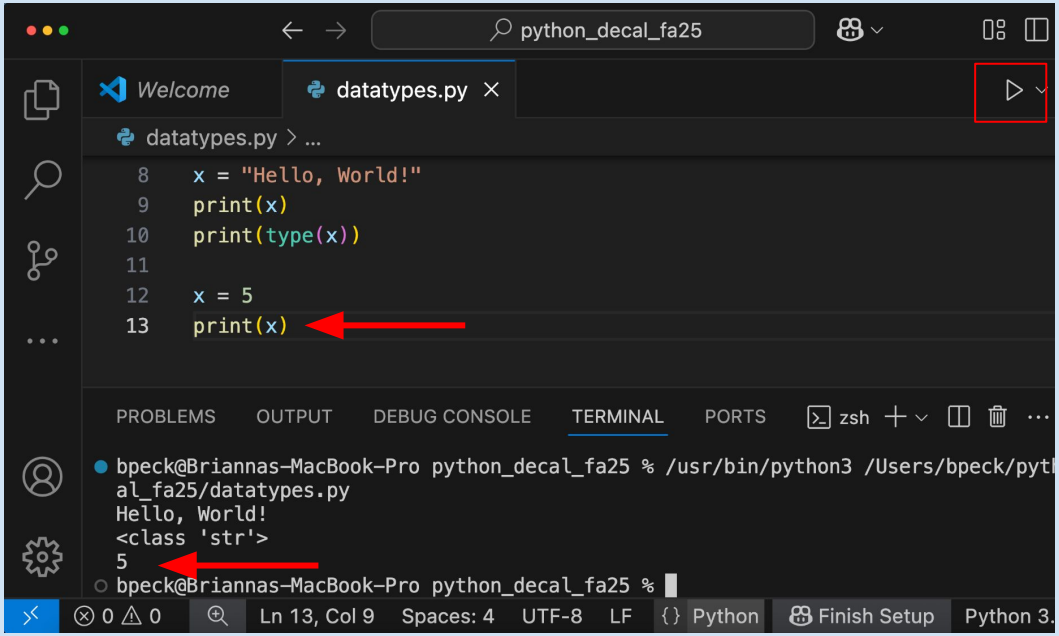
Note: If you reuse the same variable in Python. It will override the previous entry!

Notice how we have:

- x = “Hello, World!”
- x = 5

What happens if we call print(x)? Discuss.

Use Different Names

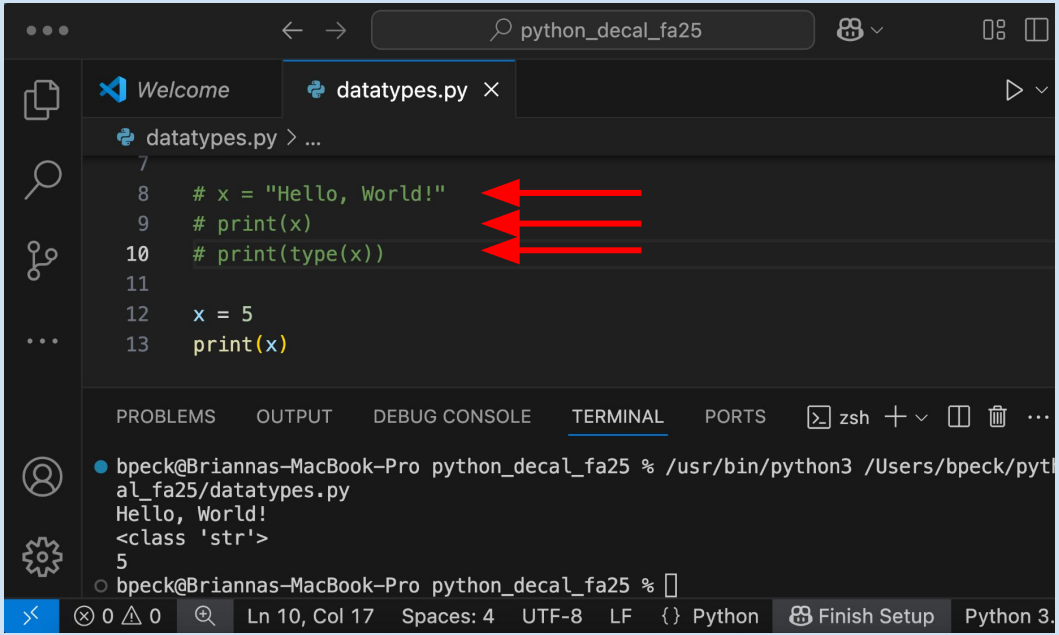


Type: print(x)

Instead of returning "Hello, World!" like the first x, Python now returns 5!

Just keep this in mind, it's good to use different variable names

comment out code

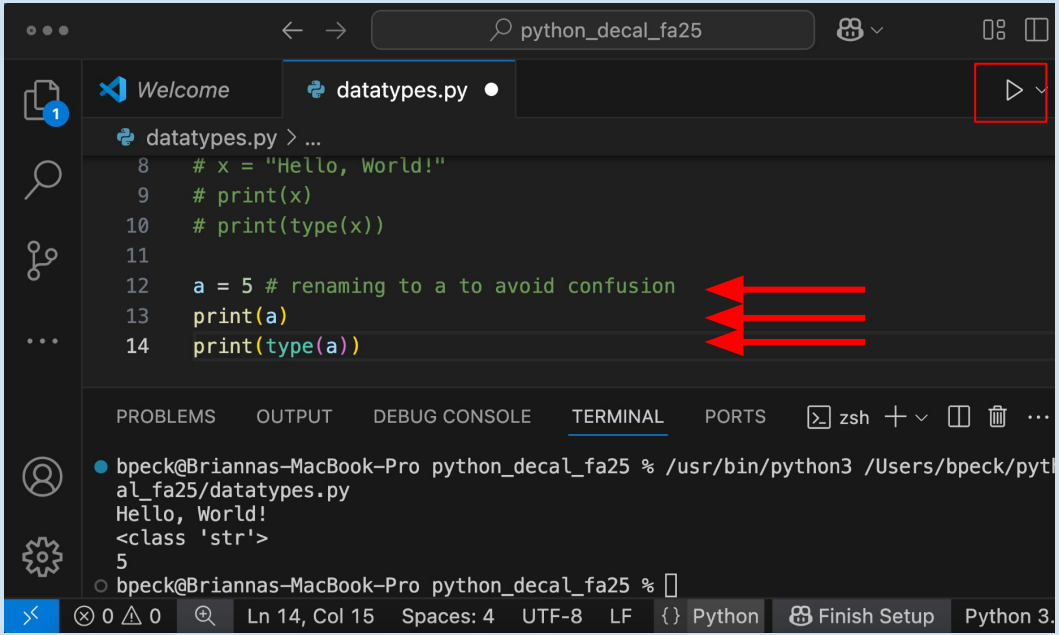


To avoid multiple outputs in the terminal as we move on... comment out the top chunk of code

A short cut is highlight the text you want to comment, then call:

Cmd + Q

Data Types: Integer

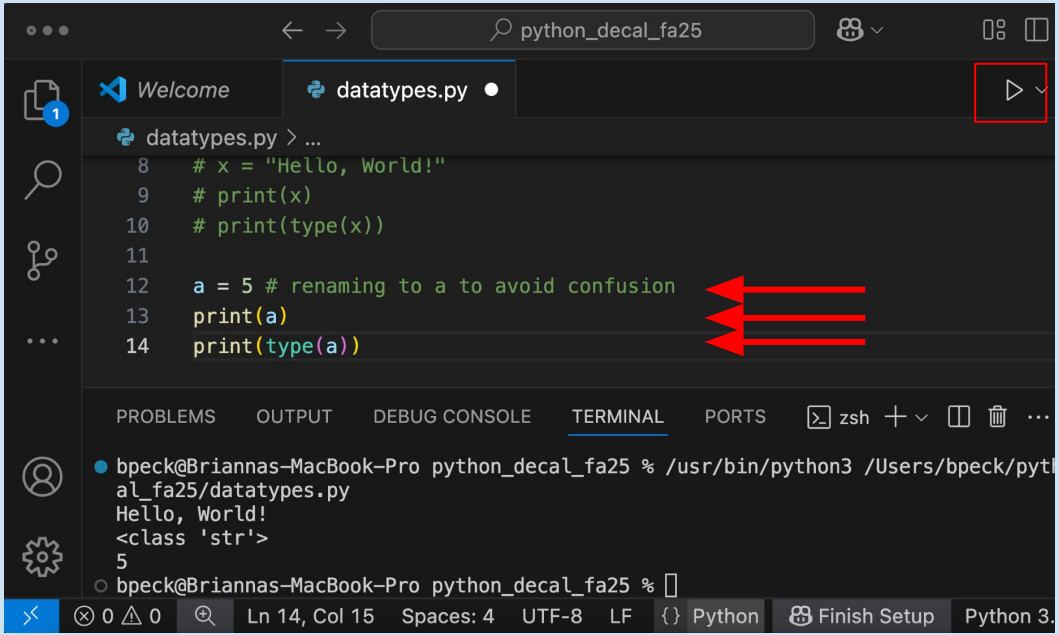


To avoid confusion, let's rename the second "x" to a

Next let's print a and print what type of variable it is

Click the arrow in the box

Data Types: Integer

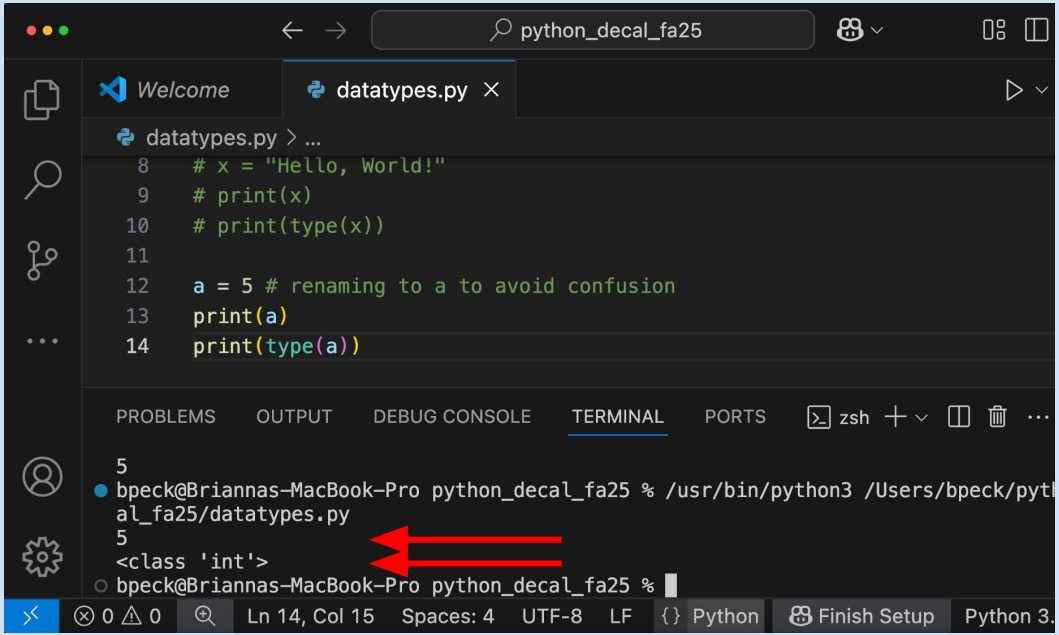


To avoid confusion, let's rename the second "x" to a, we can add a comment

Next let's print a and print what type of variable it is

Click the arrow in the box

Data Types: Integer

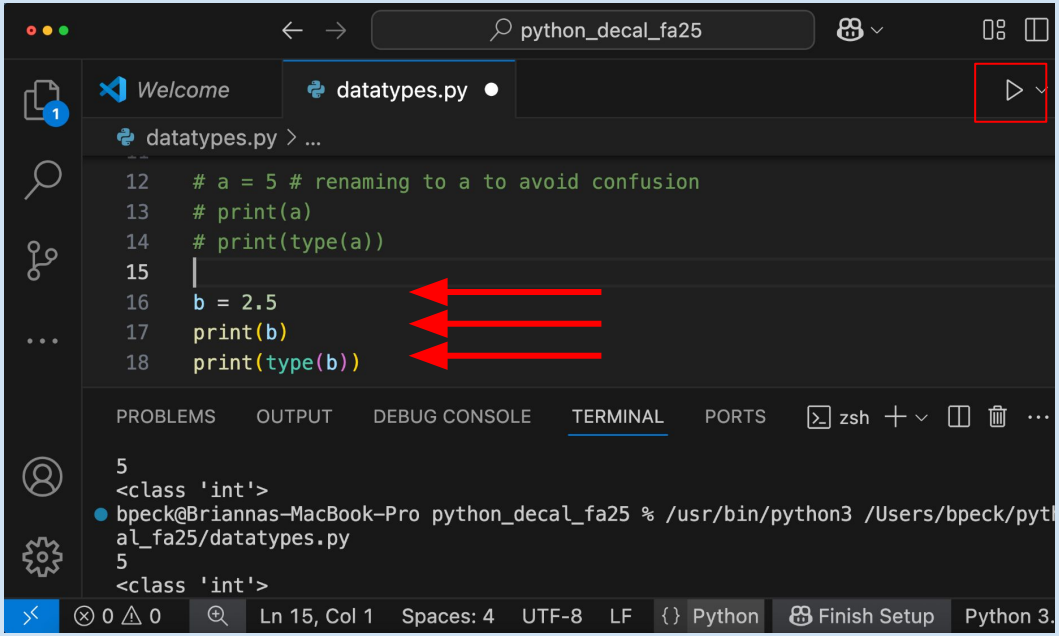


On the terminal, we see the “5” output again

But now we see a new data type: int

int = integer, a whole number

Data Types: Float



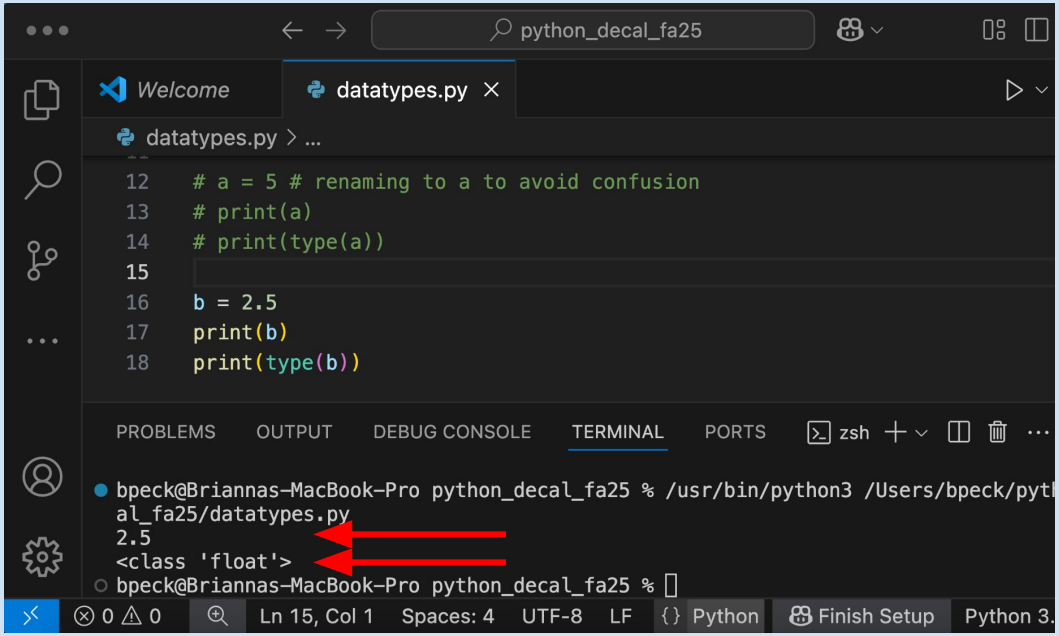
First comment out the code about a.

Then type:

- b = 2.5
- print(b)
- print(type(b))

Click the arrow in the box

Data Types (cont.)



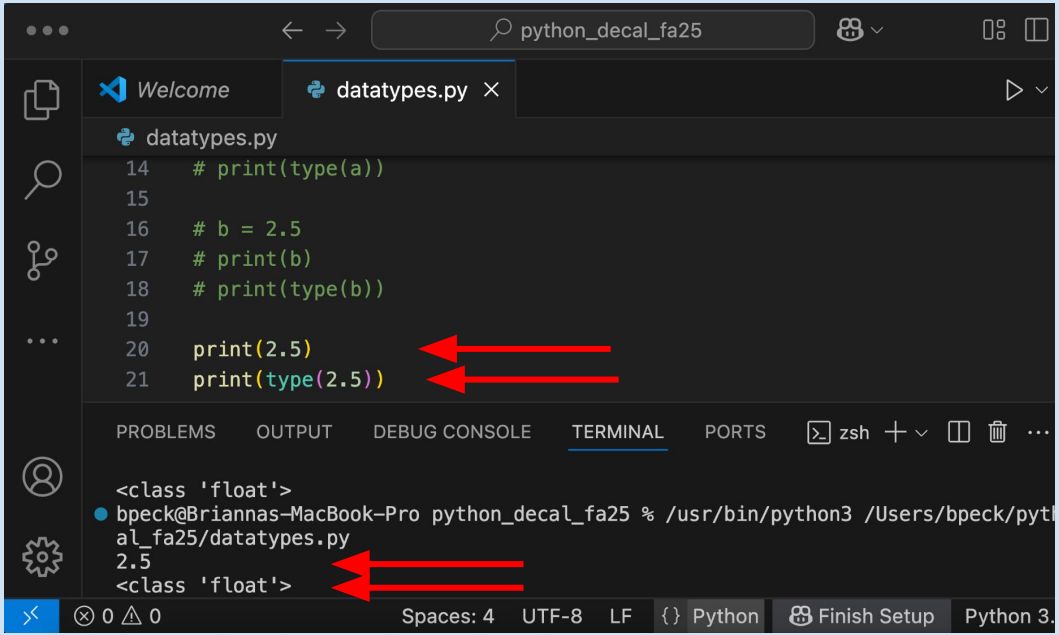
The terminal outputs b as 2.5, which is to be expected

But now we get:

<class 'float'>

float = a fractional number, or a number with a decimal

Data Types (cont.)

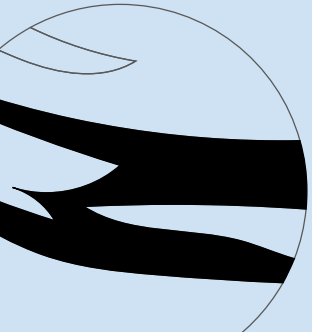


Note: You don't have to use a variable to use the built-in type() function

Just like print(), you can directly type into your type() function

Click the arrow to run

Let's practice



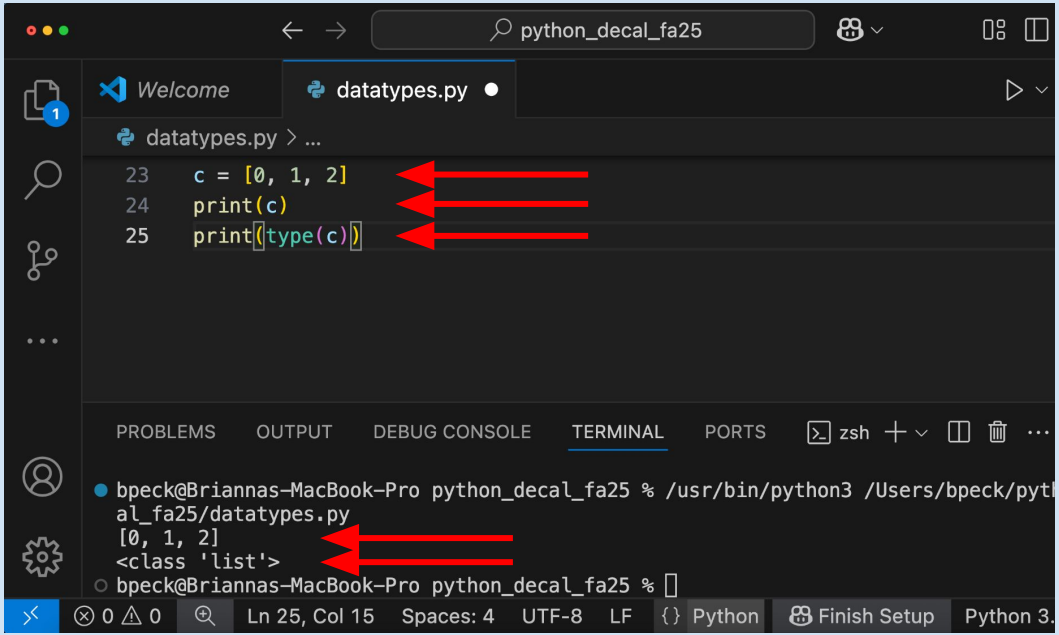
Practice Question

If there is time, let's answer these questions:

Close those laptops! With a partner, discuss the following.

1) How would you write a script to assign a variable to:
   - [0, 1, 2]
2) How would you print that variable?
3) How would you determine the data type of that variable?
4) How would you run the script on VS Code?

Practice Question Answer

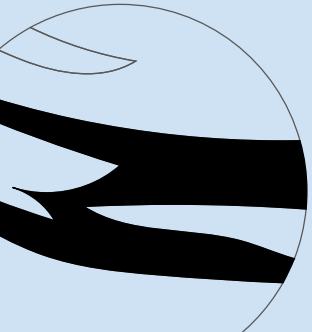


To the left:

Answers to (1), (2), (3)

4) Click the arrow to run the script in the VS Code terminal

command line + terminal



Terminal Background

We already started working with the terminal on VS Code

Now we will work with the actual application on your computer

Windows users: Make sure you are on Git Bash!!!

