

Homework 5

also posted on Brightspace
due next Wed (Oct 12) in class

Homework5.pdf (written description)

Homework5.ipynb

BoysBW.jpg

Homeworks (in general)

note: if a homework says to calculate something or to show something or to verify something, then you do need to print something (e.g., to the notebook) or display something (e.g., like a matplotlib plot) that displays the result of that calculation or shows the thing you have been asked to show in the question

Jason will verify results or intermediate calculations, but the primary elements that answer a question, need to be explicitly printed out or displayed by your code

download from Brightspace

`Functions.ipynb`

`Week7.zip` (contains .py files)

download Week7.zip file from Brightspace
(contains .py files)

copy to this folder to the folder (and unzip) that you
created when setting up and testing PyCharm

Functions

Functions.ipynb

A Whirlwind Tour of Python, Jake VanderPlas
Chapter 8: Defining Functions

<https://jakevdp.github.io/WhirlwindTourOfPython/08-defining-functions.html>

https://docs.python.org/3/reference/compound_stmts.html#function-definitions

<https://realpython.com/defining-your-own-python-function/>

functions

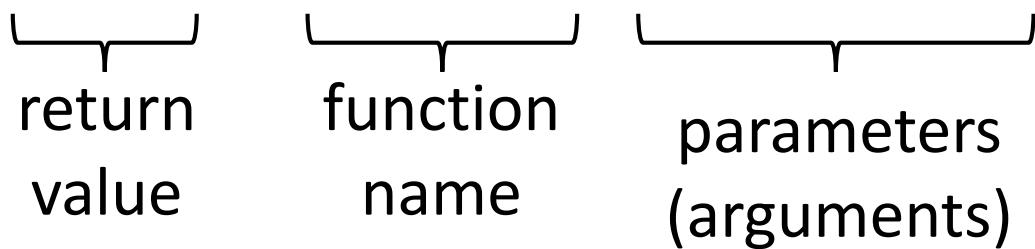
Why do we use functions?

- reuse code over and over again
- makes code far more readable
- easier to write modular code
- easier to debug modular code

functions in Python

calling a function

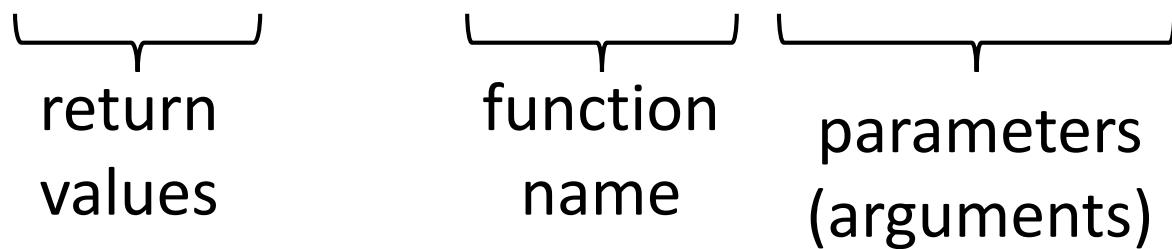
```
a = myfun(x, y, z)
```



functions in Python

calling a function

(a, b) = myfun(x, y, z)



functions in Python

calling a function

```
r = myfun(x, y, z)
```

tuple
(> 1 thing) function name parameters
 (arguments)

```
a = r[0]
```

```
b = r[1]
```

functions in Python

```
import mymodule as mym
```

mymodule.py file

-
-
-

```
r = mym.myfun(x, y, z)
```

functions in Python

```
def myfun(d, e, f):
```

expressions

```
    return (n, m)
```

•

•

•

```
r = myfun(x, y, z)
```

functions in Python

order matters (for arguments without names)

```
def myfun(d, e, f):    colon required  
indenting expressions  
    return (n, m)  
  
.  
.  
.  
  
r = myfun(x, y, z)
```