- Not PowerShell

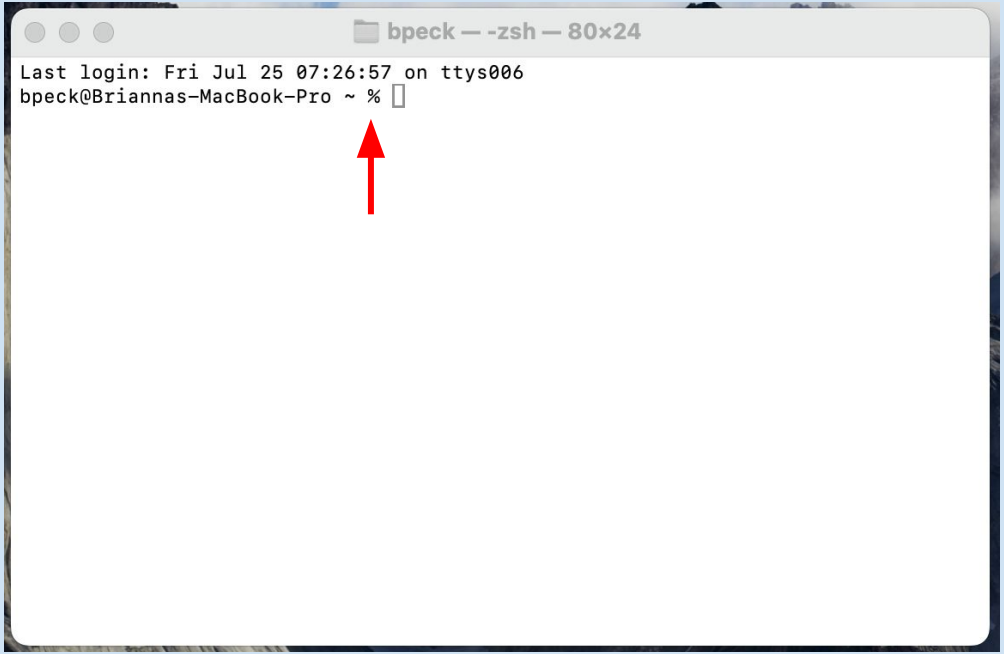
Open the Terminal

Moving away from VS Code, let's work directly with the terminal

Search for Terminal on your computer and open the application

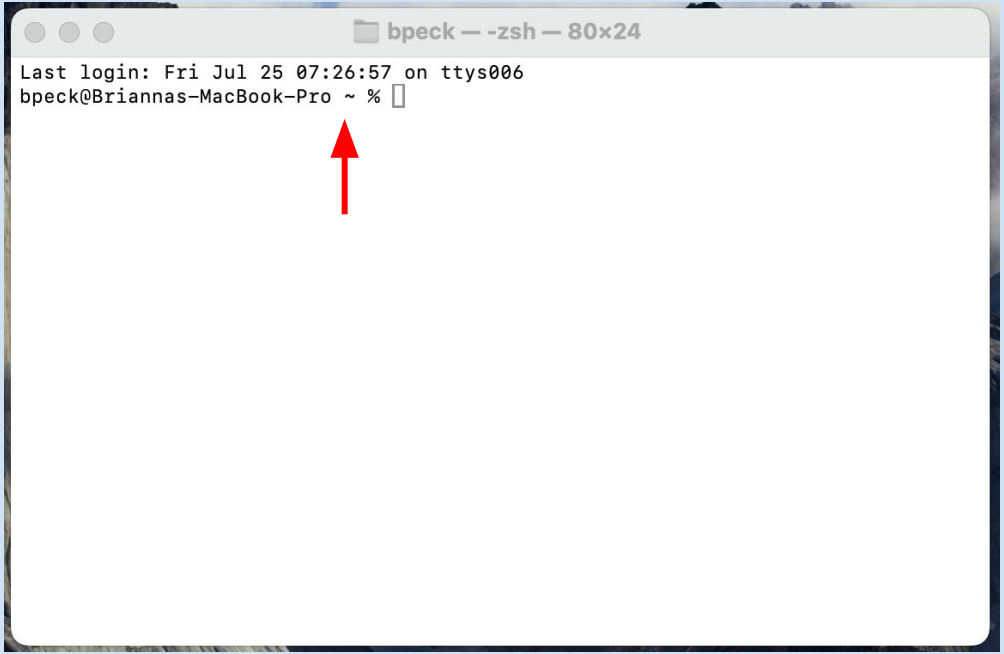
This is what mine looks like

Ready for a command



Just like with VS Code, we have the percentage % symbol which tell us that our command line is awaiting it's next command from us

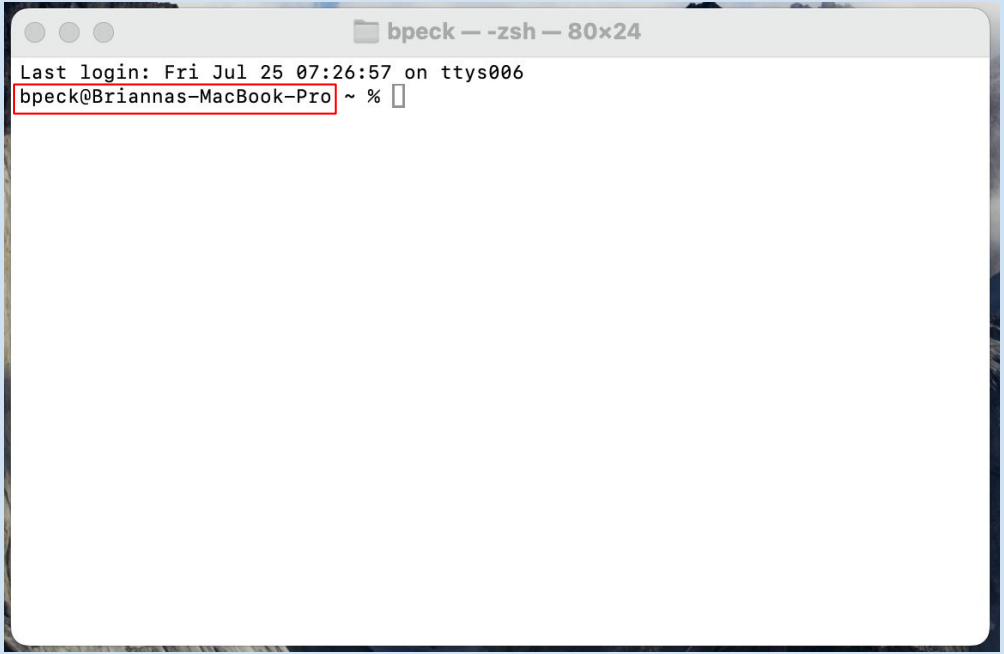
Home = Default Directory



The tilde ~ symbol means that we are in our home directory

Home directory = default location when you first open up your terminal, designated by the ~ symbol

Home = Default Directory



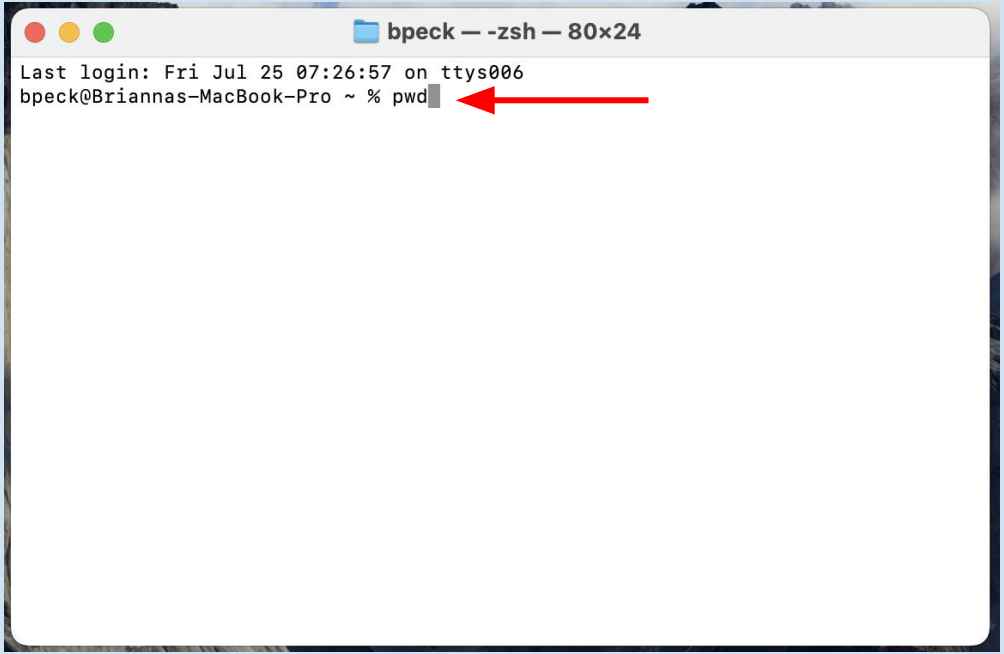
The stuff in the box tells you who is using the computer and what computer is being used

Users = bpeck

Computer:

- Briannas-MacBook-Pro (it might also just be numbers = IP Address)

PWD = PRINT CWD

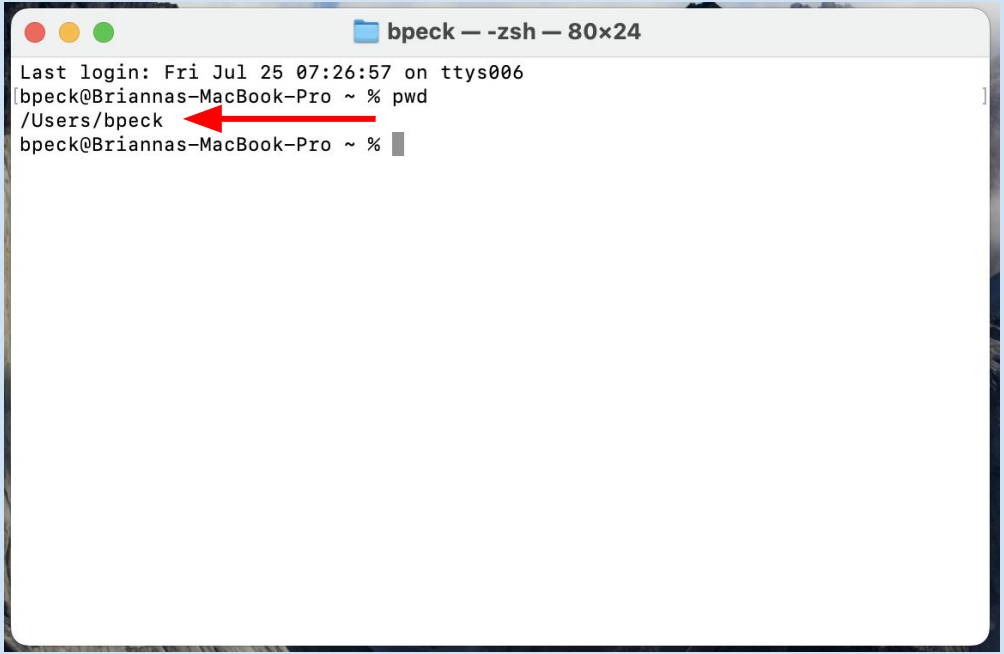


Type the command: pwd

This will tell your command line too: print the current working directory

current working directory = the folder you are currently in (we are in the “home” directory)

current working directory

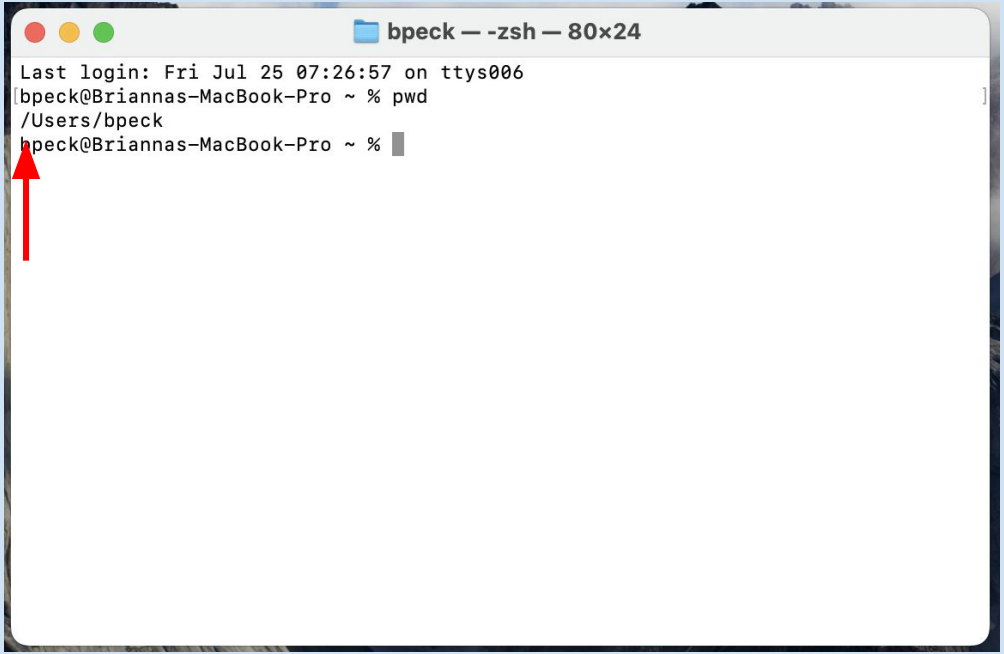


My terminal tells me, that I am currently in: /Users/bpeck/

Each folder is separated by a forward slash / symbol

So the top folder is Users, which holds bpeck

The root directory

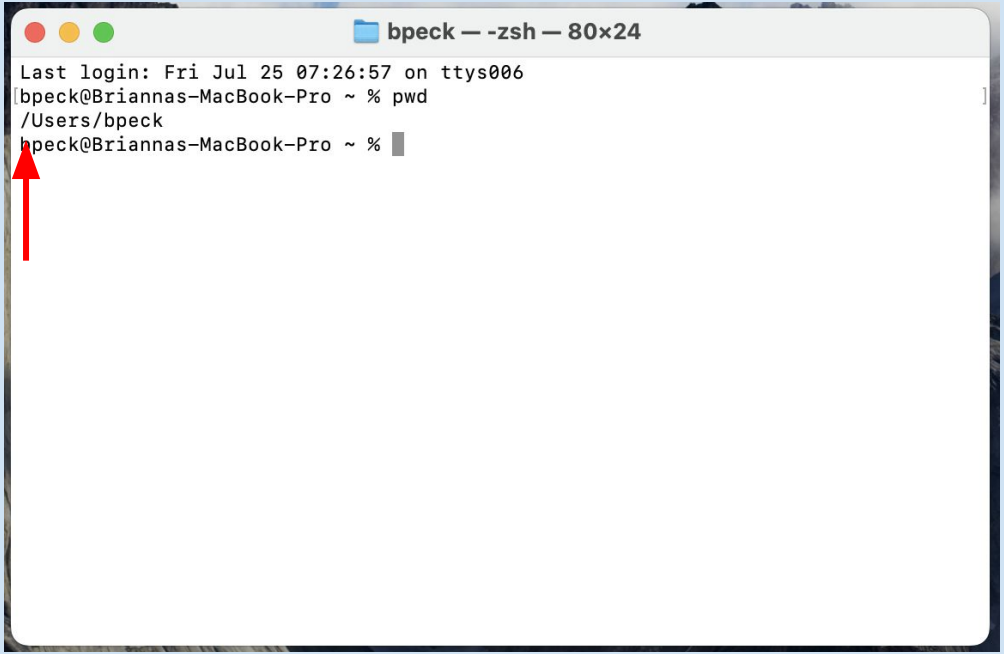


The forward slash / at the very beginning of the line indicates the root directory

It's the very top folder of your entire computer, everything can be found from this folder

So: /Users/ = root directory

Differences in Terms



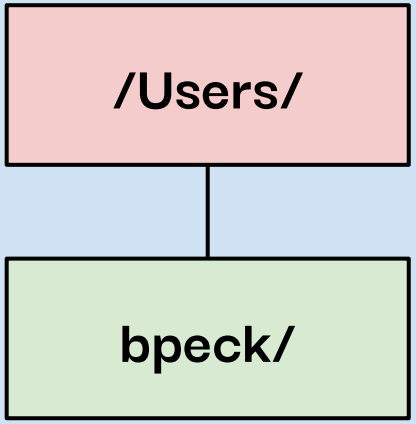
So:

/Users/ = root directory

bpeck/ = a directory (folder)

/Users/bpeck/ = home directory and current working directory

Quick Note: Directory Trees

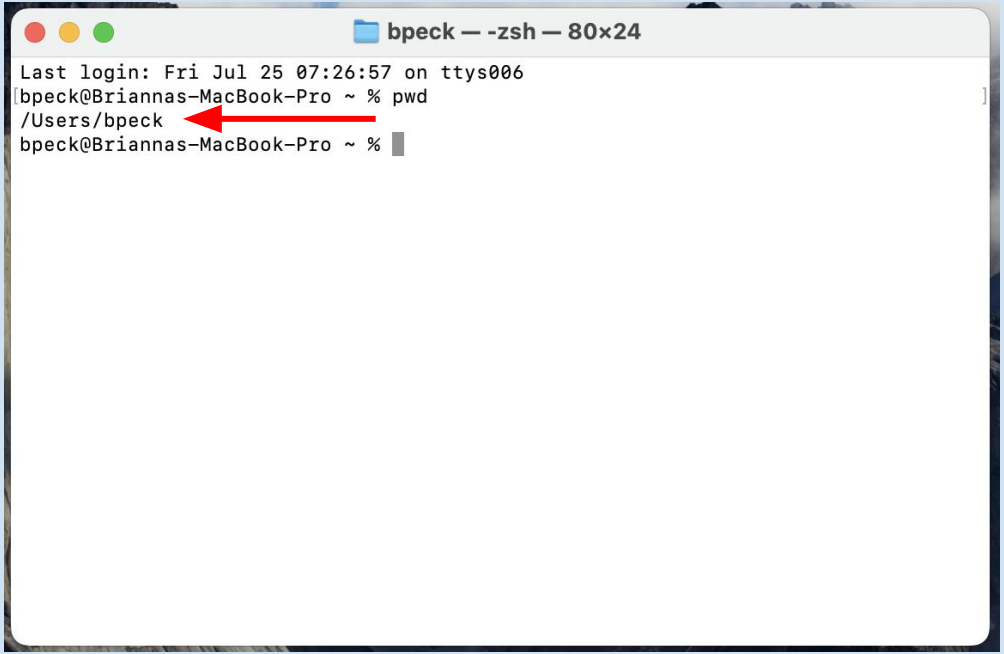


Note: Directories can live in other directories

So bpeck/ is living in /Users/

Left: Is what is called a directory tree, we will work more with these later

Differences in Terms



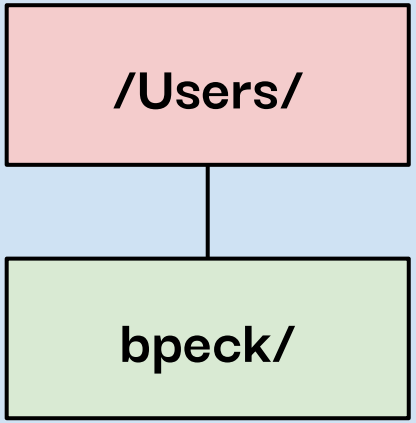
This whole line:

/Users/bpeck

is called a file path

File path = an address on your computer saying where a folder or file is located

Quick Note: Directory Trees

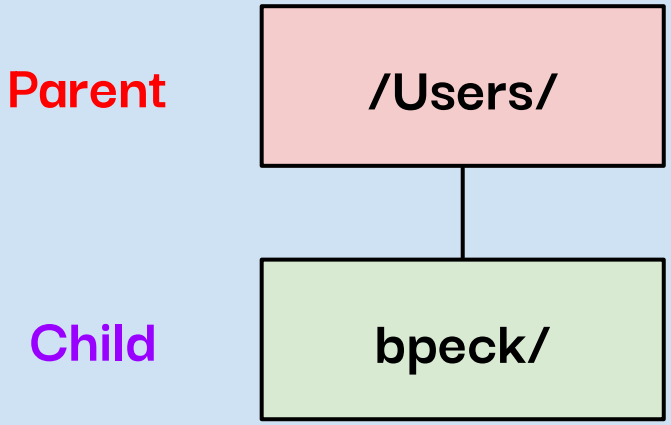


/Users/ is a different colors than /bpeck/ because it is a special directory

Red = root directory

Green = regular directory

Parent vs. Child Directories



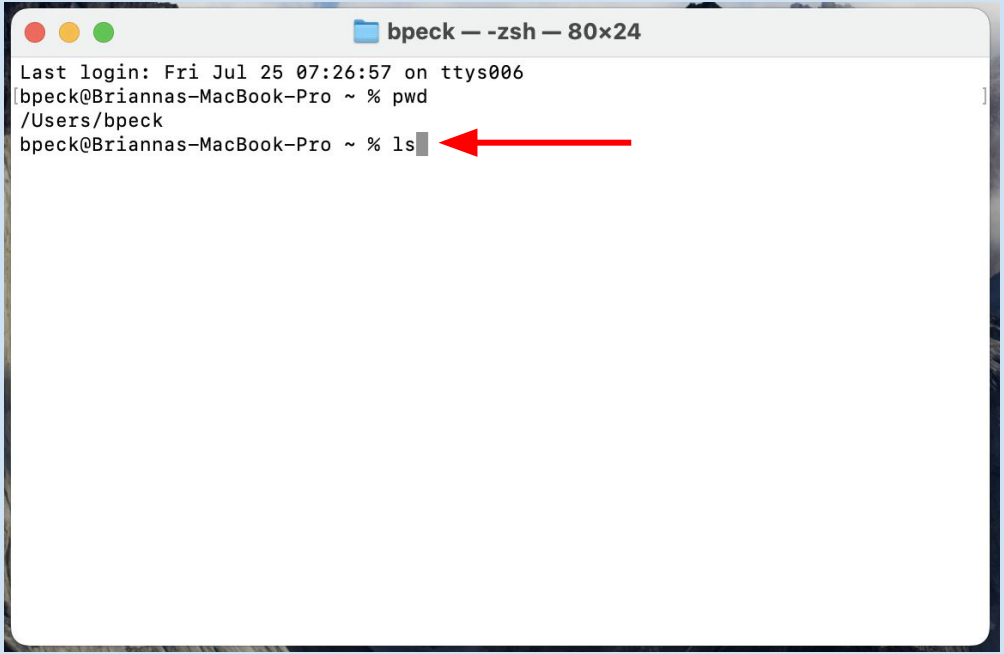
Directories are often referred to as parent and child directories