functions in Python

order matters

```
def myfun(d, e, f):
```

expressions

```
return (n, m)
```

return required
if returning values

if returning more than
one variable, returns
as a tuple

•

•

•

```
r = myfun(x, y, z)
```

functions in Python

order matters

```
def myfun(d, e, f):
```

expressions

```
return (n, m)
```

return required
if returning values

if returning more than
one variable, returns
as a tuple

•

•

•



```
(a, b) = myfun(x, y, z)
```

argument / parameter passing in Python

arguments of base types (int, float, string) and immutable types (tuple) are passed by value

arguments of compound types (lists, dicts, sets, numpy arrays) are "kind of" passed by reference

review various cases on Functions.ipynb

variable scope in Python

any variable not inside of a function is a "global" variable (at least it can be accessed anywhere)

functions inside a function are "local"

variables can be explicitly declared `global`

in this class, I do not want to see any use of global variables inside of functions; arguments should be passed to the function and variables should be returned from the function, with local variables used inside the function

passing a function to a function

a function in Python is just another object and can be passed to another function

```
def myfun(func, x):  
    y = func(x)  
    return (y)
```

```
def myop(x):  
    y = x**2  
    return (y)
```

```
print(myfun(myop, 2))
```

default function arguments

optional parameters (arguments) with default values

```
def psiphys(x, alpha=1., beta=2., gam=.5, lam=.1):  
    F = 1 - np.exp(-(x/alpha)**beta)  
    psi = gam + (1. - lam - gam)*F  
    return (psi)
```

```
psi2 = psiphys(x, lam=.2, beta=1.)
```

default function arguments

* makes the argument names required past the *

```
def psiphys(x, *, alpha=1., beta=2., gam=.5, lam=.1):  
    F = 1 - np.exp(-(x/alpha)**beta)  
    psi = gam + (1. - lam - gam)*F  
    return (psi)
```

```
psi2 = psiphys(x, lam=.2, beta=1.)
```

doc strings (in functions)

```
def psiphys(x, *, alpha=1., beta=2., gam=.5, lam=.1):  
    '''Computes psychophysical function
```

Arguments:

x (numpy array):	point at which to evaluate psi(x)
alpha (float):	scale parameter of the Weibull
beta (float):	shape parameter of the Weibull
gam (float):	chance (floor on performance)
lam (float):	lapse rate (ceil on performance)

Returns:

psi (numpy array): psychophysical function psi(x)

'''

```
F = 1 - np.exp(-(x/alpha)**beta)  
psi = gam + (1. - lam - gam)*F  
return (psi)
```

```
help(psiphys)
```

type hints

```
def psiphys(x      : np.ndarray,
            *,
            alpha : float = 1.,
            beta  : float = 2.,
            gam   : float = .5,
            lam   : float = .1) -> np.ndarray:
    '''Computes psychophysical function
```

Arguments:

x (numpy array):	point at which to evaluate psi(x)
alpha (float):	scale parameter of the Weibull (optional)
beta (float):	shape parameter of the Weibull (optional)
gam (float):	chance (floor on performance) (optional)
lam (float):	lapse rate (ceil on performance) (optional)

Returns:

```
psi (numpy array): psychophysical function psi(x)
...
F = 1 - np.exp(-(x/alpha)**beta)
psi = gam + (1. - lam - gam)*F
return (psi)
```


Python (.py) Files, Modules, and Debugging in PyCharm

<https://www.jetbrains.com/help/pycharm/>

download Week7.zip file from Brightspace
(contains .py files)

copy to this folder to the folder you
created when setting up and testing PyCharm

Jupyter Notebooks are great as Scripts

- Jupyter Notebooks are great for trying out snippets of Python code (as an easy-to-use Python Console)
- “Scripts” are a series of commands (though you can define functions within them). Scripts are not stand-alone programs nor are they modules/packages that can be used by other programs.
- Jupyter Notebooks are fantastic for fully documenting a data analysis / modeling pipeline. They are a computational equivalent of a paper lab notebook. Preferred over more ad hoc (and often undocumented) analysis pipelines too commonly used.

Jupyter Notebooks are not so great ...