Parent = a folder that holds another folder

Child = the folder being held

So /Users/ is the parent and bpeck/ is the child

LS = LIST



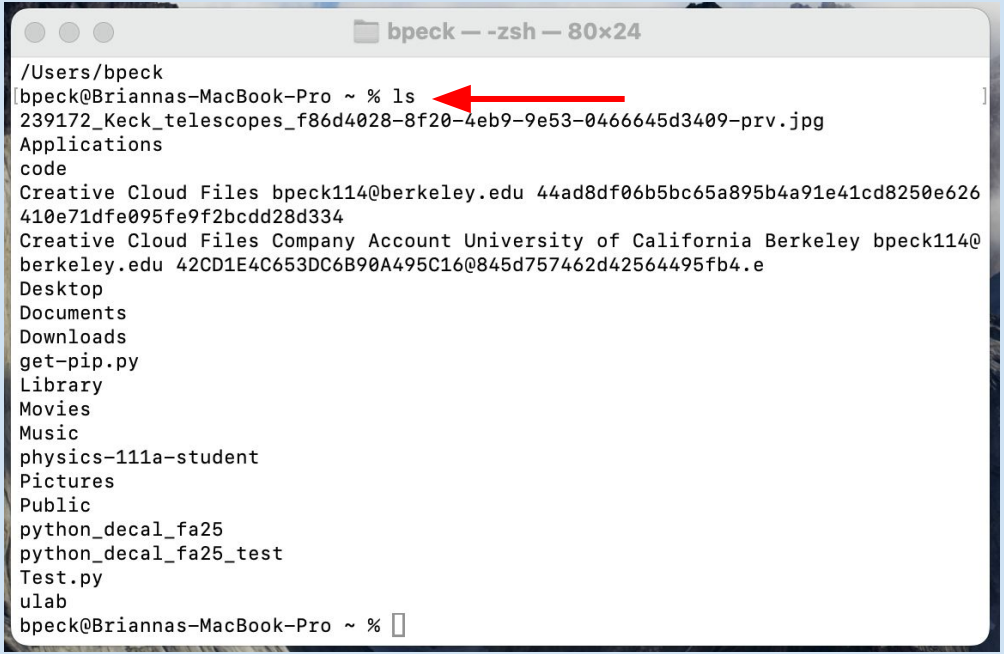
Type: ls

This command tells your terminal to list the contents of your folder

It will give you all the files and folders*

*except hidden one

LS = LIST



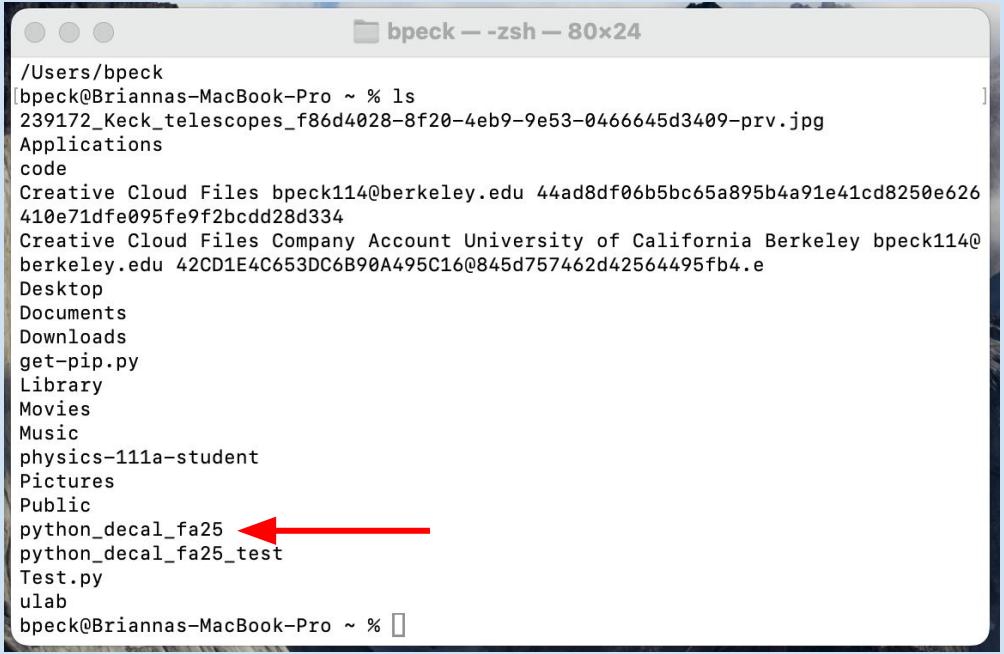
Each line after ls, shows a different file or folder

files = have extensions

- Like .py, .txt., .jpg

folders = have nothing

Locate New Folder

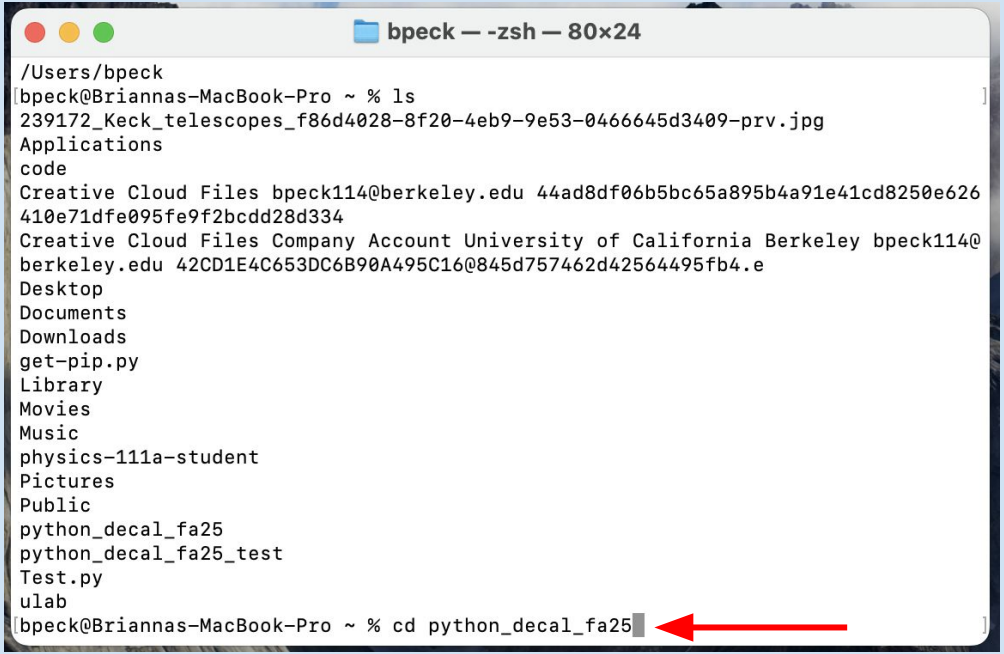


Search for the folder we made earlier called: python_decal_sp26

It should be in your home directory

Windows Users: It may not be, if so, ask for help

Change Directories



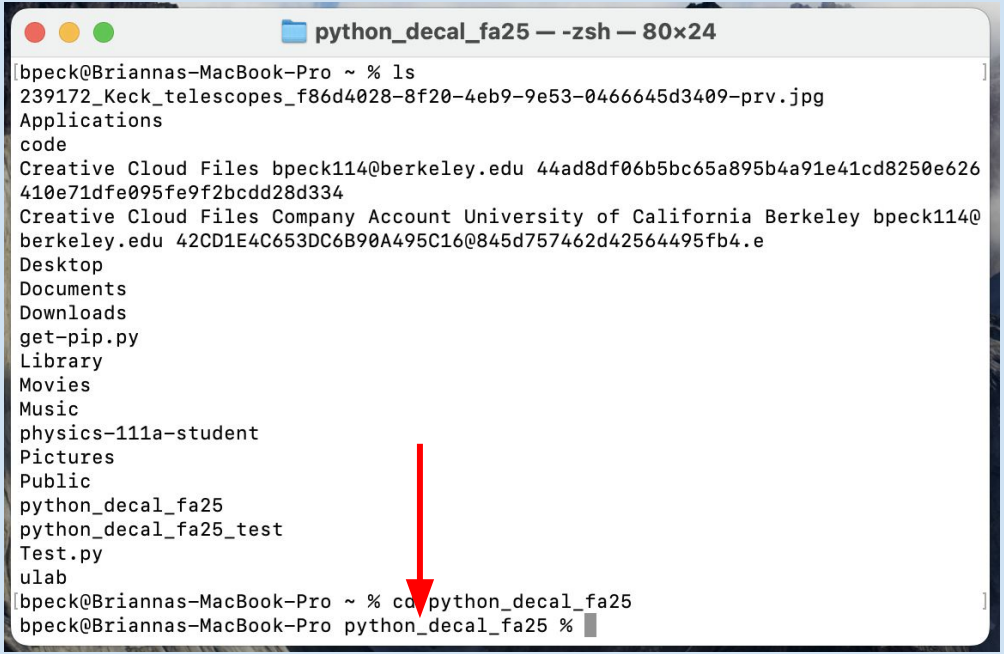
Now type:

cd python_decal_sp26

cd = change directories, changes the folder you are currently in

We are moving into a new folder inside your home directory

Change Directories

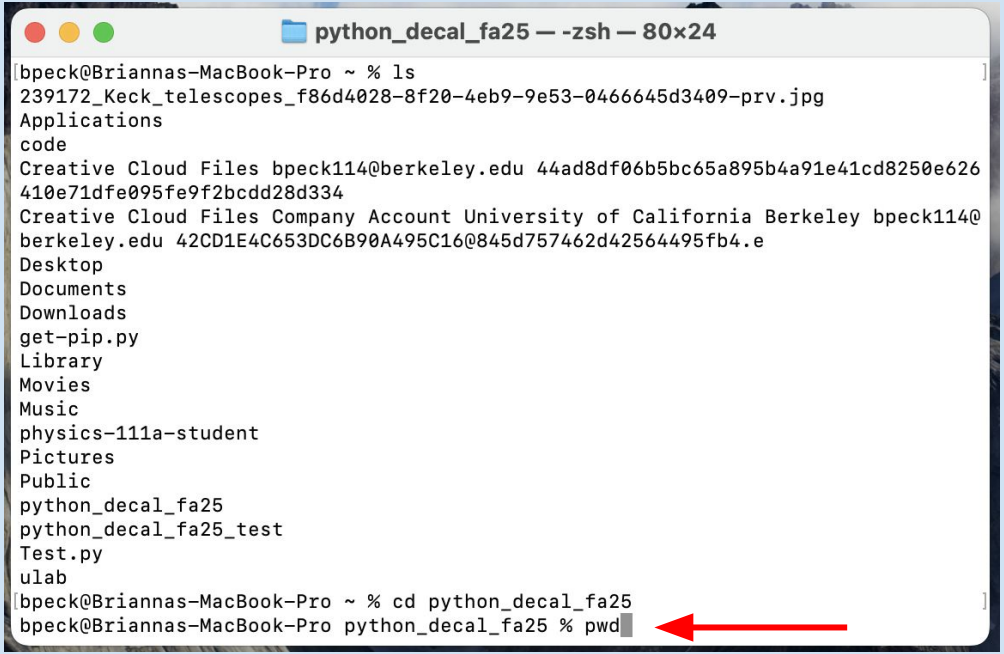


After you run the command, your terminal might display what folder you are in

If you computer acts like mine, instead of ~ you now see python_decal_sp26

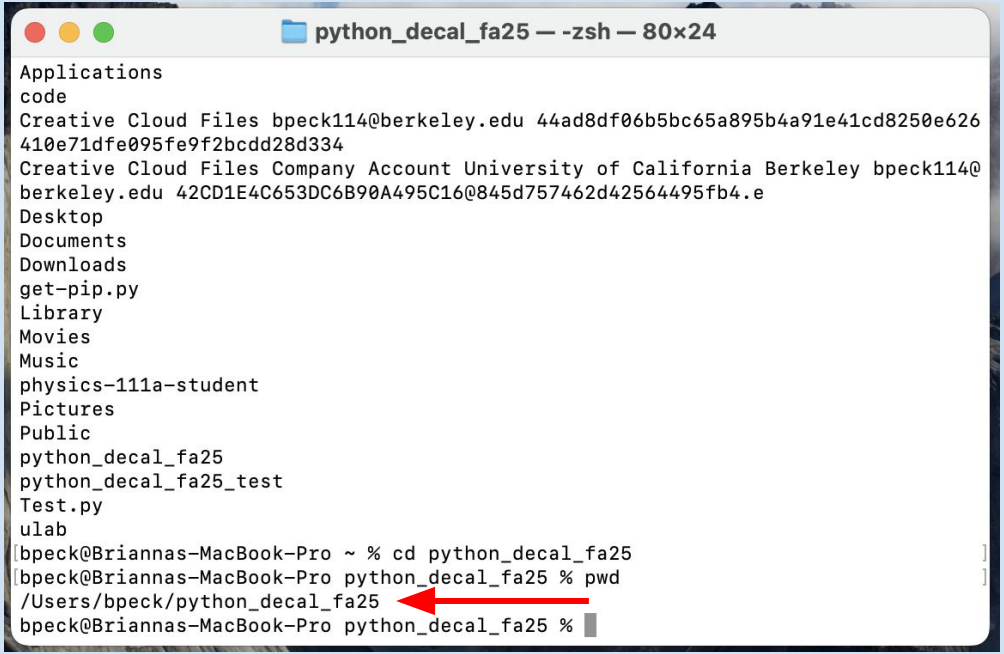
If your terminal, doesn't have this, no worries

Change Directories



To officially see what folder we are in now, type: pwd again

change directories



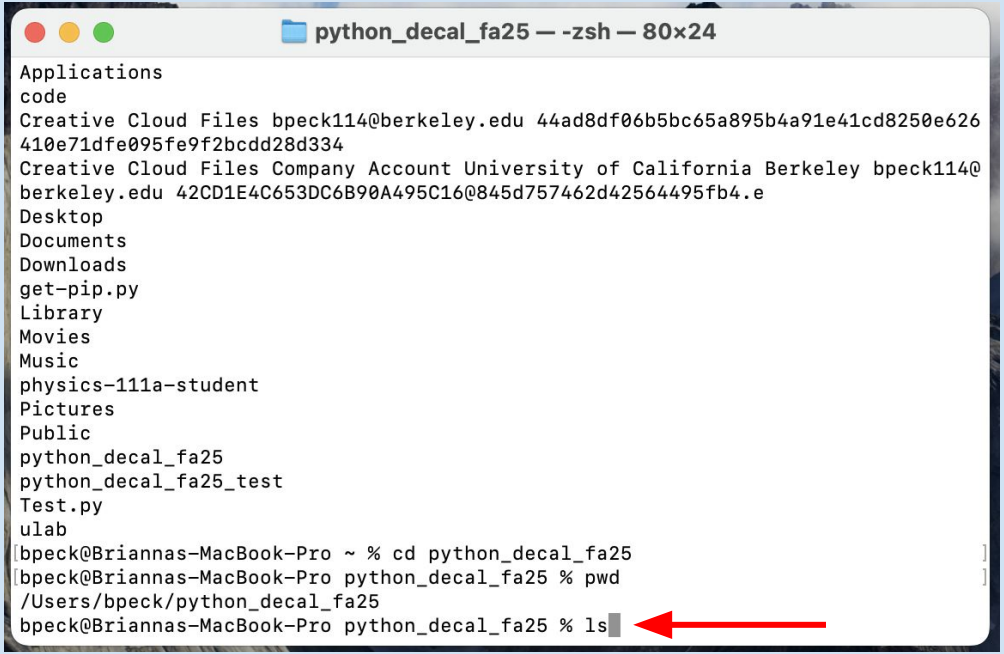
We still see the original structure

root directory = /Users/

home directory = bpeck/

new directory = /python_decal_sp26

Change Directories

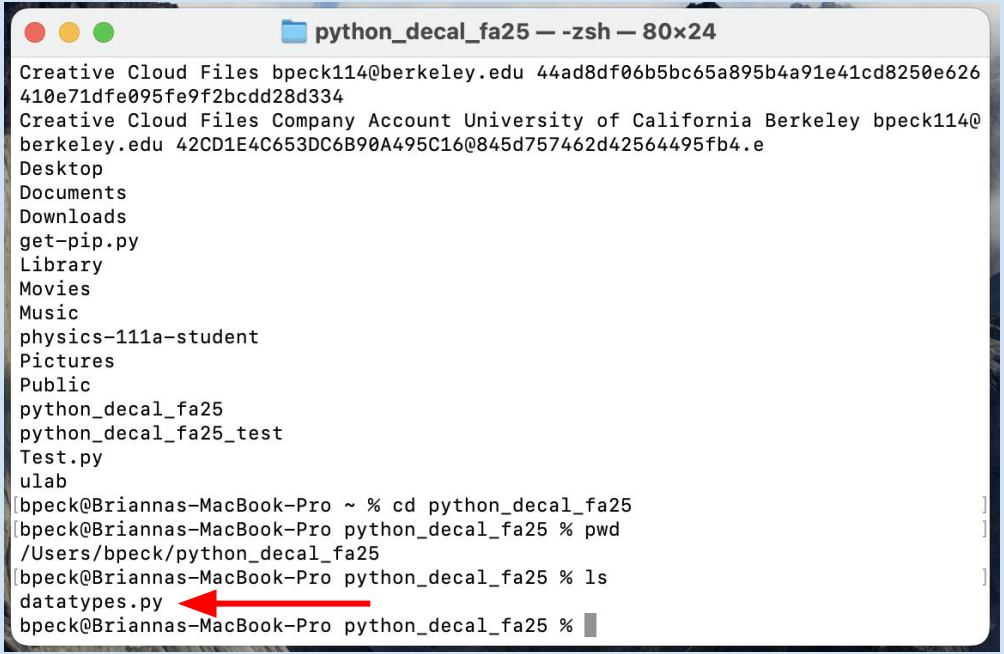


Type: ls

We are listing all the contents of the folder

Will it be empty? Discuss.

LIST CONTENTS

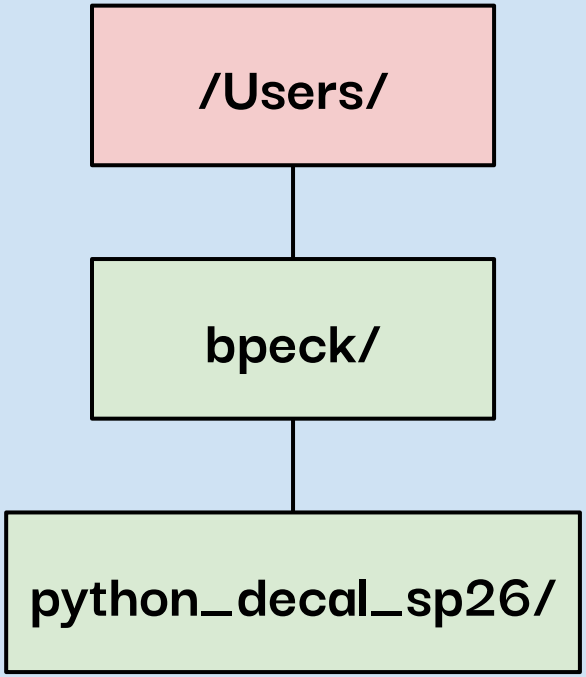


Nope! It's not empty.

It holds the file we made earlier in VS Code!

File: datatypes.py

Quick Note: Directory Trees

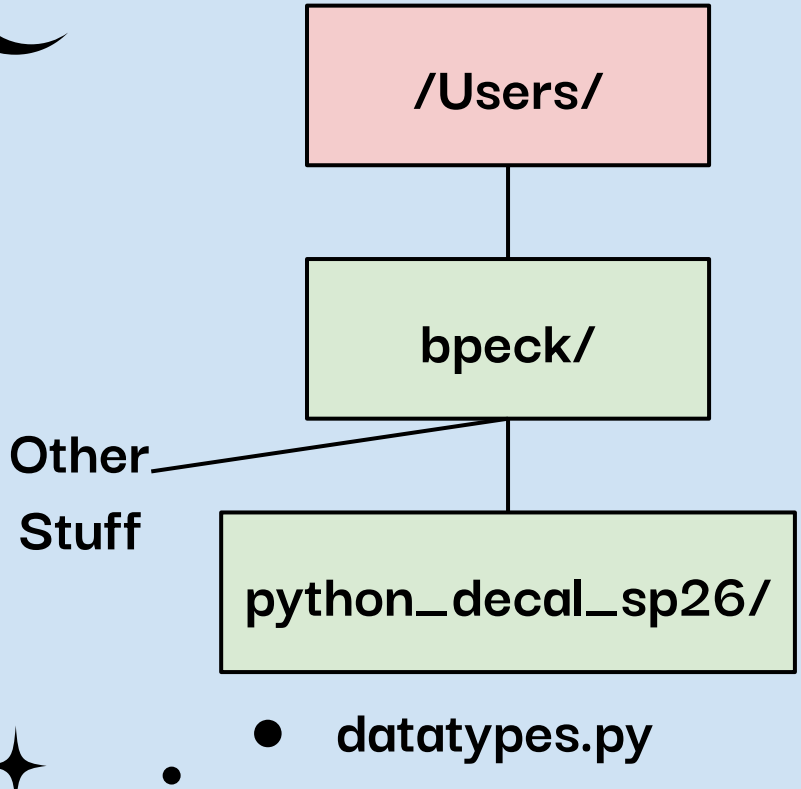


If we return to our directory tree again, we see a structure like the one to the left.

bpeck/ holds a lot of other files and folders, but we are not showing that currently

python_decal_sp26/ holds the file datatypes.py

Quick Note: Directory Trees

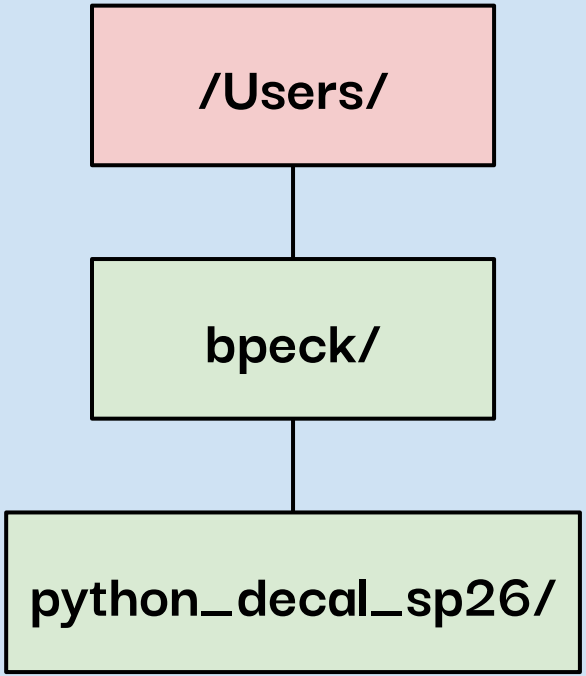


If we return to our directory tree again, we see a structure like the one to the left.

bpeck/ holds a lot of other files and folders, but we are not showing that currently

python_decal_sp26/ holds the file datatypes.py

Quick Note: Parent/Child



- datatypes.py

If we think about parent vs. child directories again:

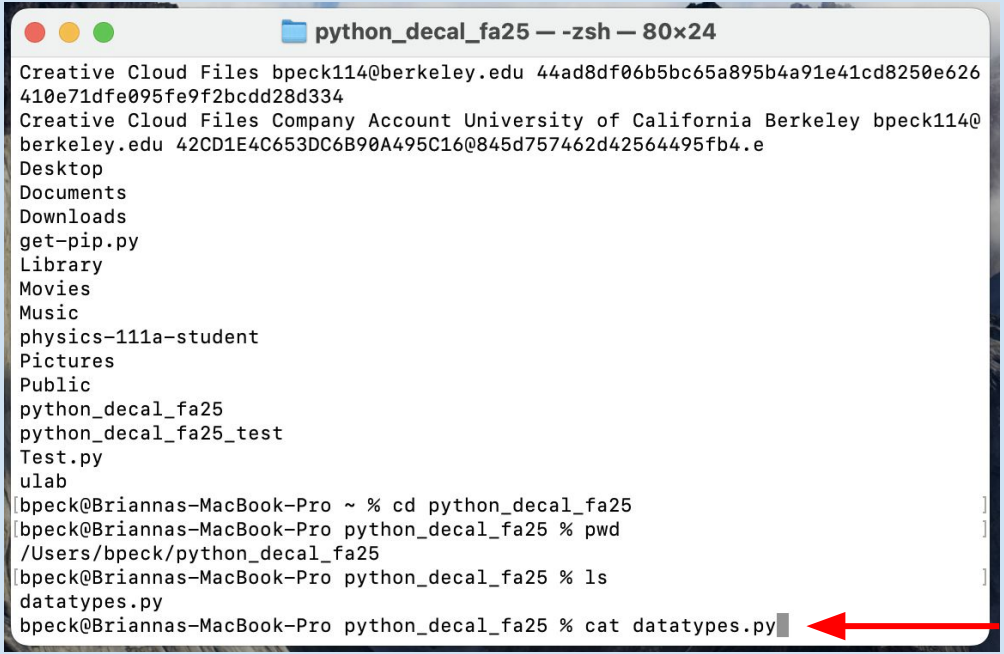
/Users/ is a parent to bpeck/

bpeck/ is a parent to python.../

python.../ is a child to bpeck/

bpeck/ is a child to /Users/

concatenate a file

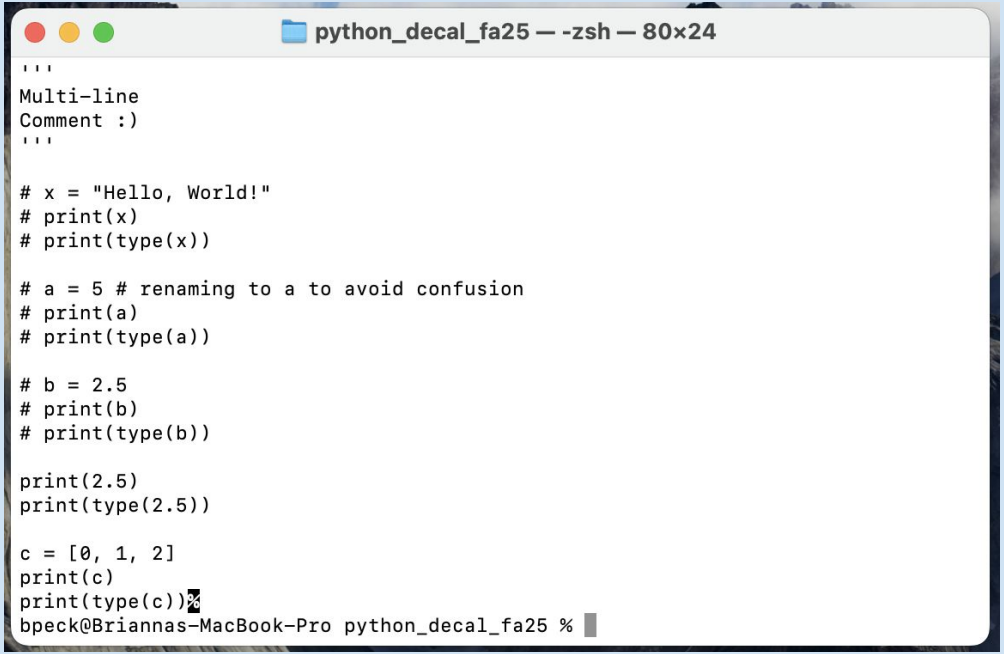


Type: cat datatypes.py

The cat command tells your computers to print out all the contents of a file

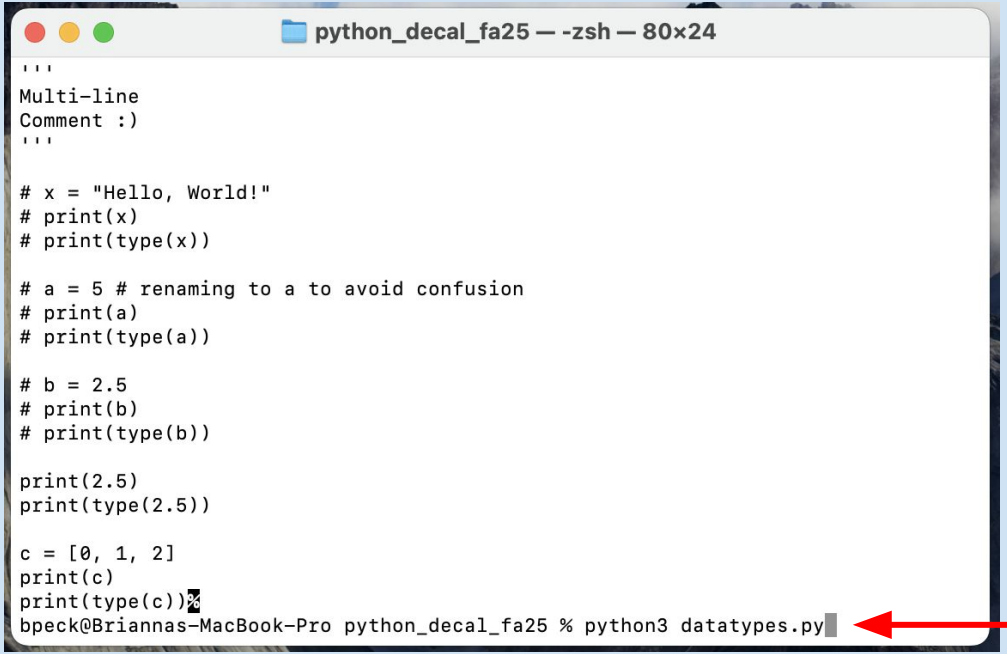
Cat = concatenate (🐱)

Concatenate a File



The terminal spit out every single line in our datatypes.py file

Run a script on Terminal

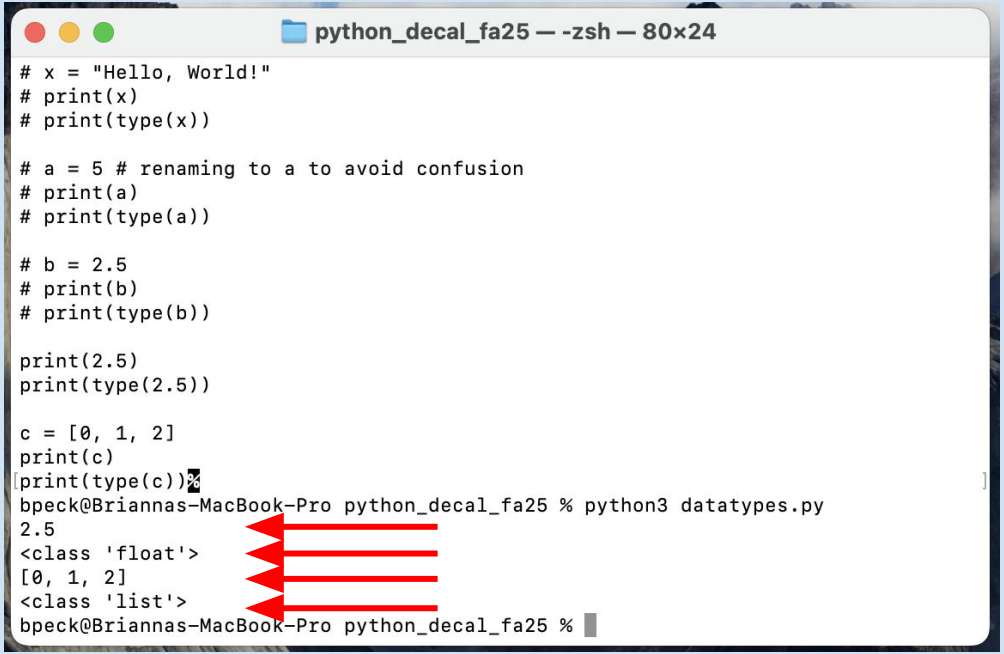


To run a script on the terminal without VS Code, type the following:

python3 datatypes.py

This tells your terminal to use python to run your script

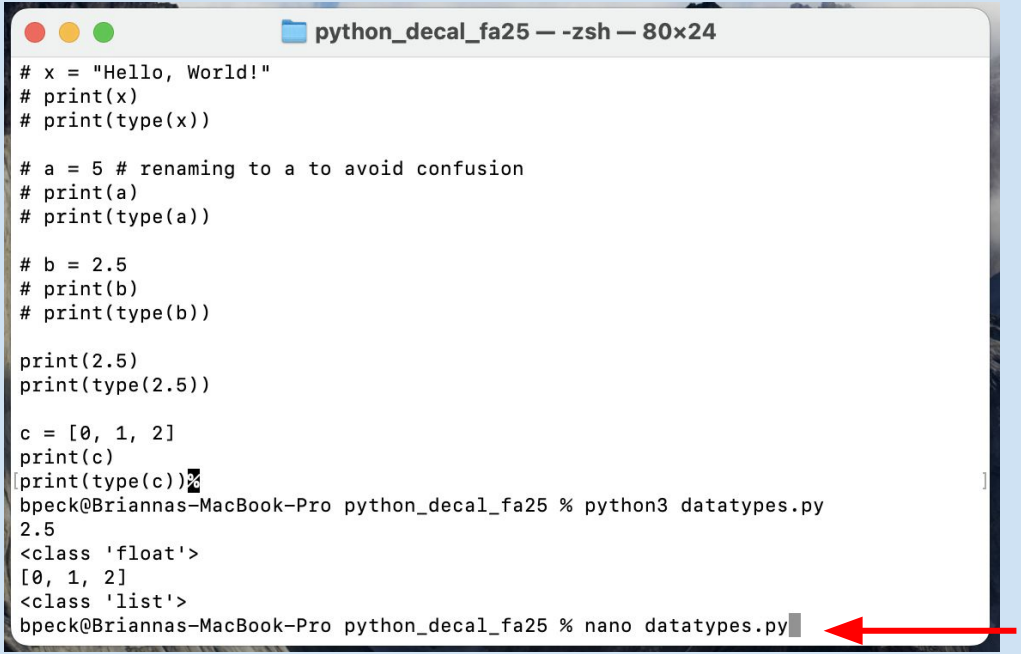
Run a script on Terminal



You might have slightly different outputs if you didn't comment out as much as I did

But the terminal just ran our script in Python and returned the print statements

Edit a file on the terminal



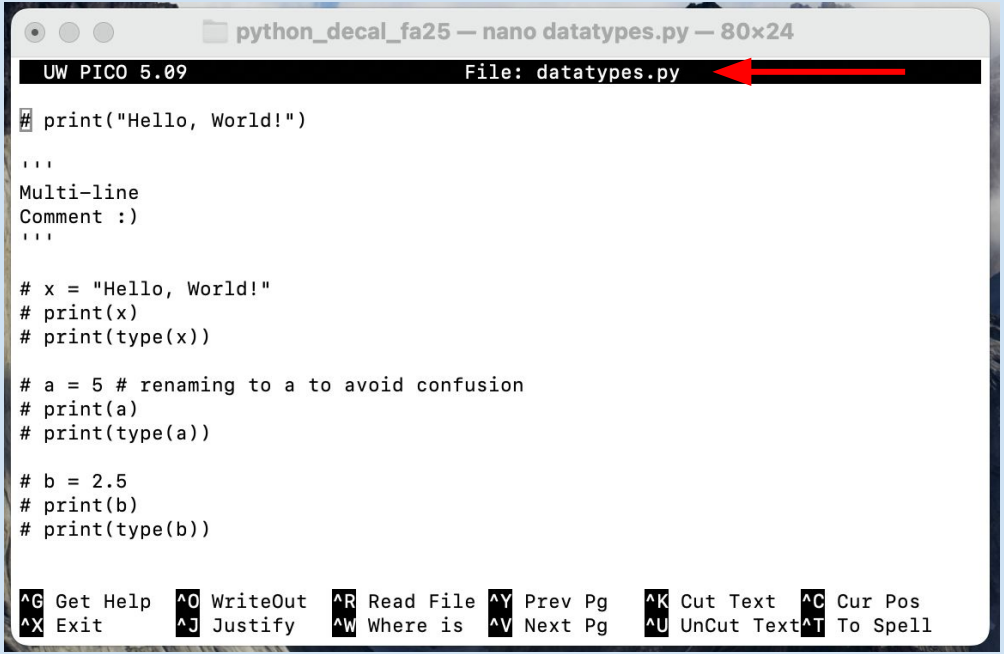
To edit your file on the command line:

Type: nano datatypes.py

There are other commands that are similar to nano like:

- vim
- emacs
- micro

EDIT A FILE ON THE TERMINAL



A new “window” appeared but you are still on the terminal

At the top is the name of the file: datatypes.py

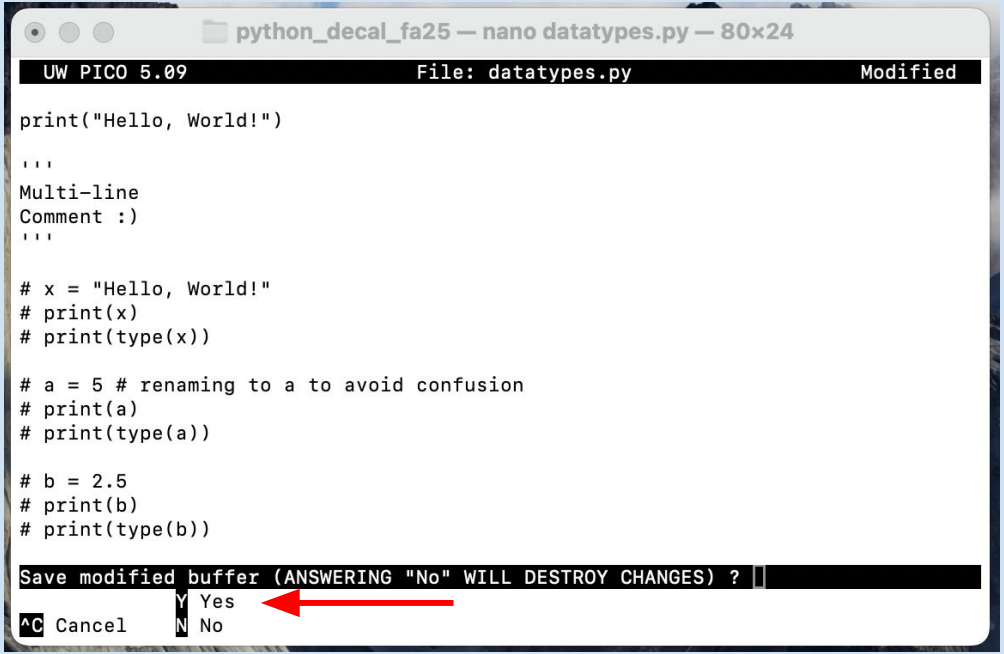
Below that, you have the file contents

Edit a File on the Terminal

Uncomment the very first line in the file

Then press Ctrl + X

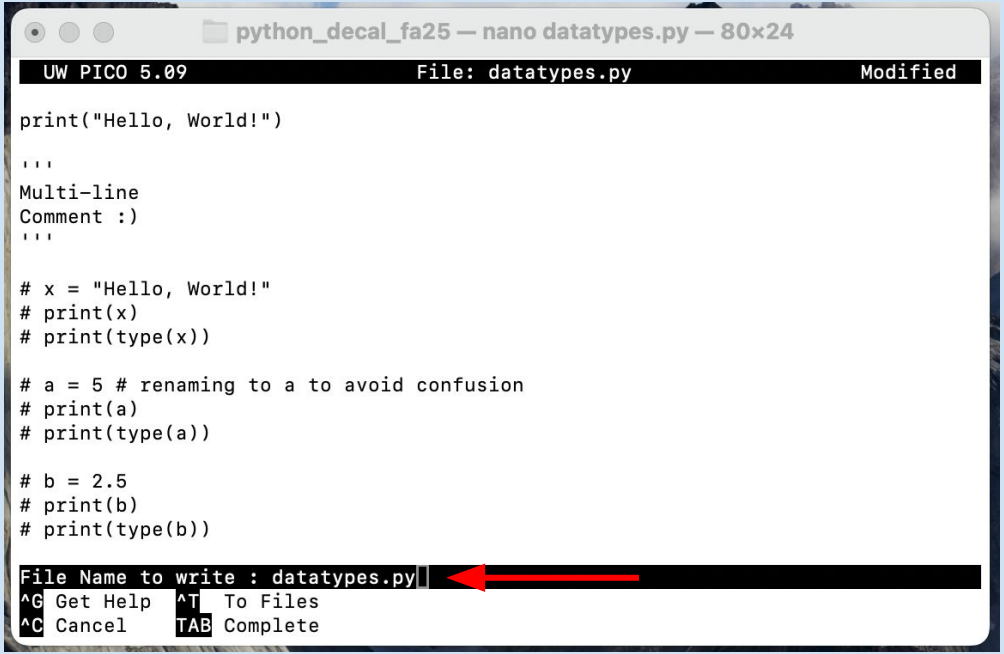
Edit a File on the Terminal



The nano window will prompt you will asking you if you want to save change

Press: Y to save your changes

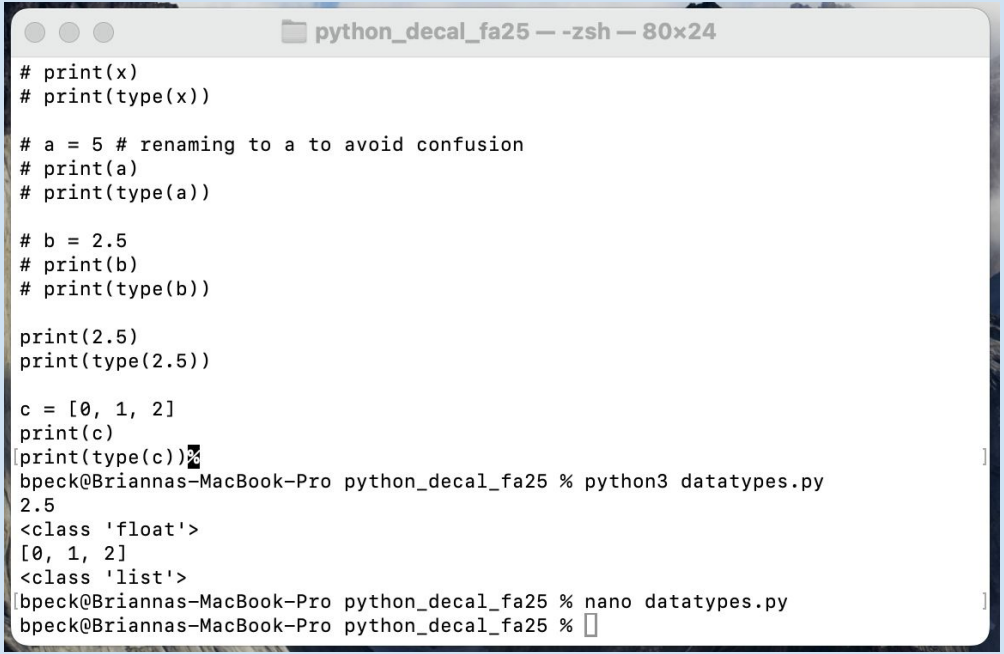
Edit a File on the Terminal



Then it will ask what file you want to save your contents too

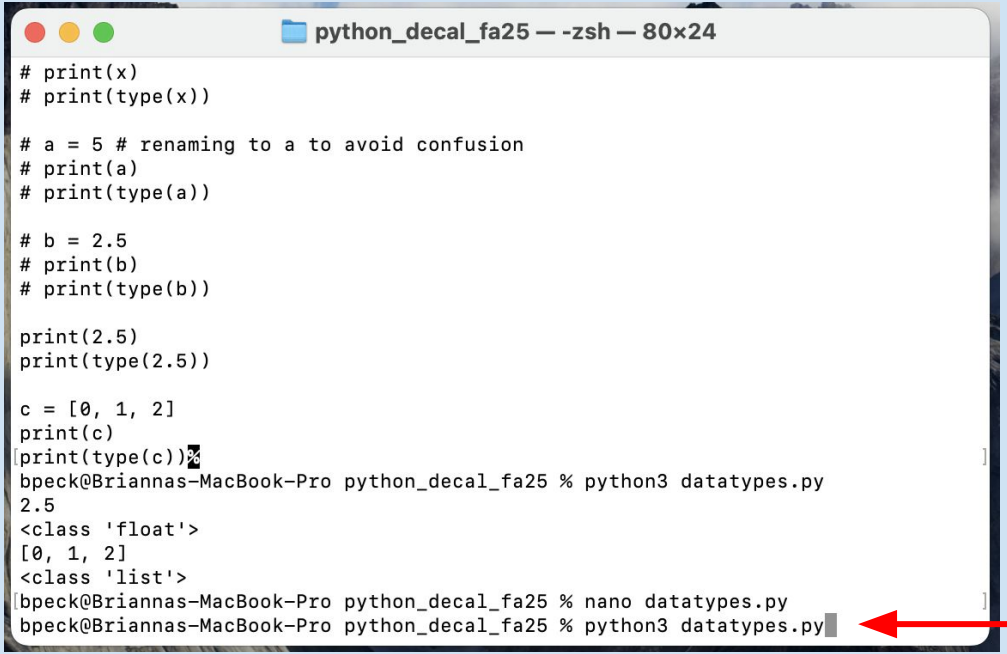
Just press: Return/Enter to not change the file name

Edit a file on the terminal



Once you are done, it will dump you back to this screen

Run a script on the terminal

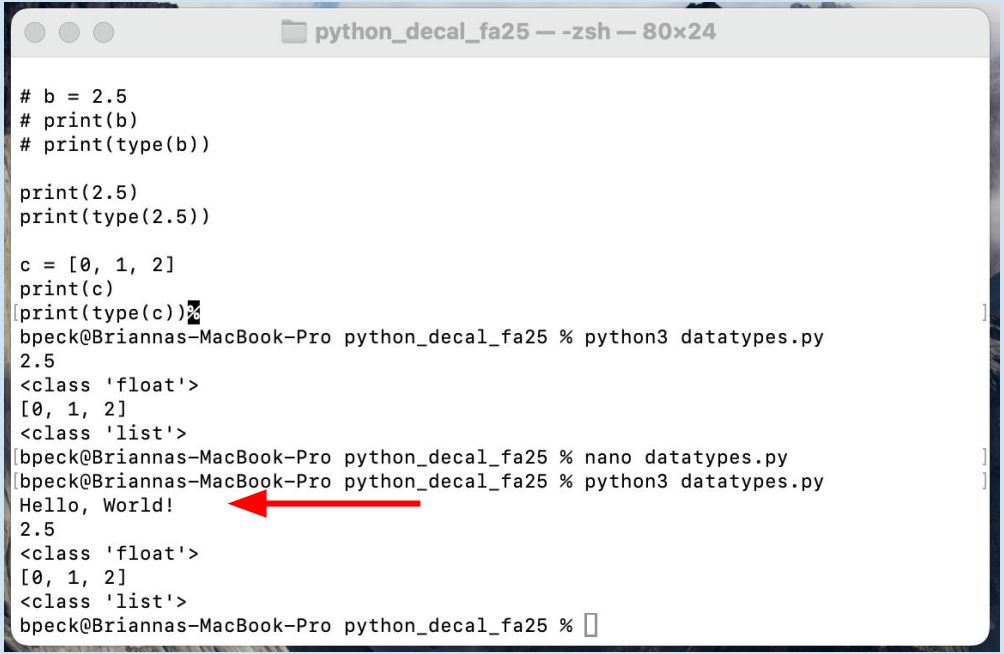


To run the script on your terminal again

Type:

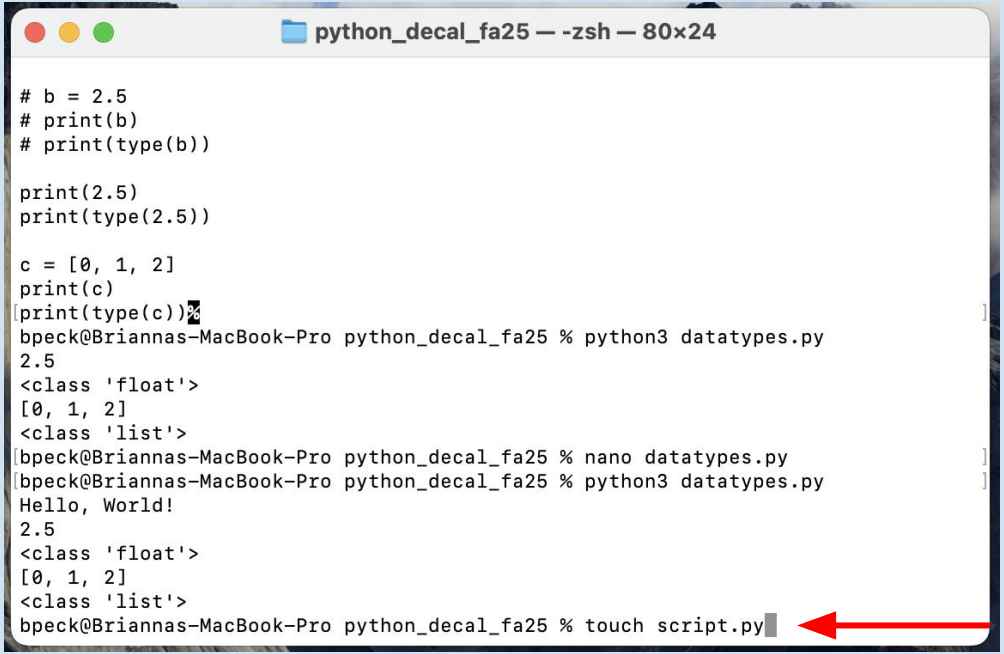
python3 datatypes.py again

Run a script on the terminal



The new “Hello, World!” appeared on the terminal!