- Hard to debug code (.py files in fully-functional IDEs are way better) - `print` statements and weak debugging magics (we did not talk about) only go so far
- Harder to share code (copy and paste from one Notebook to another)
- Git/Github are possible (but less natural, more cumbersome, more as backups than repositories with notebooks)
- Cannot produce general purpose modules and packages
- Often not appropriate for large-scale analysis packages, complex modeling and simulation (unless called from Jupyter Notebook), or standalone applications

Python .py files (ASCII text file)



The image shows a screenshot of a Python code editor. The title bar of the window reads "MyPsiPhys.py". The code itself is a Python script with the following content:

```
import numpy as np
import matplotlib.pyplot as plt

# %%

def psiphys(x      : np.ndarray,
            *,
            alpha   : float = 1.,
            beta    : float = 2.,
            gam     : float = .5,
            lam     : float = .1) -> np.ndarray :
    """Computes psychophysical function

    Arguments:
        x (numpy array): point at which to evaluate psi(x)
        alpha (float): scale parameter of the Weibull
        beta (float): shape parameter of the Weibull
        gam (float): chance (floor on performance)
        lam (float): lapse rate (ceil on performance)

    Returns:
        psi (numpy array): psychophysical function psi(x)
    """
    F = 1 - np.exp(-(x / alpha) ** beta)
    psi = gam + (1. - lam - gam) * F
    return (psi)

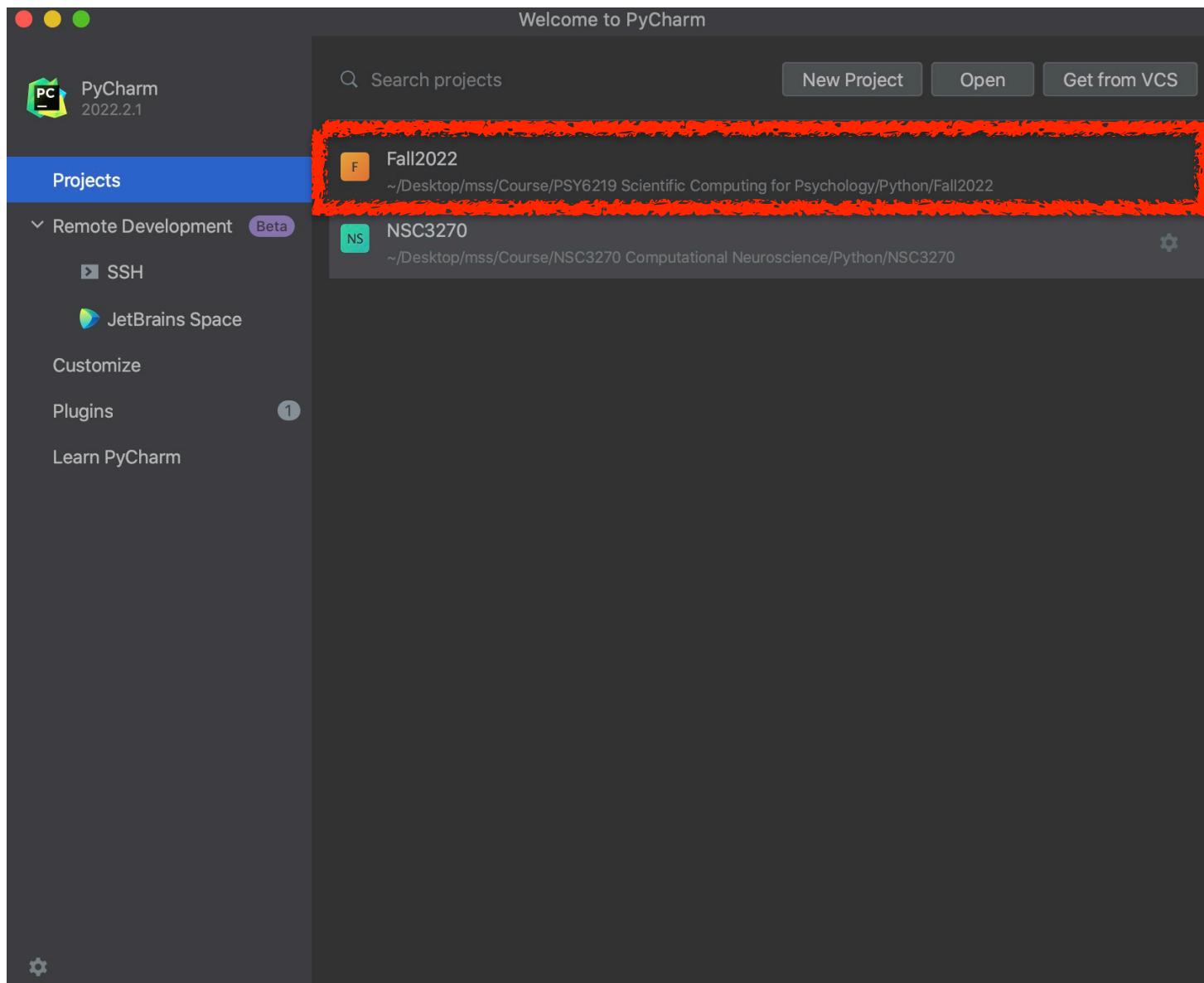
# %%

x = np.arange(0, 5, .01)
psi1 = psiphys(x)
psi2 = psiphys(x, lam=.2, gam=.4)

# %%

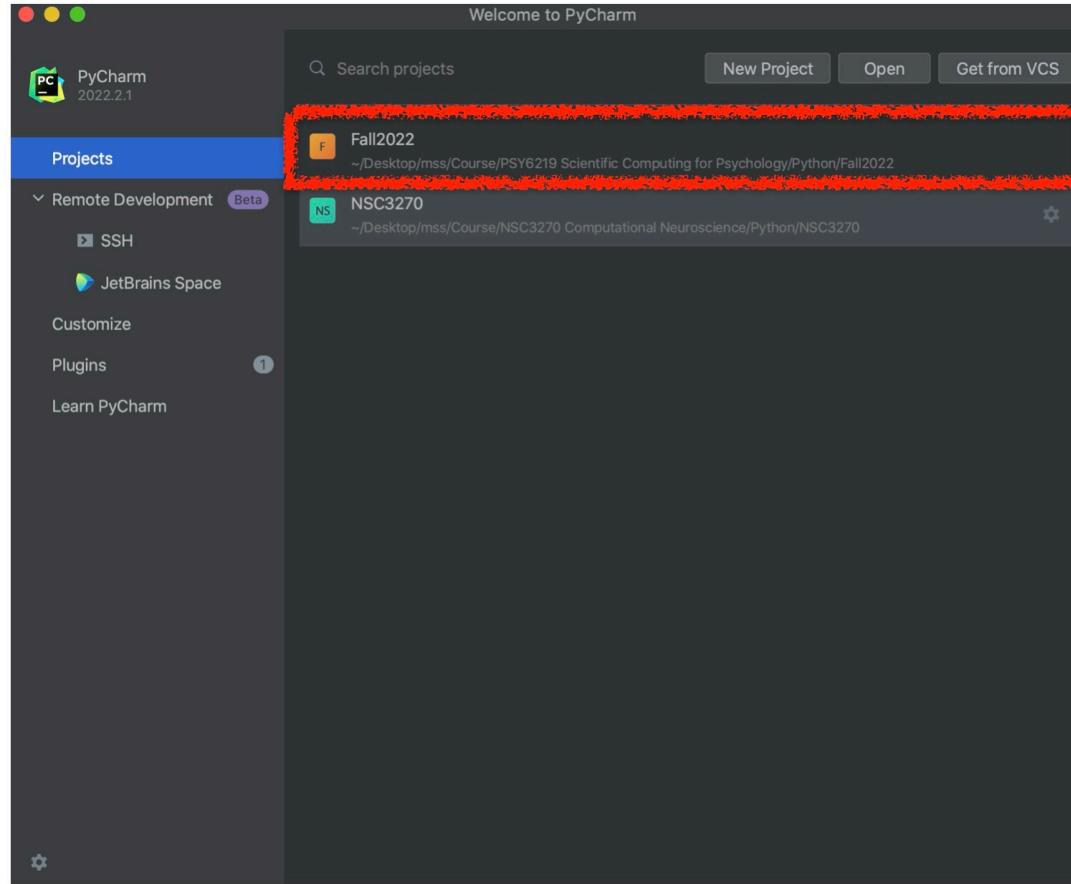
plt.plot(x, psi1, 'r-', x, psi2, 'g-')
plt.xlabel("stimulus manipulation")
plt.ylabel("accuracy")
plt.title("psychophysical function")
plt.ylim(.3, 1)
plt.show()
```

starting PyCharm



it may just open right up to your project
(you get this when a project was closed)

starting PyCharm



- a "Project" resides in a folder and can include many .py files
- it can be one interconnected set of code (many .py files)
- or a bunch of independent folders and .py files sharing the same environment (which is what I do for much of the course code)

The screenshot shows the PyCharm IDE interface with the following details:

- Project Bar:** PyCharm, File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help.
- Toolbar:** Standard icons for file operations like Open, Save, and Run.
- File:** Fall2022 – TestPsiPhys.py
- Project Structure:** Fall2022 (~/Desktop/mss/Course/PSY6219), Week1, Week7, MyPsiPhys.py, PsiPhys.py, TestPsiPhys.py, External Libraries, Scratches and Consoles.
- Code Editor:** TestPsiPhys.py (Line 19). The code imports numpy, matplotlib.pyplot, and psiphys, then plots two psychophysical functions (psi1 and psi2) against a stimulus manipulation variable (x).
- SciView:** Data, Plots. A message says "Run Python Console or Debugger to view available data".
- Documentation:** Fall2022 x, No documentation found.
- Python Console:** Shows the Python environment setup and a command prompt (In [2]).
- Bottom Status Bar:** Version Control, Problems, TODO, Terminal, Python Console, Python Packages, 1:1 LF UTF-8 4 spaces, Python 3.8 (PSY4219-Fall2022), AWS: No credentials selected.

this should be the environment
you created before

PyCharm File Edit View Navigate Code Refactor Run Tools VCS Window Help

Fall2022 – TestPsiPhys.py

SciView: Data Plots

Run Python Console or Debugger to view available data

Documentation: Fall2022 x

No documentation found.

Project

Structure

External Libraries

Scatches and Consoles

TestPsiPhys.py

Fall2022 ~/Desktop/mss/Course/PSY6219

Week1

Week7

MyPsiPhys.py

PsiPhys.py

TestPsiPhys.py

TestPsiPhys.py

```
#  
# test calling psiphys module  
#  
import numpy as np  
import matplotlib.pyplot as plt  
import psiphys  
  
x = np.arange(0, 5, .01)  
psi1 = psiphys.psiphys(x, lam=.1, gam=.6)  
psi2 = psiphys.psiphys(x, lam=.2, gam=.4)  
  
plt.plot(x, psi1, 'r-', x, psi2, 'g-')  
plt.xlabel("stimulus manipulation")  
plt.ylabel("accuracy")  
plt.title("psychophysical function")  
plt.ylim(.3, 1)  
plt.show()
```

Python Console x

import sys; print('Python %s on %s' % (sys.version, sys.platform))
sys.path.append(['/Users/palmerit/Desktop/mss/Course/PSY6219 Scientific Computing for Psychology/Python/Fall2022', '/Users/palmerit/Desktop'])