Make a New File on Terminal

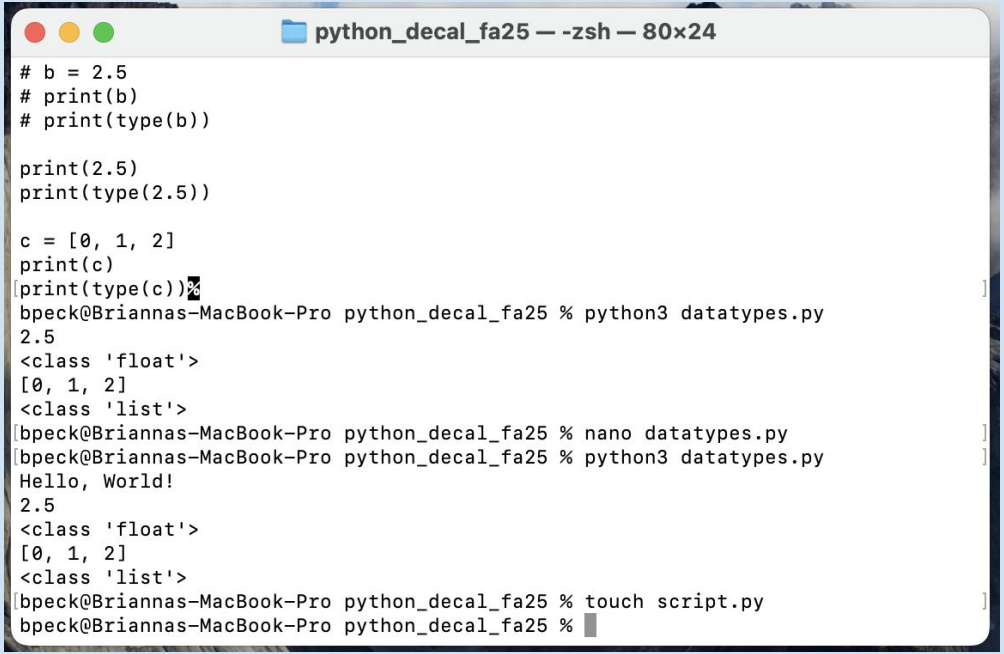


To make a new file on the terminal type:

touch script.py

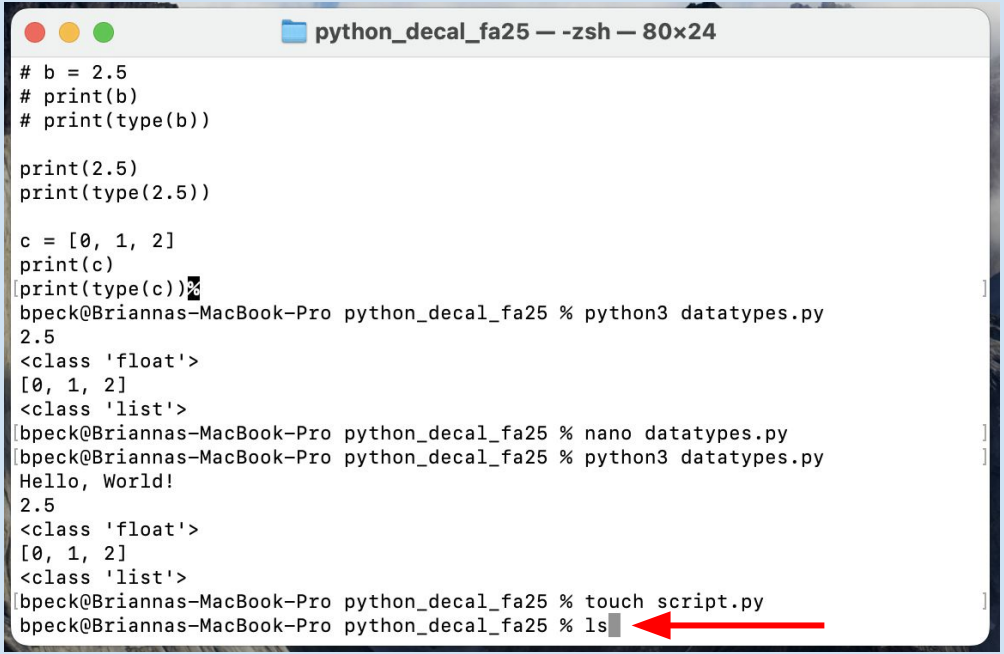
Note: You can also just call nano directly like:
nano script.py to open up an empty file.

Make a New File on Terminal



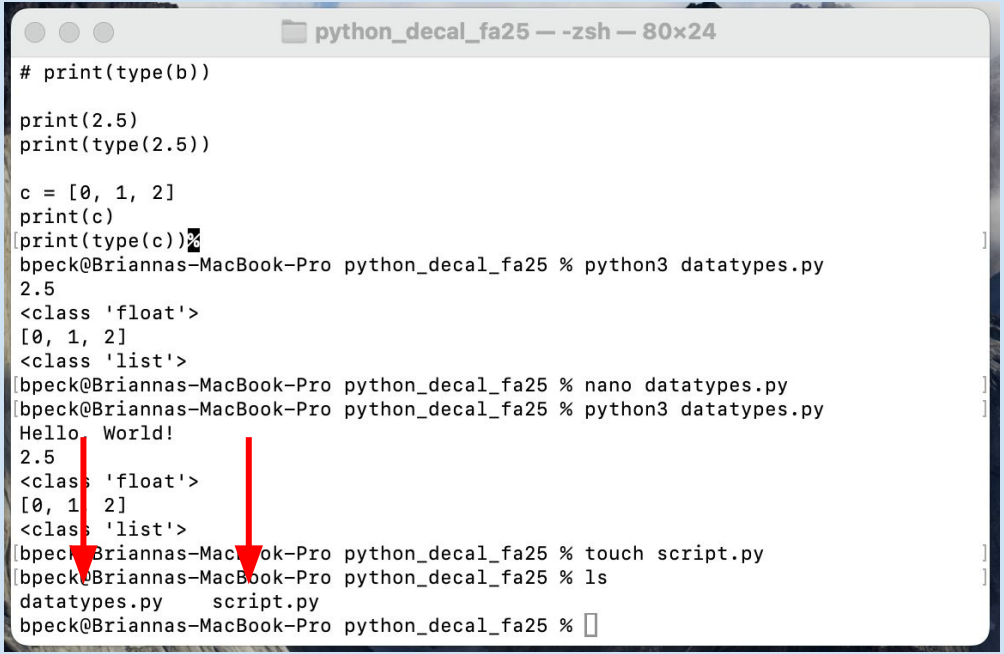
This is what your screen will look like after you call the command

Make a New File on Terminal



If you type ls on the terminal, we will see two files now

Make a New File on Terminal



If you type ls on the terminal, we will see two files now

datatypes.py

- which we just edited

script.py

- the new file we made

Quick Note: Directory Trees

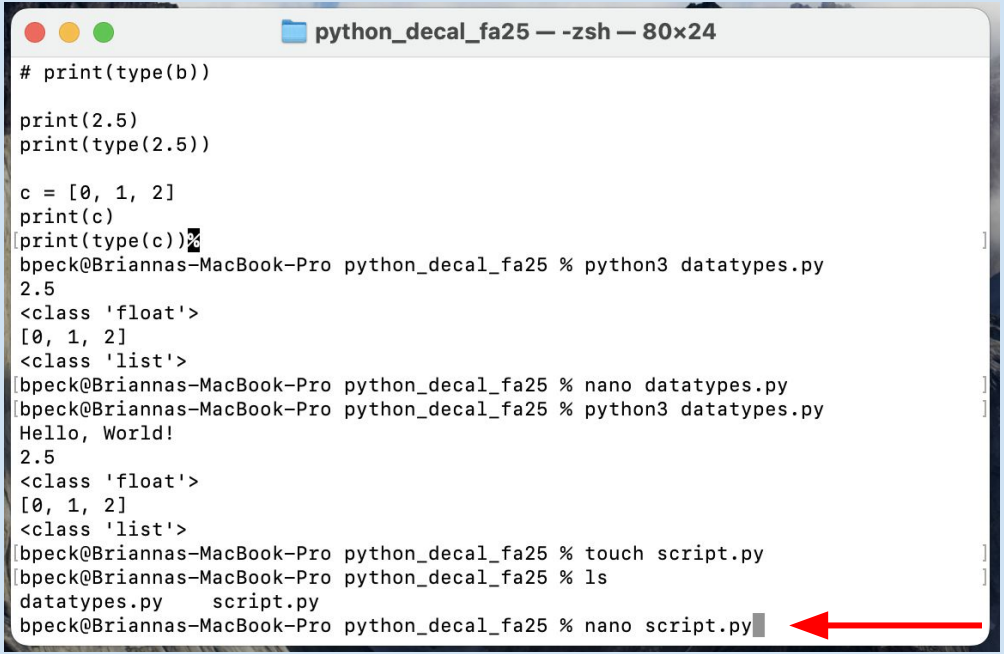
python_decal_sp26/

- datatypes.py
- script.py

Let's get rid of /Users/ and bpeck/ in our directory tree because we are running out of space..

Now our tree looks like this:

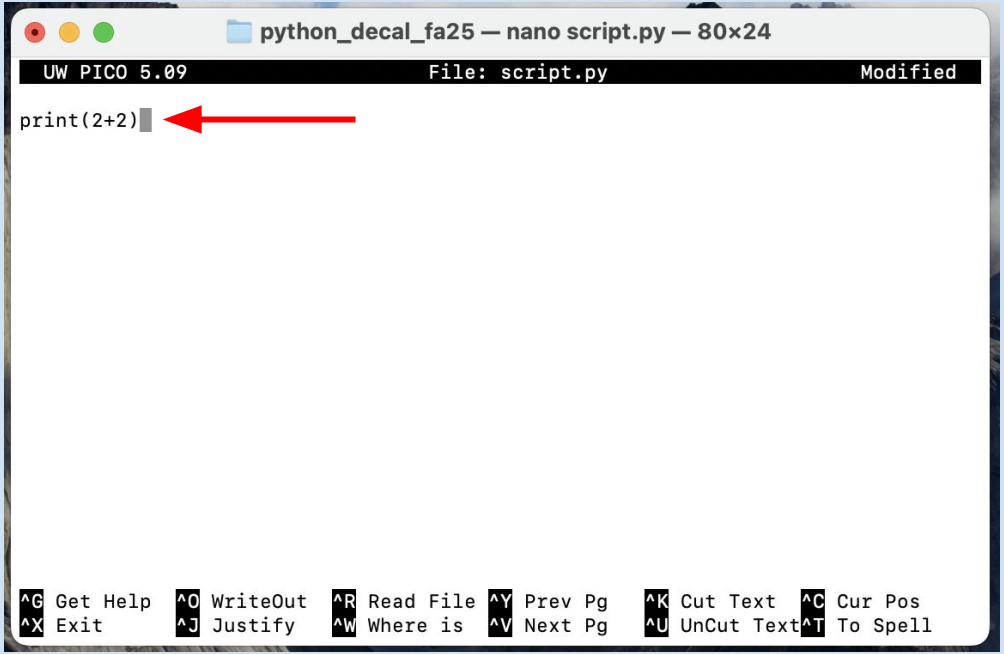
Make a New File on Terminal



We will edit this new file and run it again on the terminal

Type: nano script.py

Make a New File on Terminal

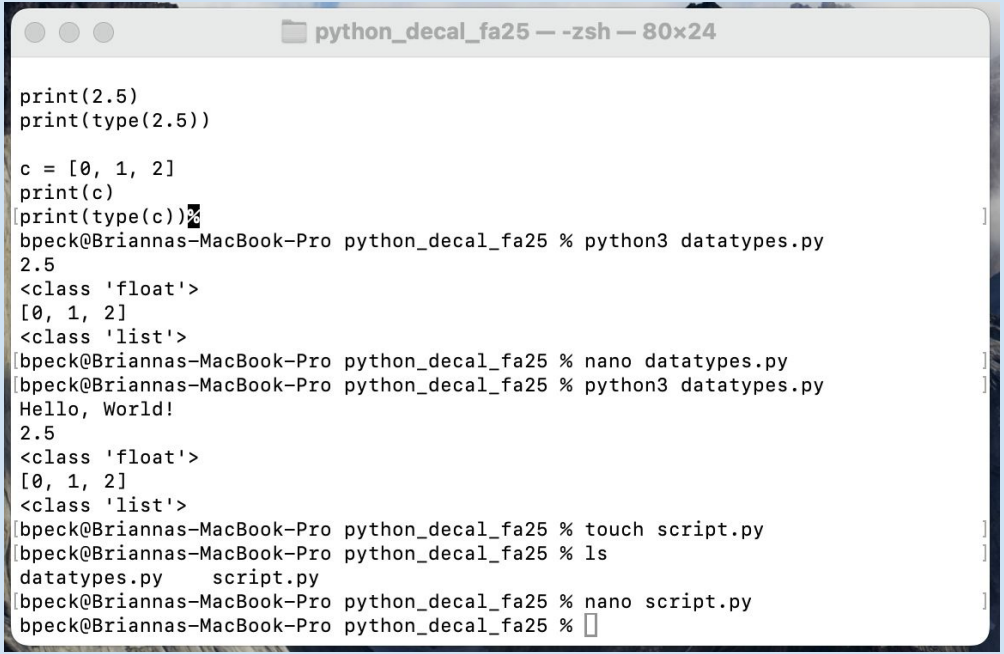


Add the following line to your script: print(2+2)

Then:

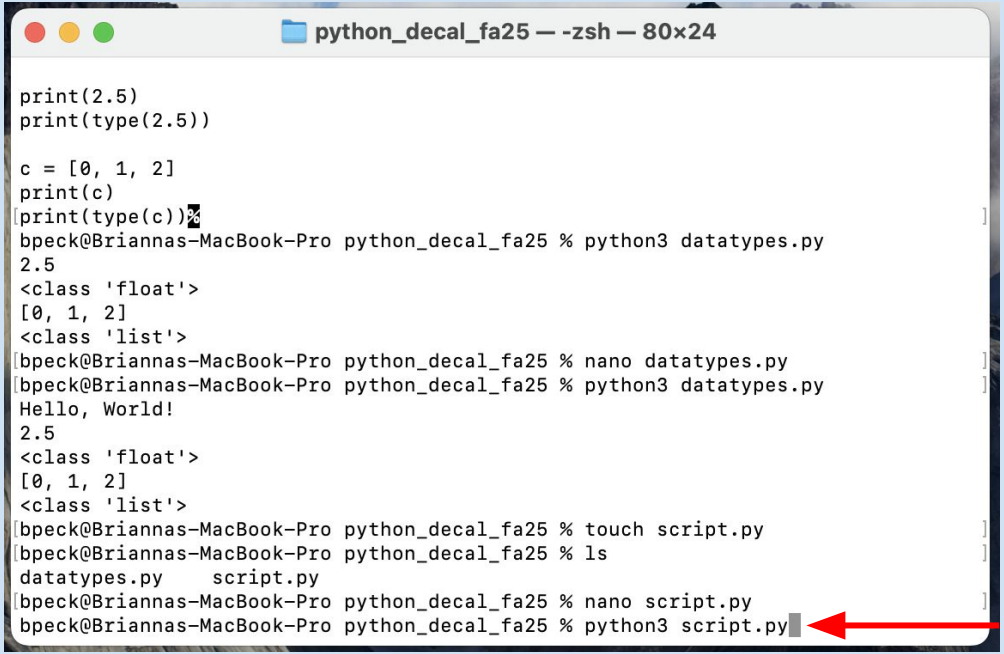
- Save your changes
- Close nano

Make a New File on Terminal



This is what your screen will output too

Make a New File on Terminal

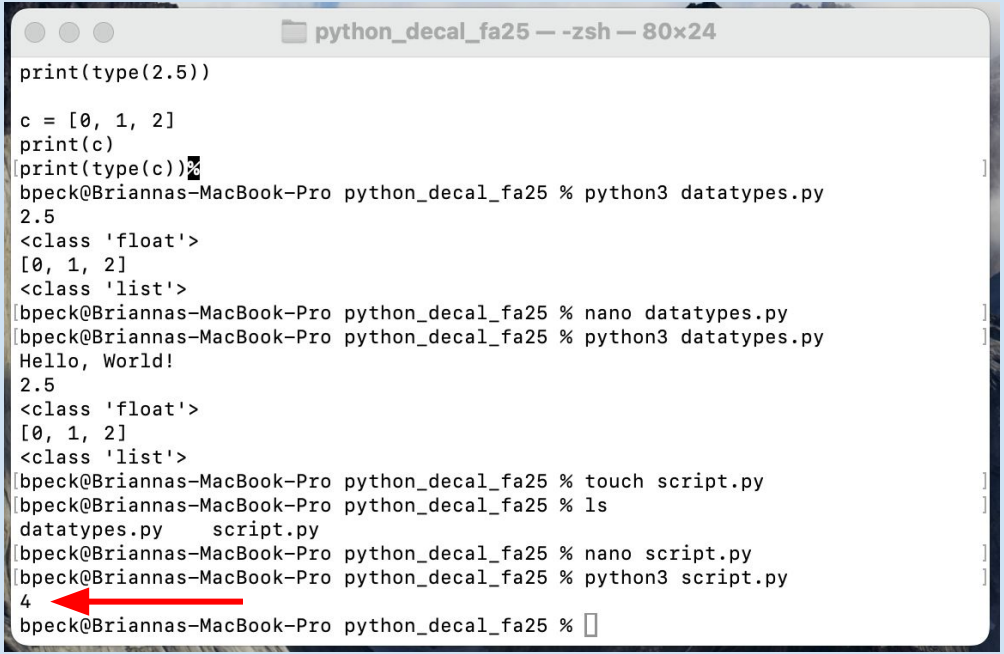


We will call Python once again on this file too

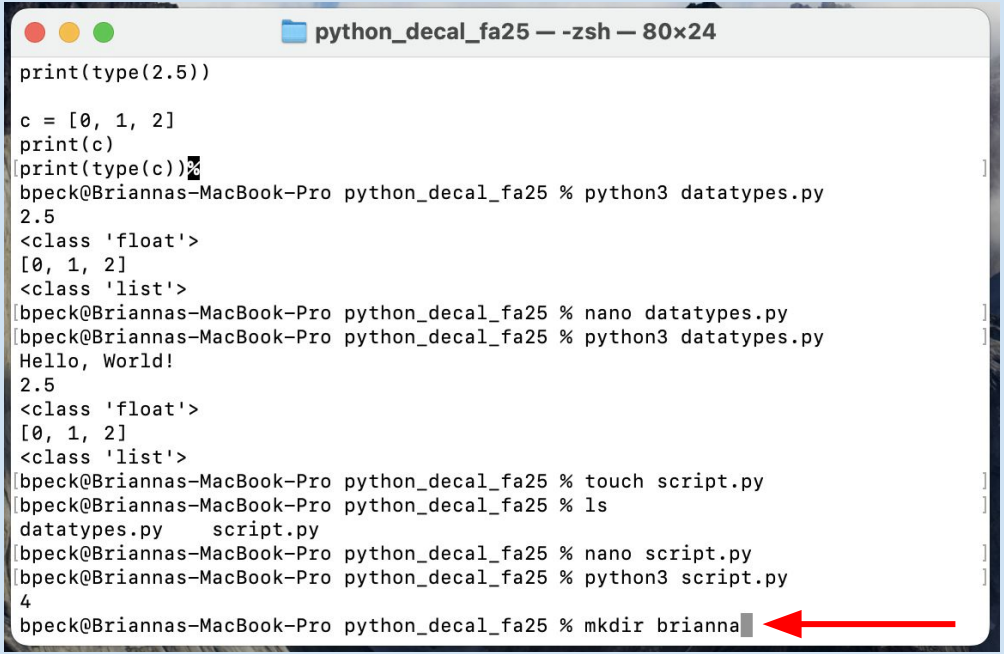
Type: python3 script.py

Make a New File on Terminal

Python just did math for us!