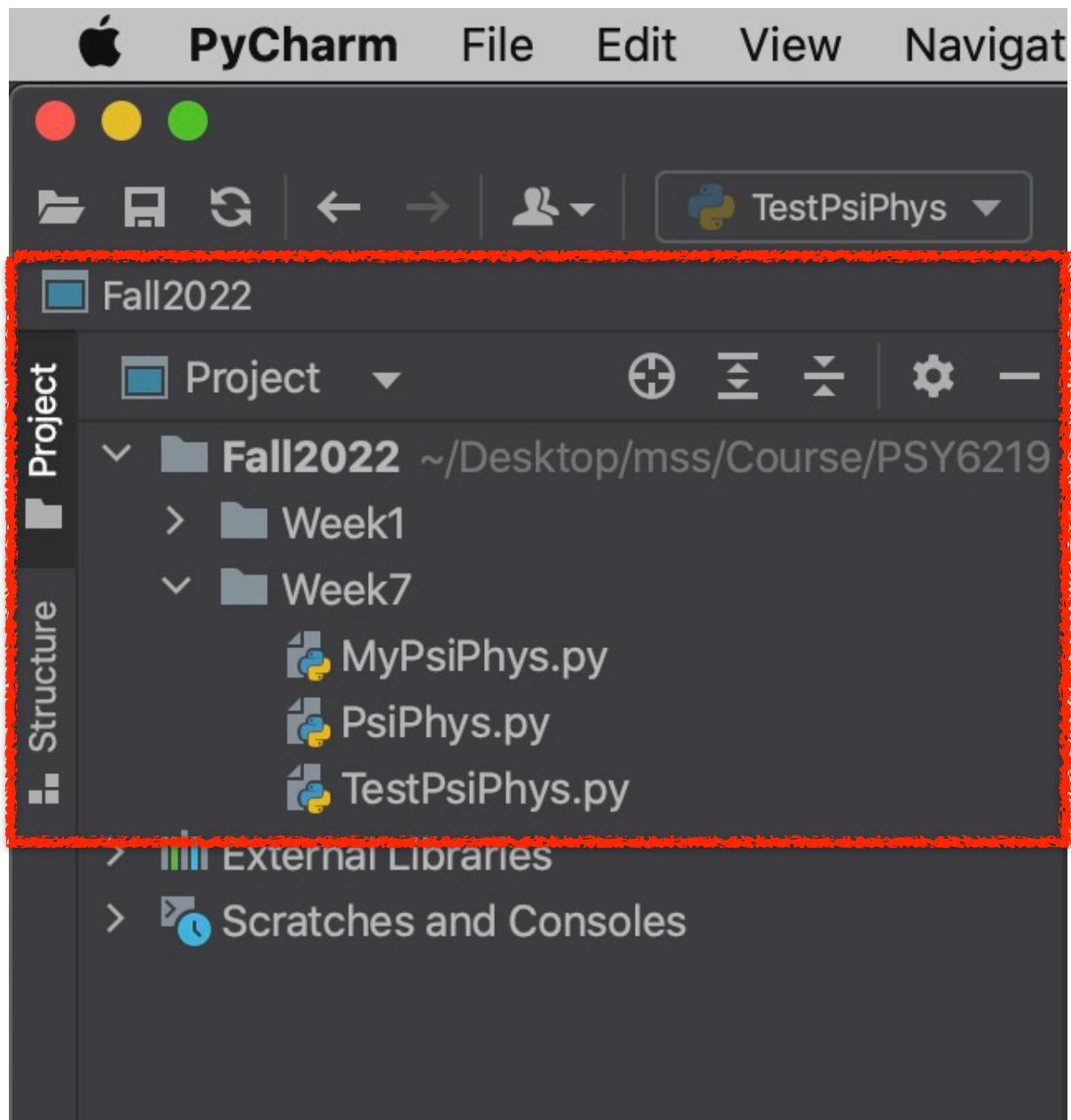
Python 3.8.13 | packaged by conda-forge | (default, Mar 25 2022, 06:06:49)
Type 'copyright', 'credits' or 'license' for more information
IPython 8.4.0 -- An enhanced Interactive Python. Type '?' for help.
PyDev console: using IPython 8.4.0

Python 3.8.13 | packaged by conda-forge | (default, Mar 25 2022, 06:06:49)
[Clang 12.0.1] on darwin

In [2]:

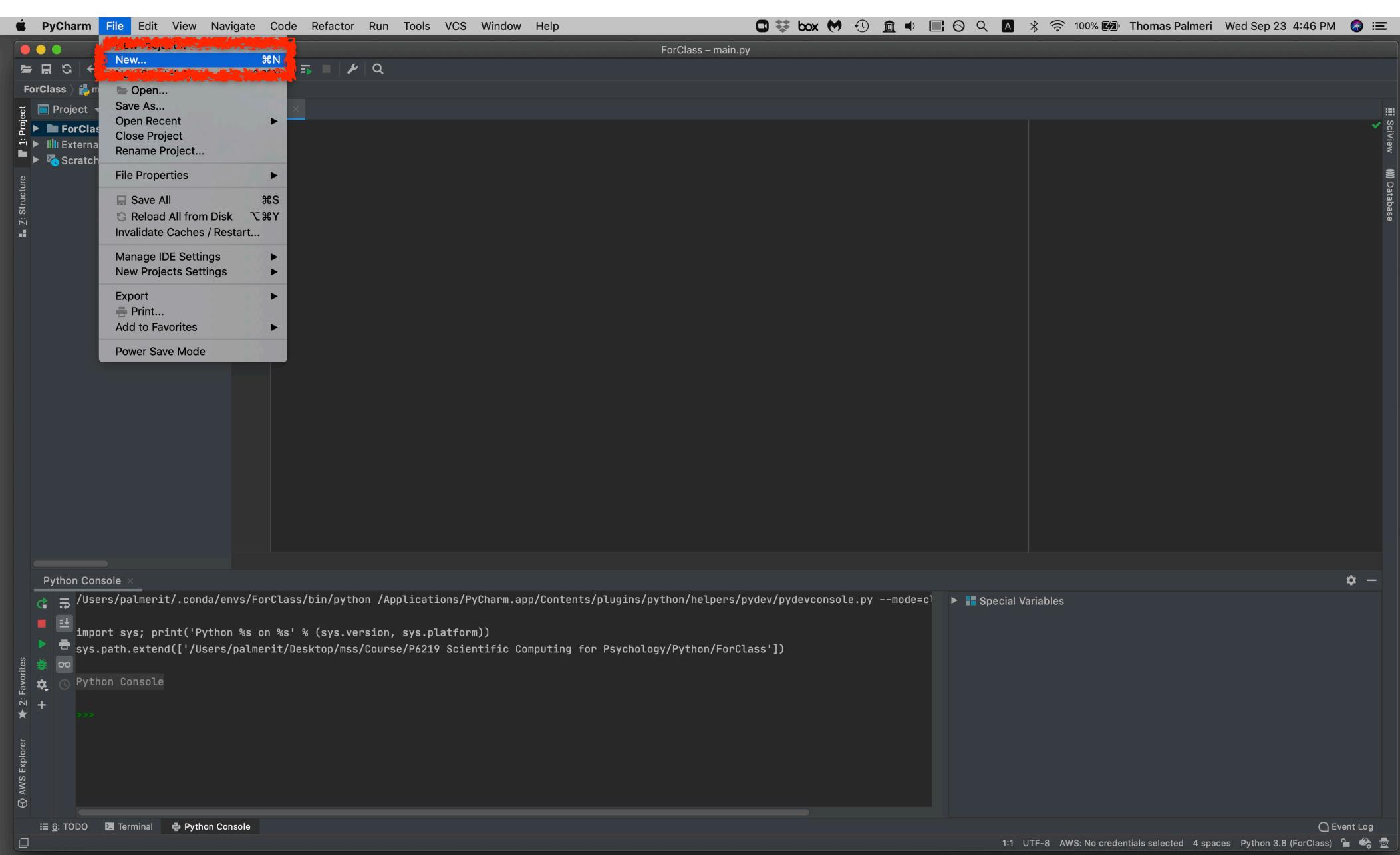
Version Control Problems TODO Terminal Python Console Python Packages