Make a New Folder



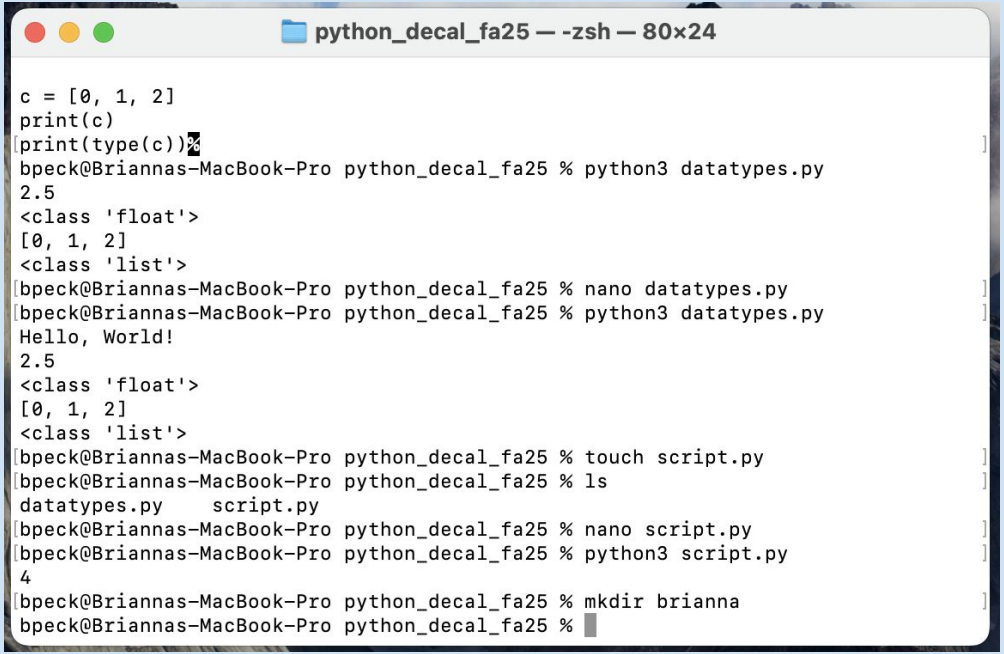
To make a new folder on the terminal, we will use the command: mkdir

Type: mkdir <yourname>

Replace <yourname> with... well... your name. So I would write:

mkdir brianna

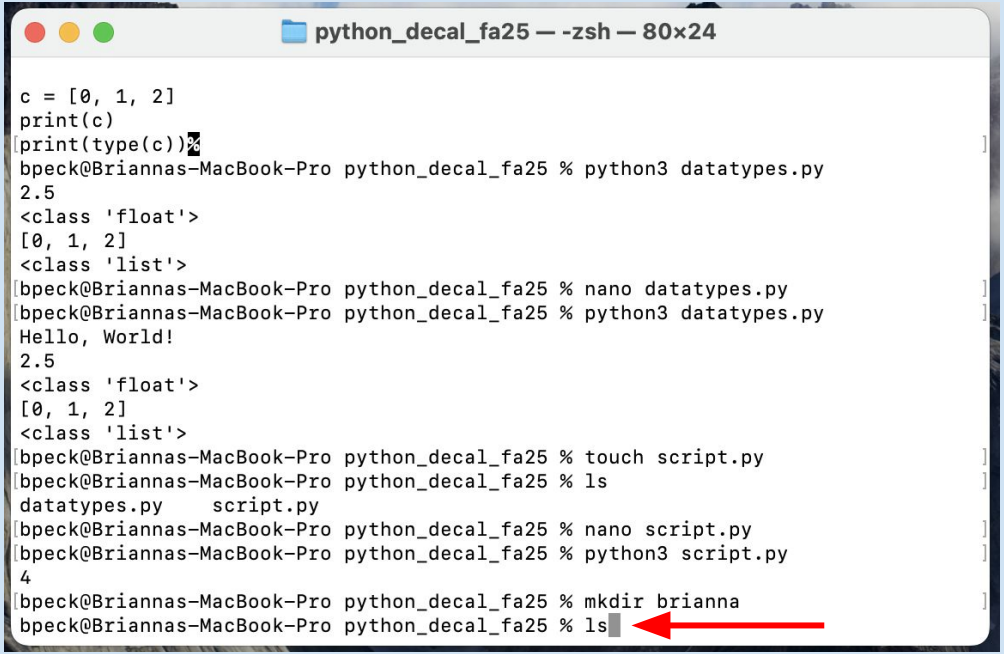
Make a New Folder



This will be another super important folder while you are taking this class

mkdir stands for “make directory”

LIST CONTENTS



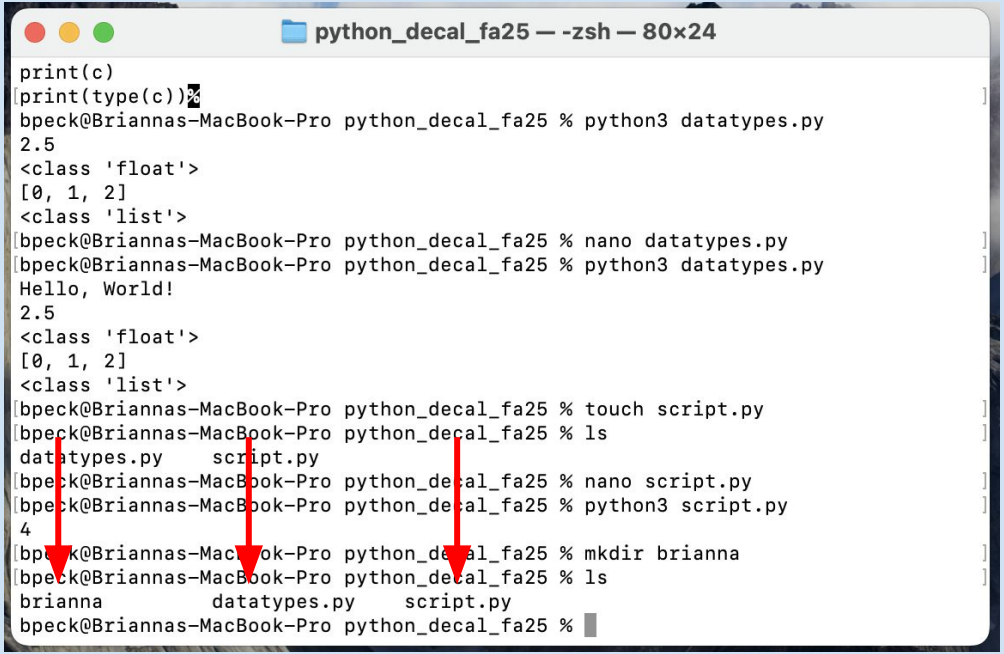
Now when we call ls on in our current working directory...

How many items should we now see?

How many files?

How many folders?

LIST CONTENTS



We see three items!

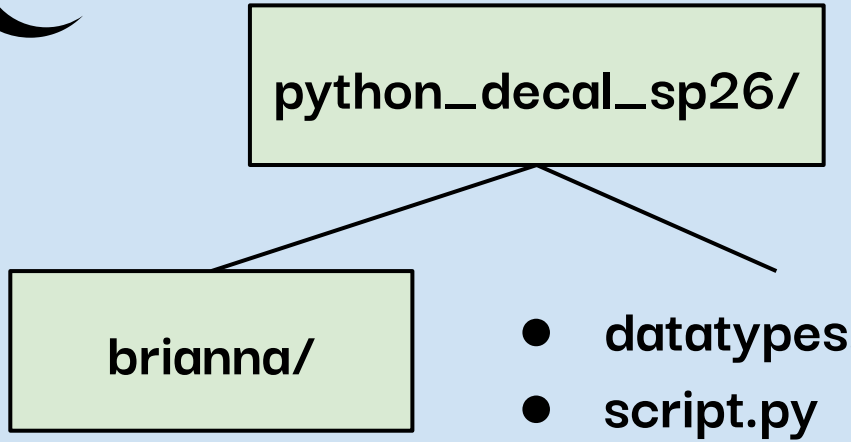
Two files:

- datatypes.py
- script.py

One folder:

- <yourname>

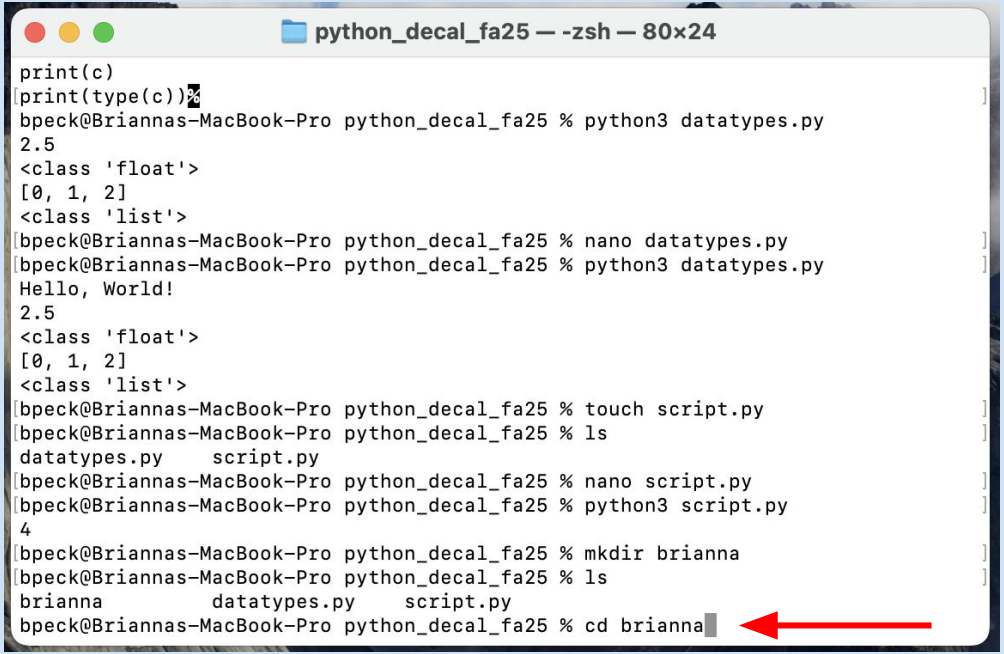
Quick Note: Directory Trees



Because we added a new folder, our directory tree now looks like:

We have a folder and two files in our python_decal_sp26 directory

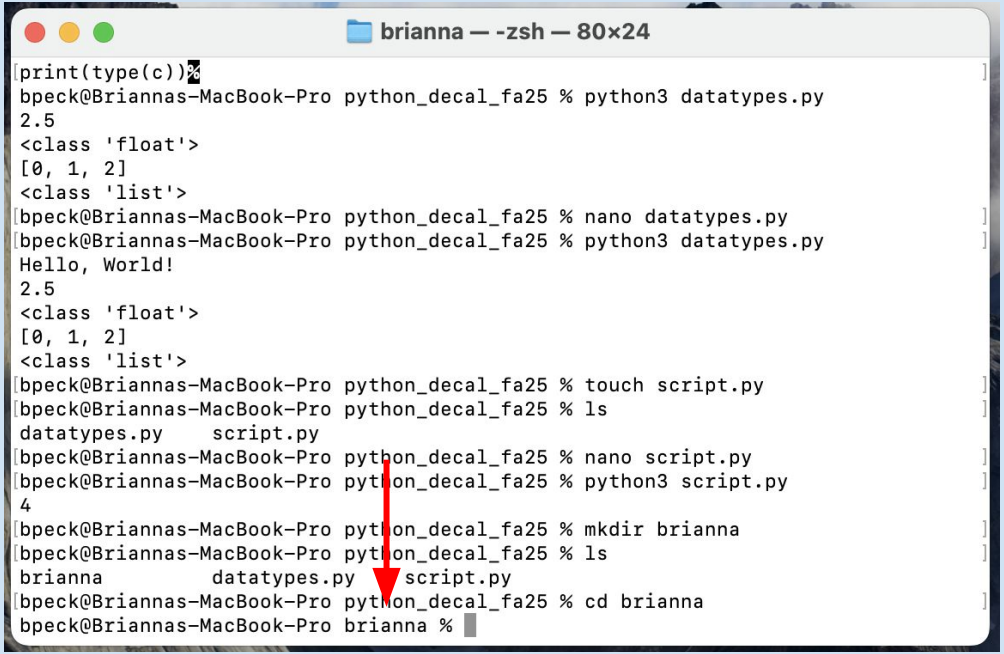
Change Directories



To move into our new directory, we will call: cd

Type: cd <yourname>

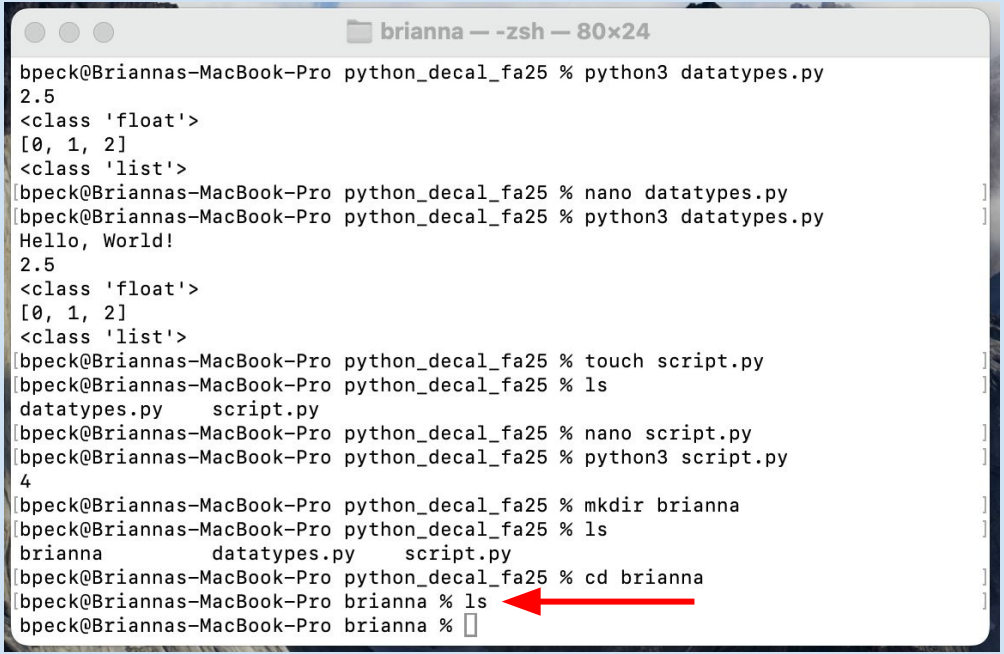
Change Directories



Once again the folder next to the percentage sign changed, this might not be displayed on your terminal

If I call ls in this folder, will my terminal display anything?

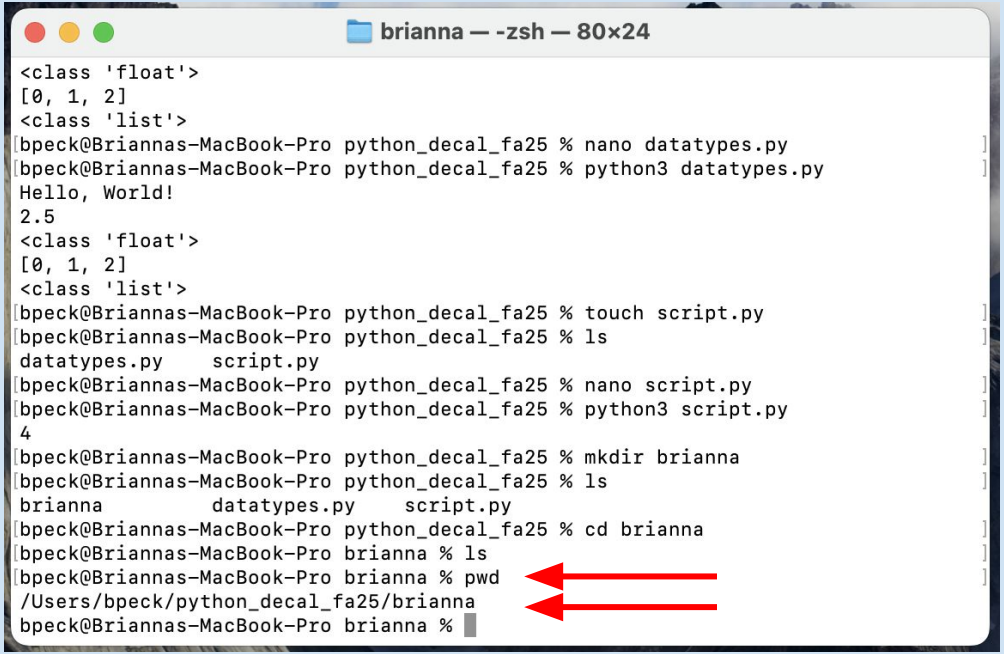
LIST CONTENTS



Nope! It's a brand new folder so it is completely empty

What will our file path look like now? Type: pwd

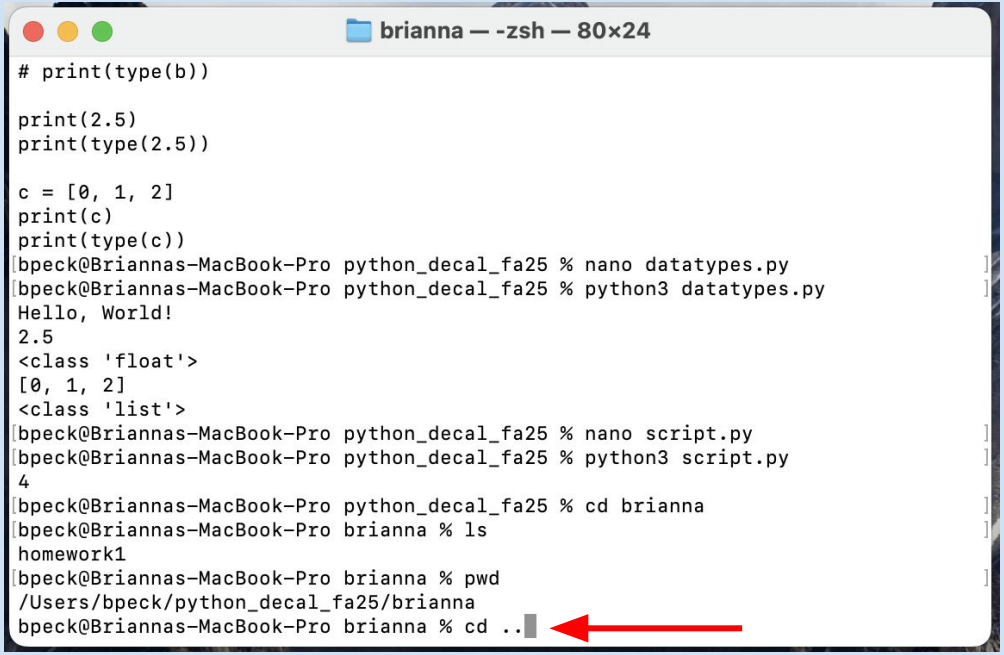
Print Working Directory



Now we have brianna, inside of python_decal_sp26, inside of bpeck, inside of /Users/

You will use this folder as part of your homework

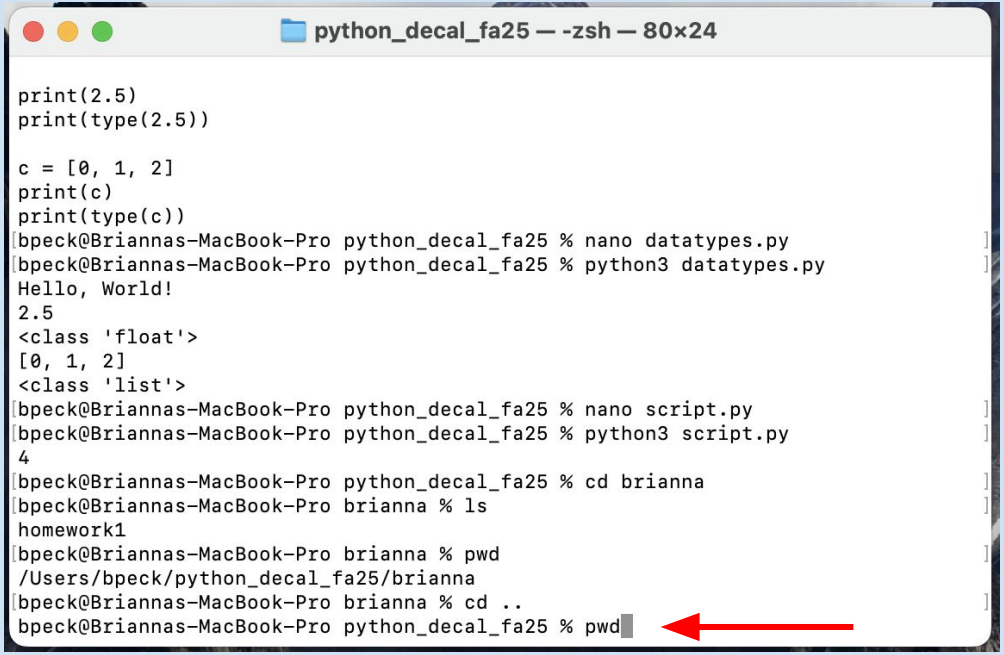
Parent Directory



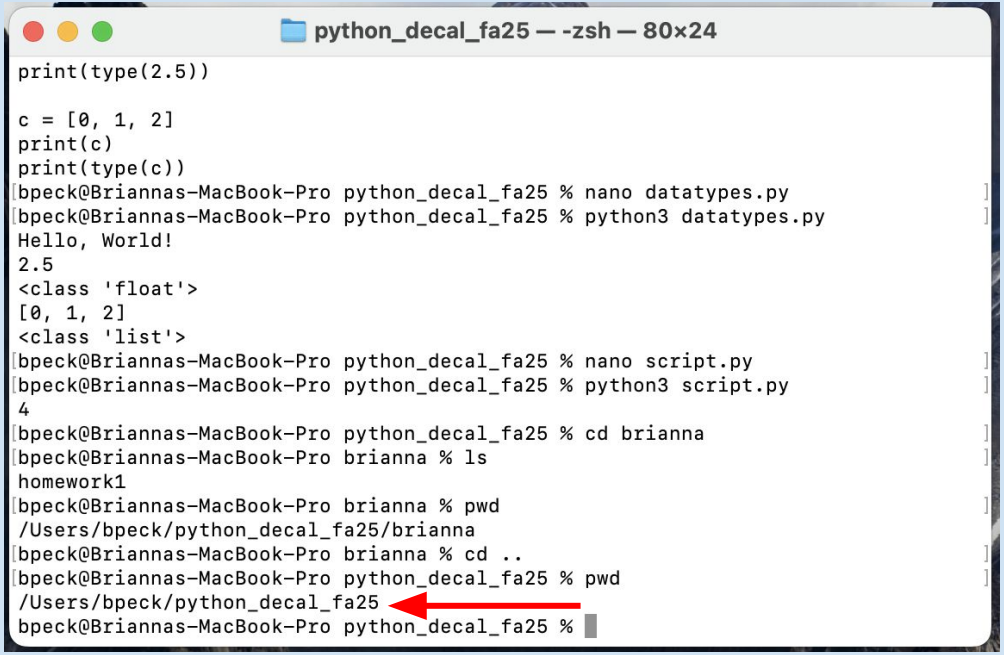
To move up to a parent directory, use the command **cd ..**

Parent Directory

Now if we call pwd



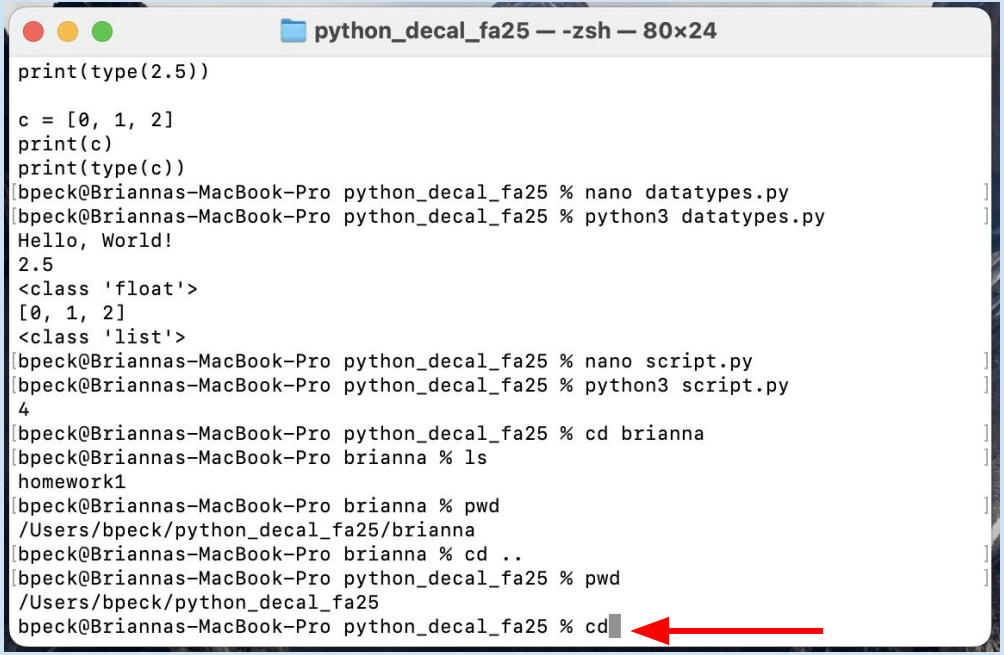
Parent Directory



Our file path changed and we are no longer inside of the yourname folder

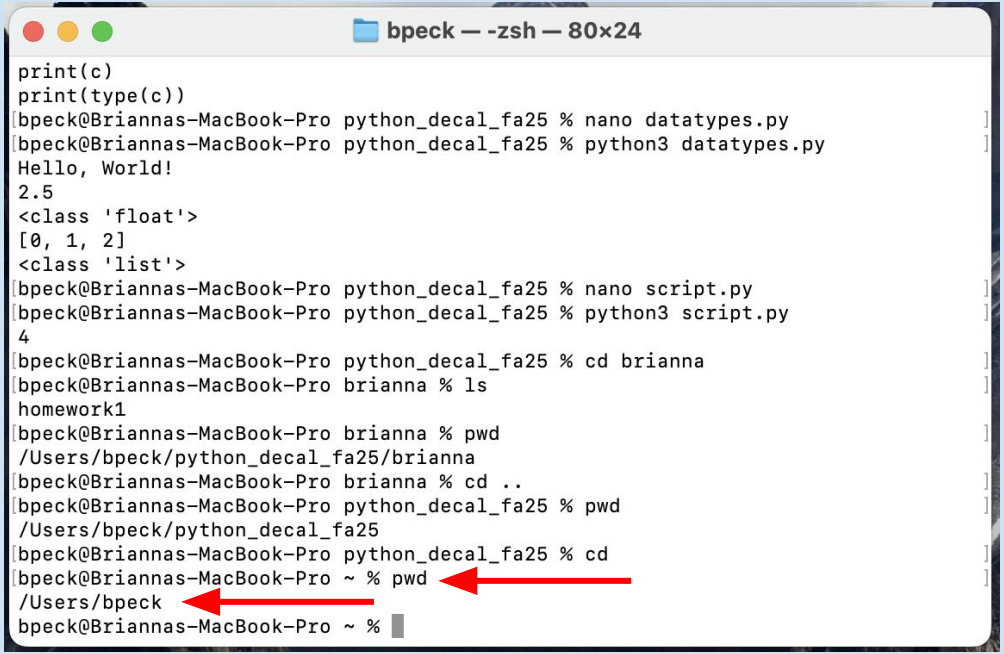
Instead we are in python_decal_sp26 again

Back to Home Directory



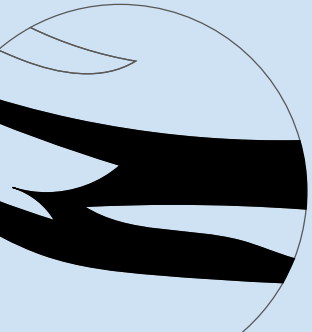
To go all the way back to your home directory, you can call the command cd

Back to Home Directory



Now when we call pwd, we are back in the home directory like when we first opened the terminal

Let's practice



Practice Question

Close those laptops!

Define each of these commands in your own words:

- pwd
- ls
- cat
- cd
- nano

Practice Question Answer

- pwd = prints your the folder you are working in
- ls = lists files and folders inside of you current folder
- cat = shows the inside of a file on the command line
- cd = allows you to move from folder to folder
- nano = allows you to open up a file and edit it

04

Lecture Engagement

Show us what you know!

Lecture Engagement

Fill out the Lecture Check!

Linked here.

Always due two days after class.

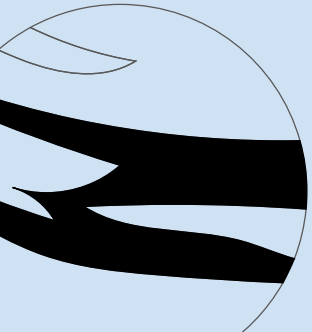
Takes 1-5 minutes to complete.

05

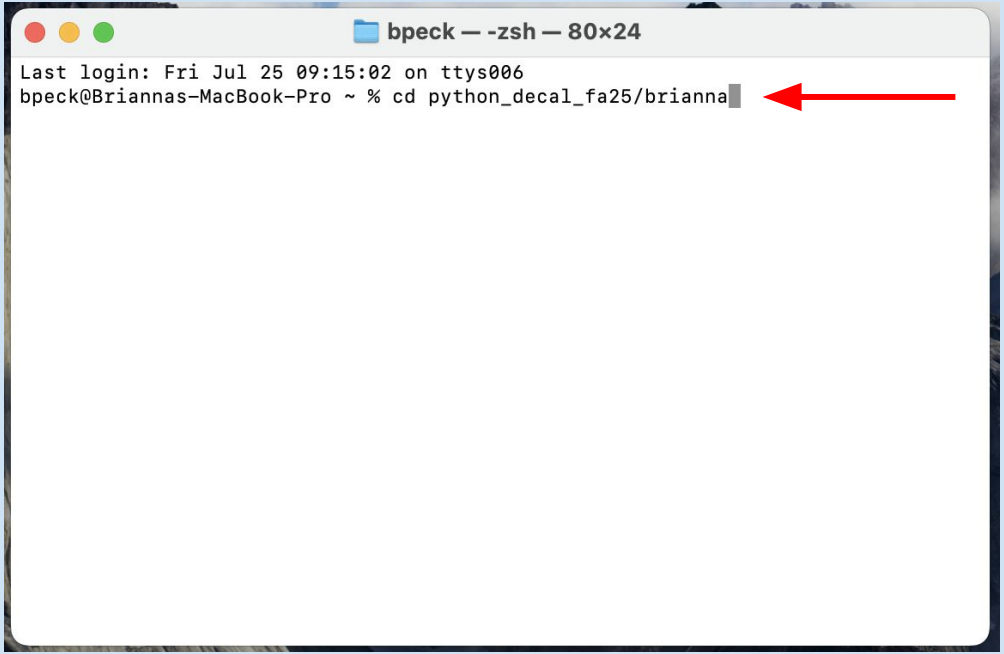
Bonus Content

For helping with homework

Problem #1



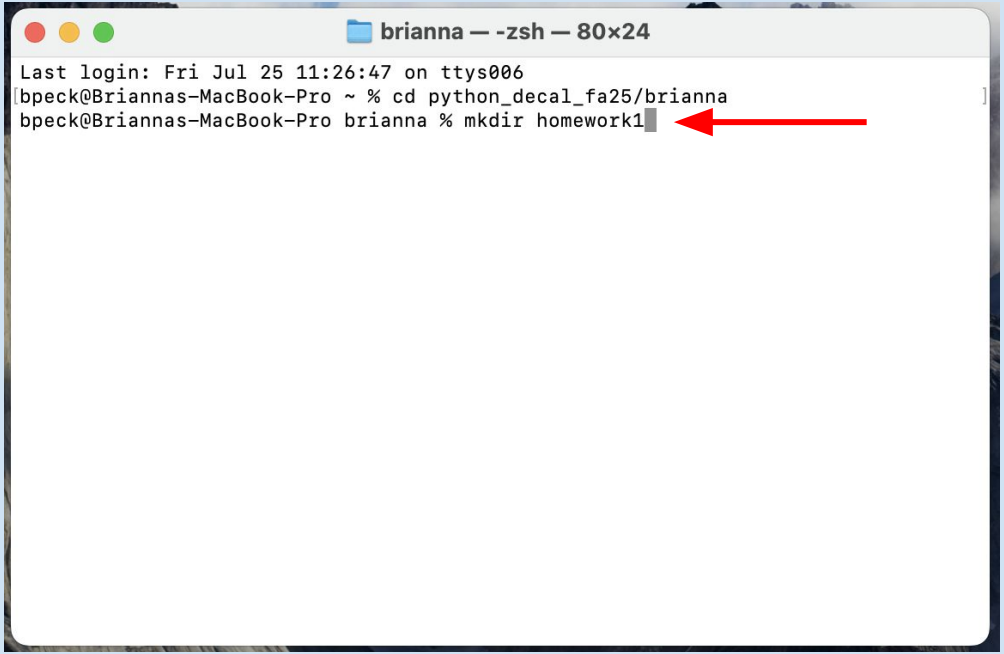
Make another a new folder



In Problem #1 of HW #1 you are asked to make a new folder called homework1 in the yourname directory.

Start by opening your terminal and navigating to the yourname folder.

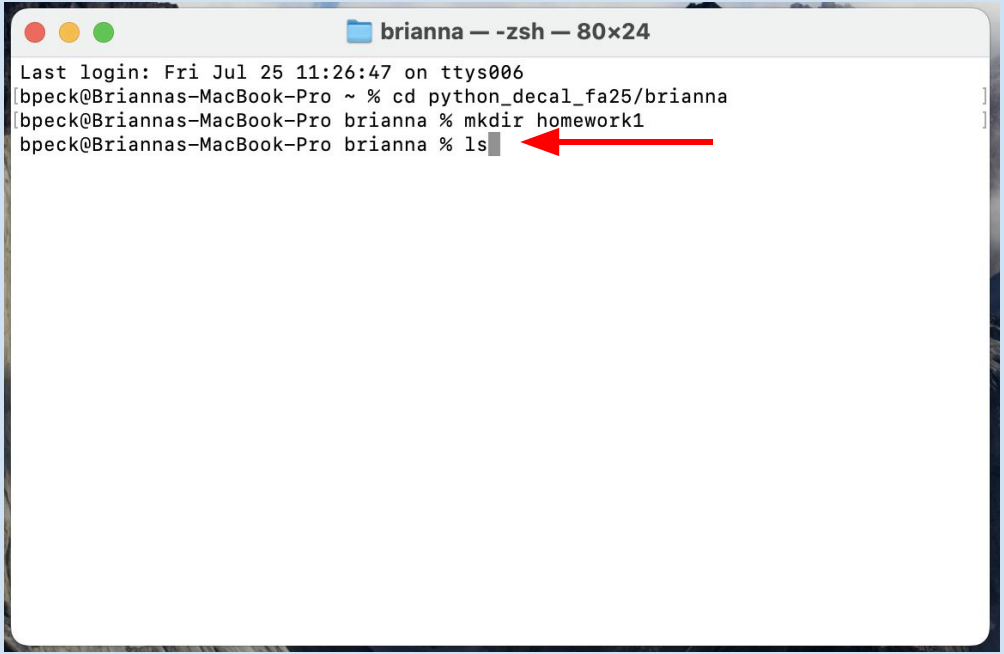
Make another a new folder



To make a new folder inside of the yourname folder, type:

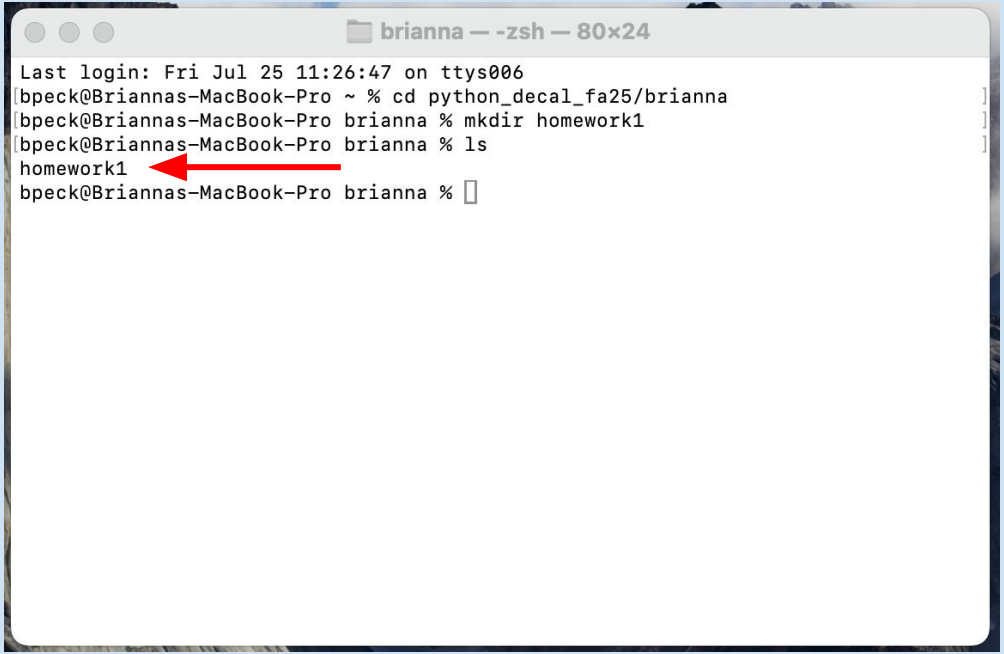
mkdir homework1

LIST CONTENTS



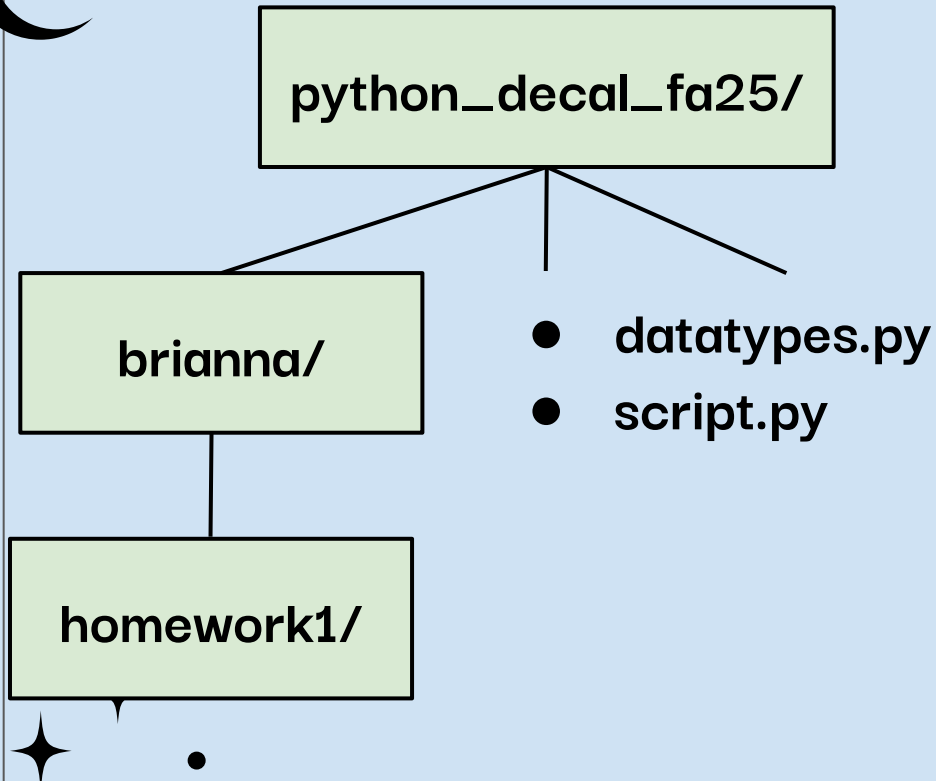
To see the contents of the yourname directory, type ls

LIST CONTENTS



You will now see a new folder called homework1

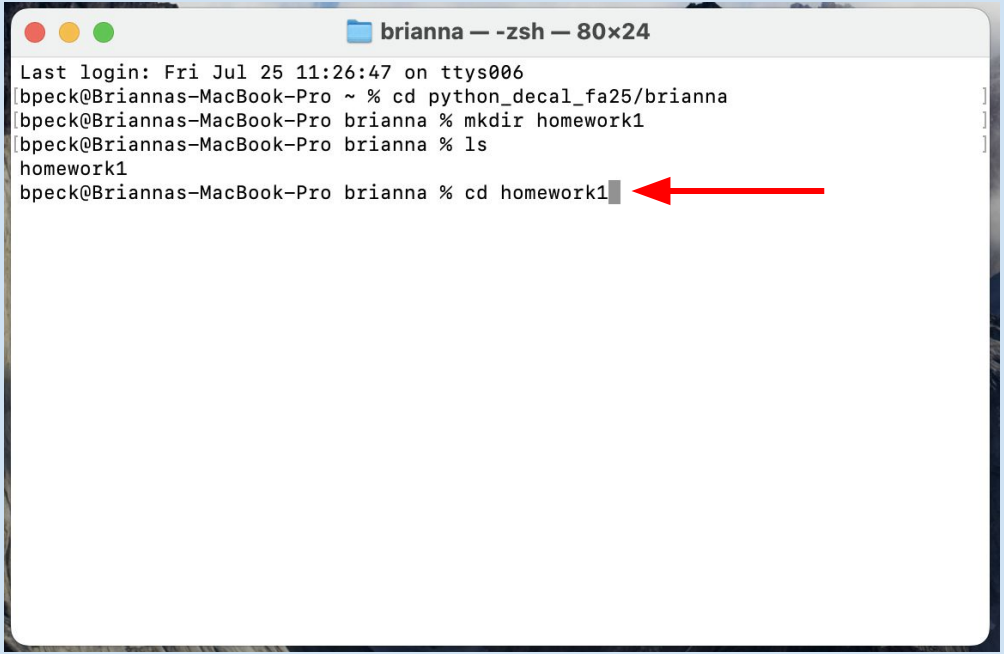
Quick Note: Directory Trees



If you recall the directory tree from lecture, this is now what your directory tree looks like

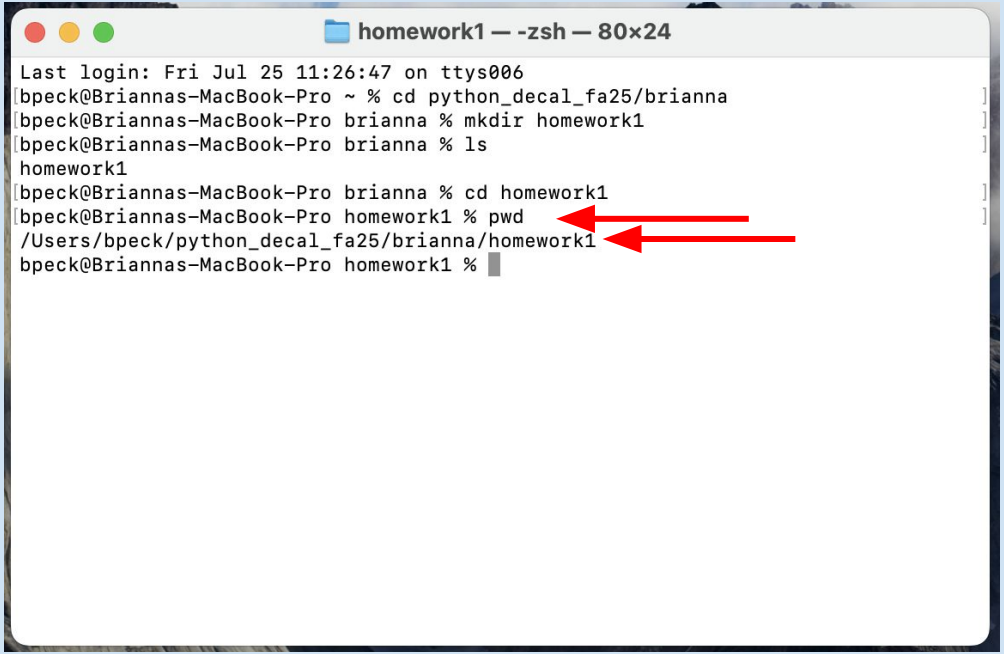
So homework1 is a folder that lives inside of brianna which lives inside of python_decal_fa25

Change Directories



To move into homework1 we will use the command cd

Print Working Directory

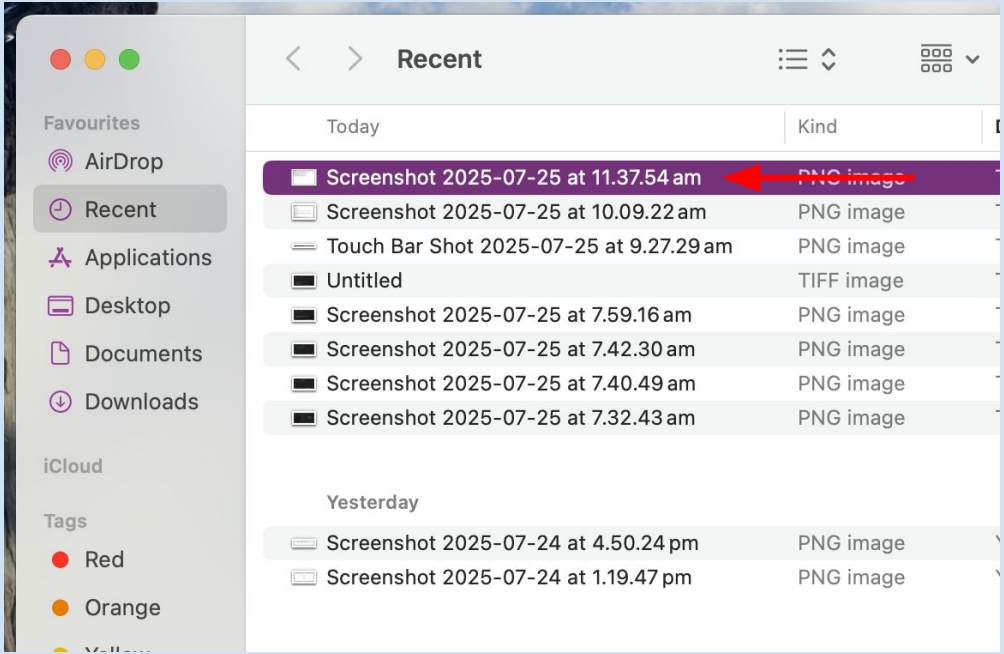


When we call pwd, which prints our current working directory

We see our entire file path

Next, you are asked to take a screenshot of this process. Your picture should look like the one to the left.

save screenshot



To save it in you homework1 folder, here is one way to do that after taking the screenshot.

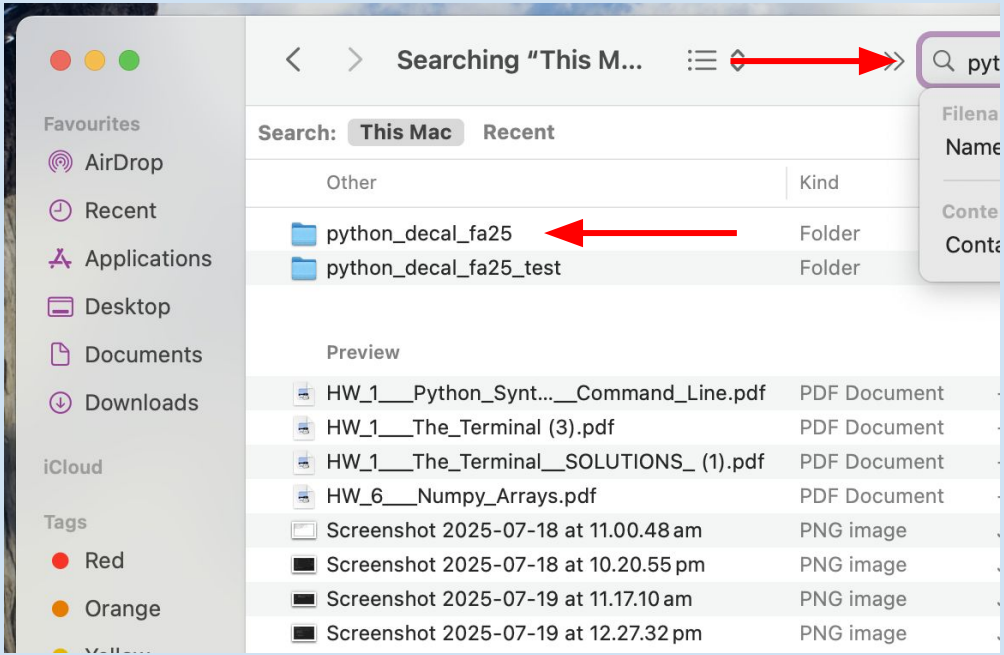
Navigate to your GUI.

Find the screenshot in the Recents/ folder or the Screenshots/ folder

save screenshot

Drag the screenshot out onto your desktop

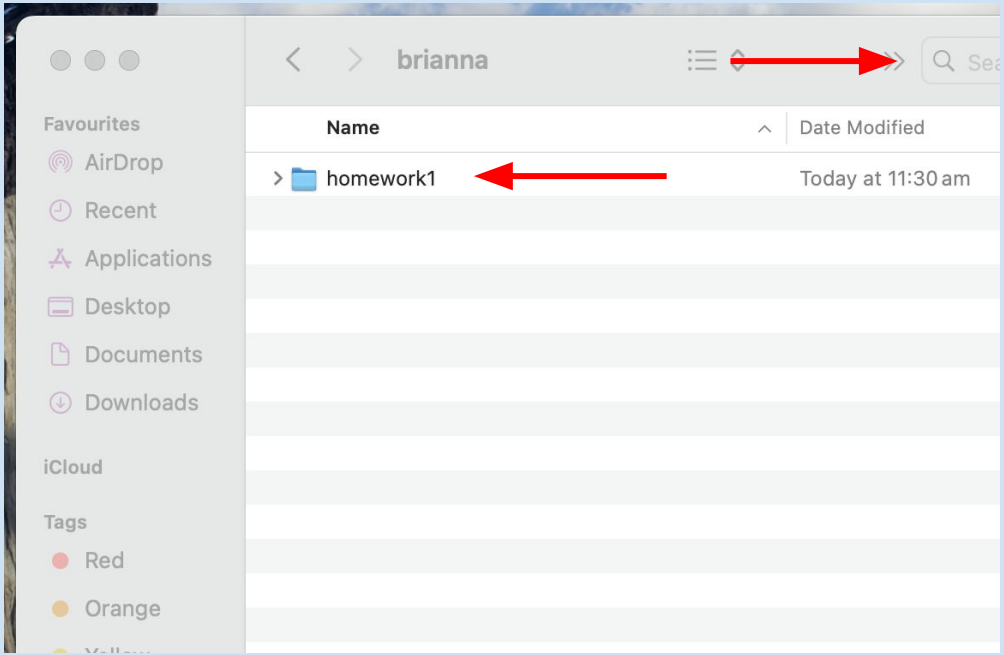
save screenshot



Search for the python_decal_fa25 folder in your GUI

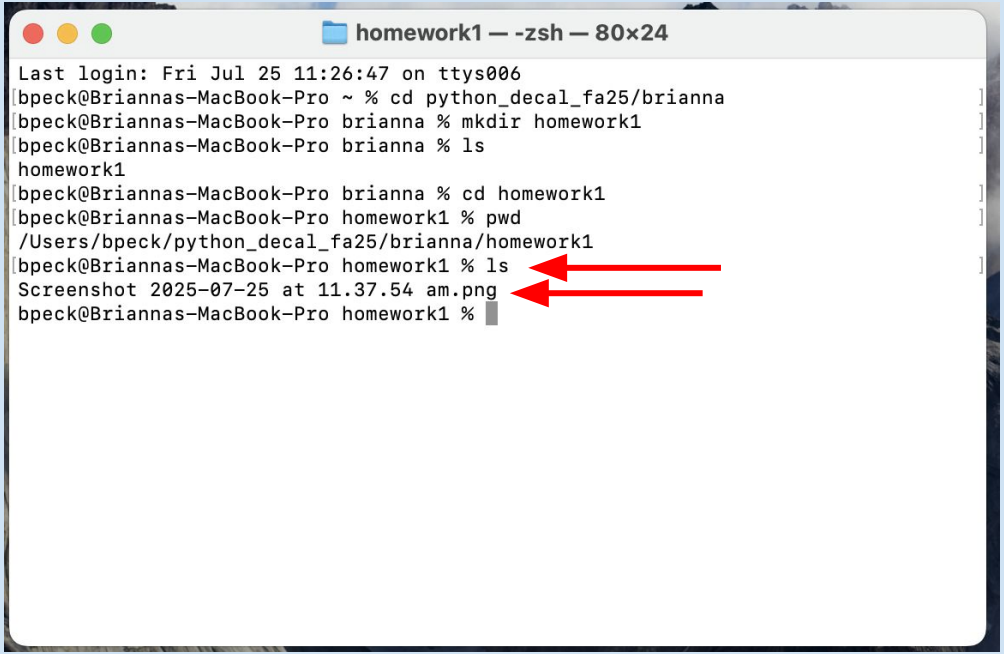
Once you have found it, open up yourname/homework1

save screenshot



Drag and drop your screenshot into this folder!

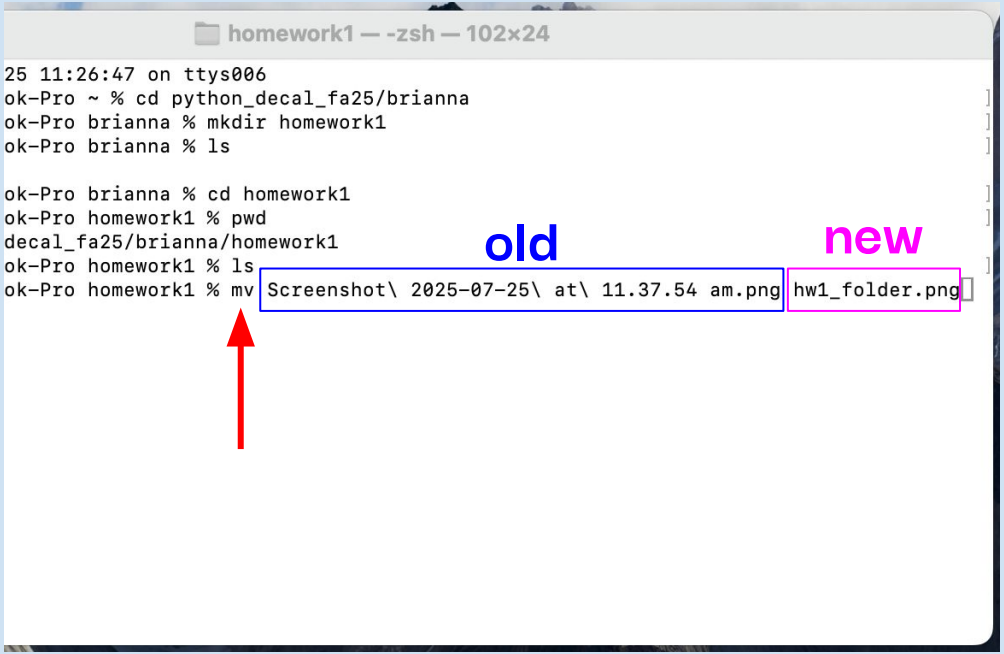
save screenshot



You can confirm the screenshot is in your terminal by calling ls

We could have renamed the file earlier, but let's practice renaming it on the command line

Rename a file with mv

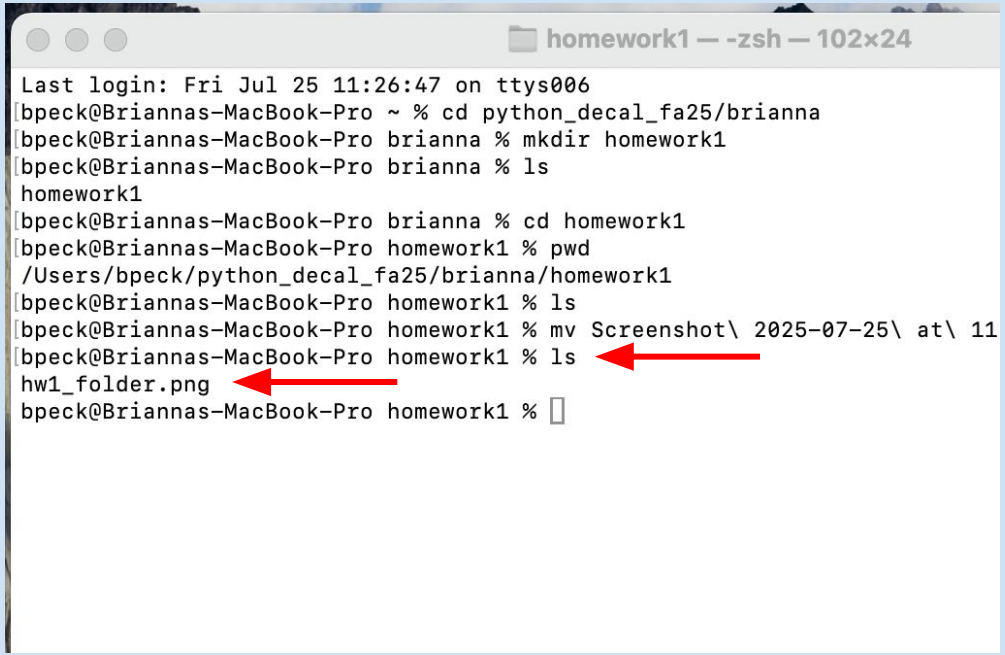


Use the mv command to rename your file (you can also use this to move files in general)

The syntax is:

mv <old_name> <new_name>

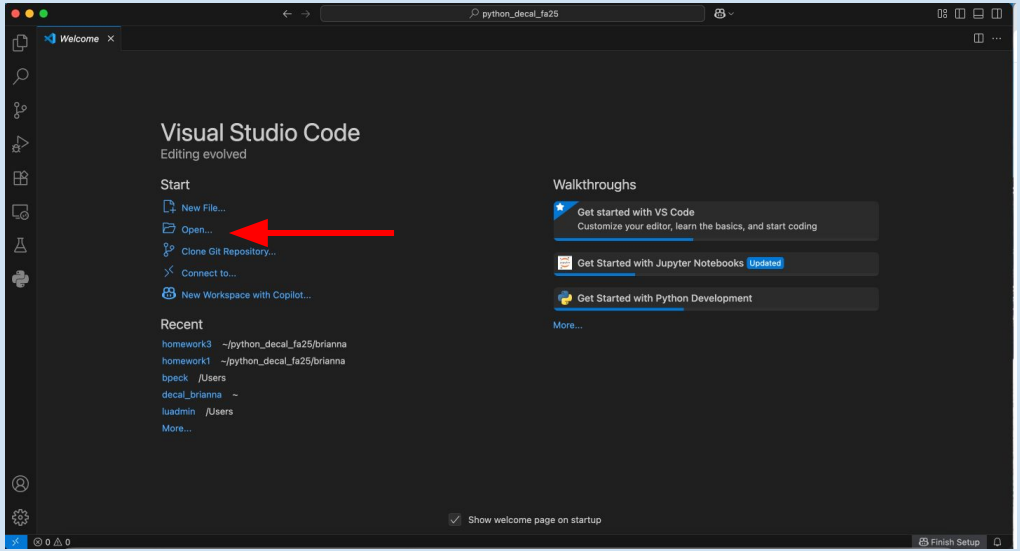
LIST CONTENTS



You can confirm the new renaming by calling ls on the command line

Problem #2

Open VS Code

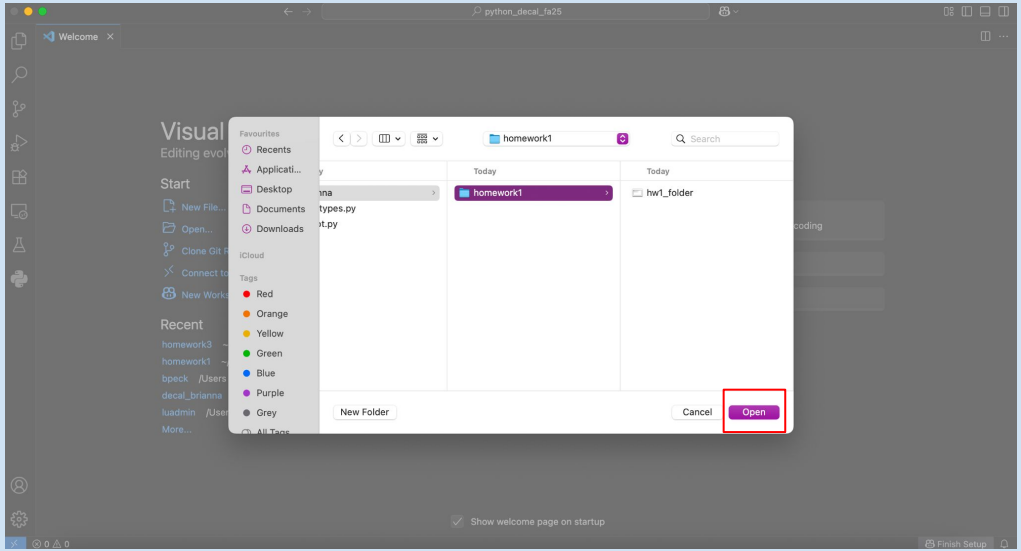


In Problem #2 of HW #1 you are asked to make a new file in VS Code.

To open a new file in VS Code, start by opening the application

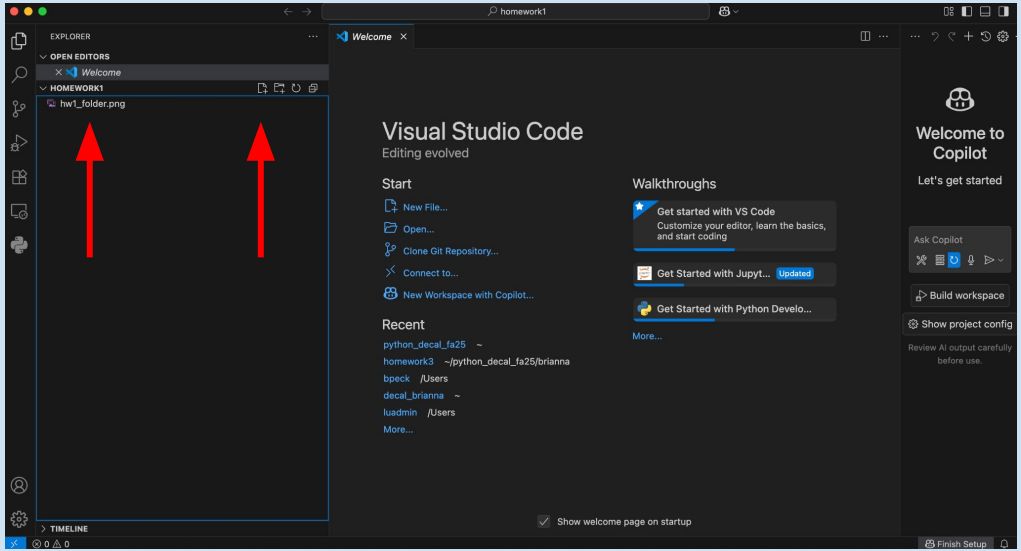
Then clicking on Open...

Open a New File



Navigate to the homework1/ folder you just made on the terminal

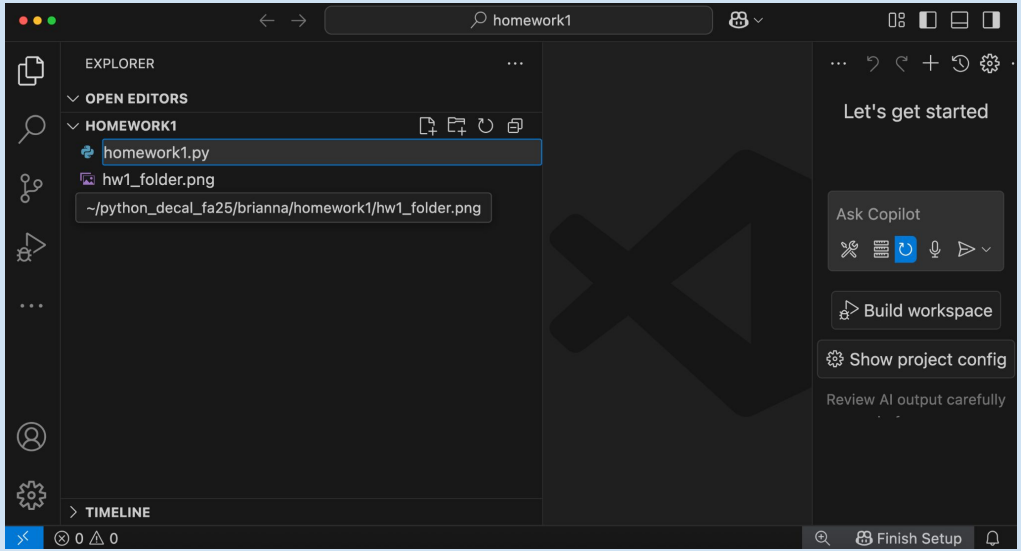
Open a New File



You now see the screenshot we just renamed: hw1_folder

Then click on the New File... icon the second red arrow is pointing to

Open a New File

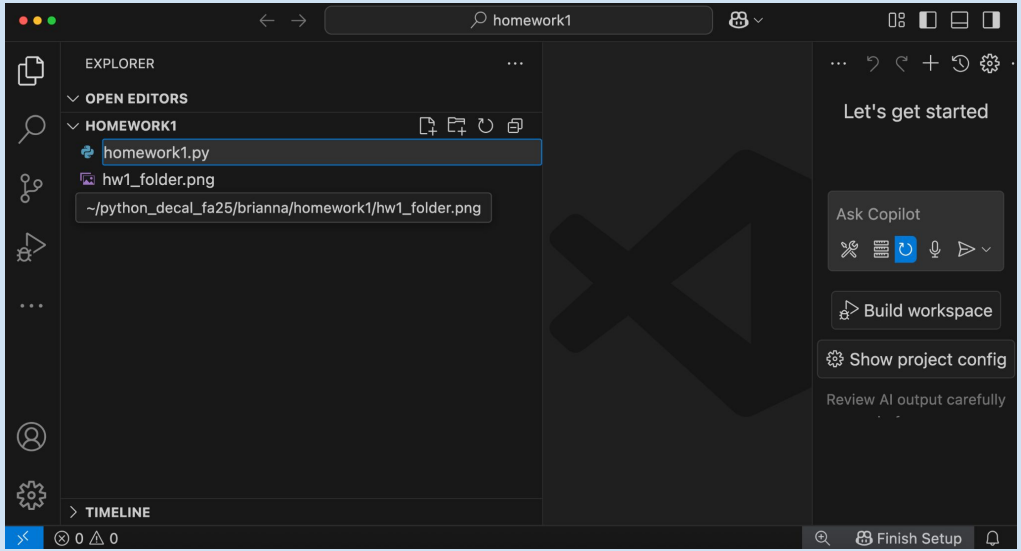


Name the new file:

homework1.py

Then start the other problems!

Make another a new folder



Name the new file:

homework1.py