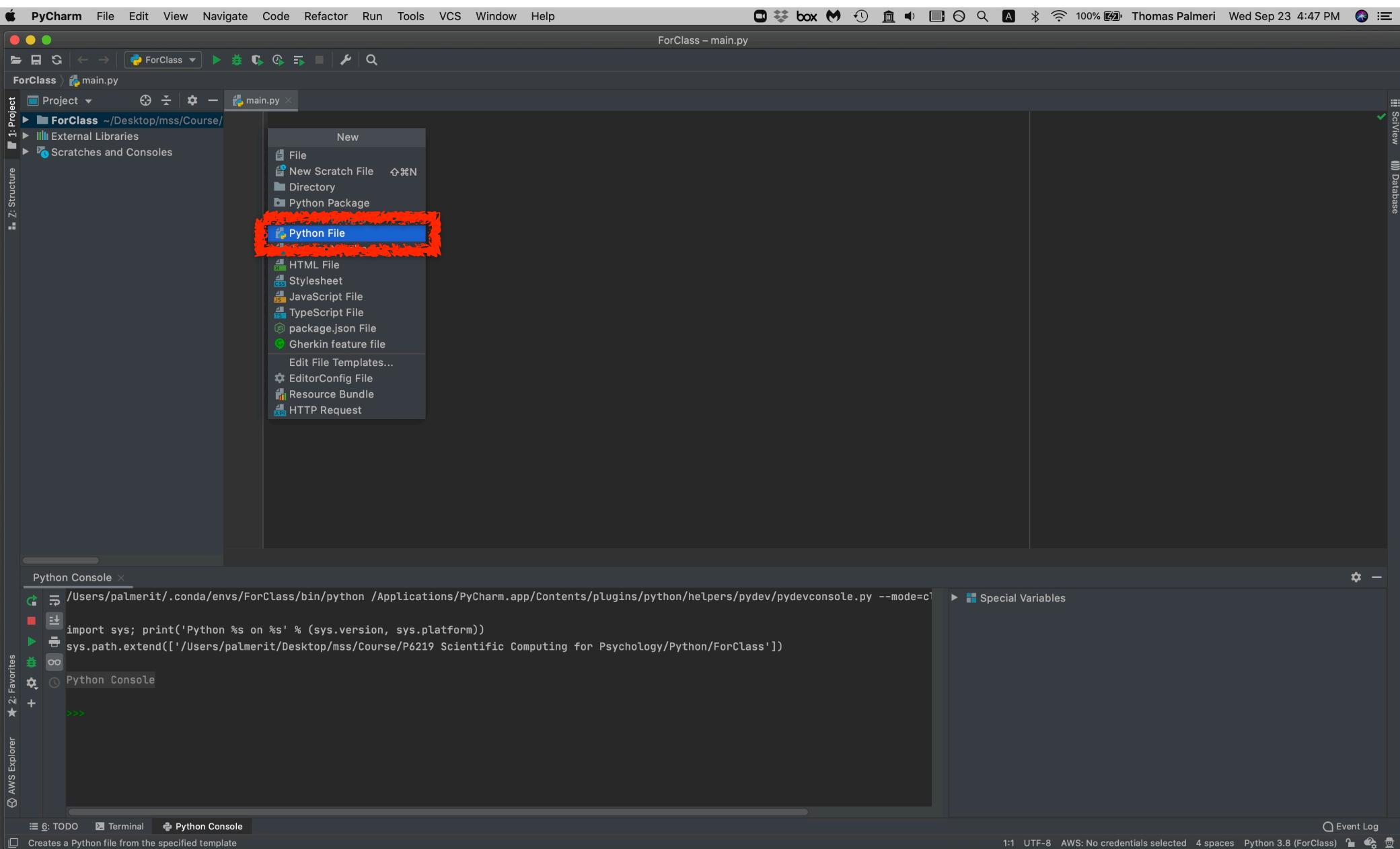
1:1 LF UTF-8 4 spaces Python 3.8 (PSY4219-Fall2022) AWS: No credentials selected

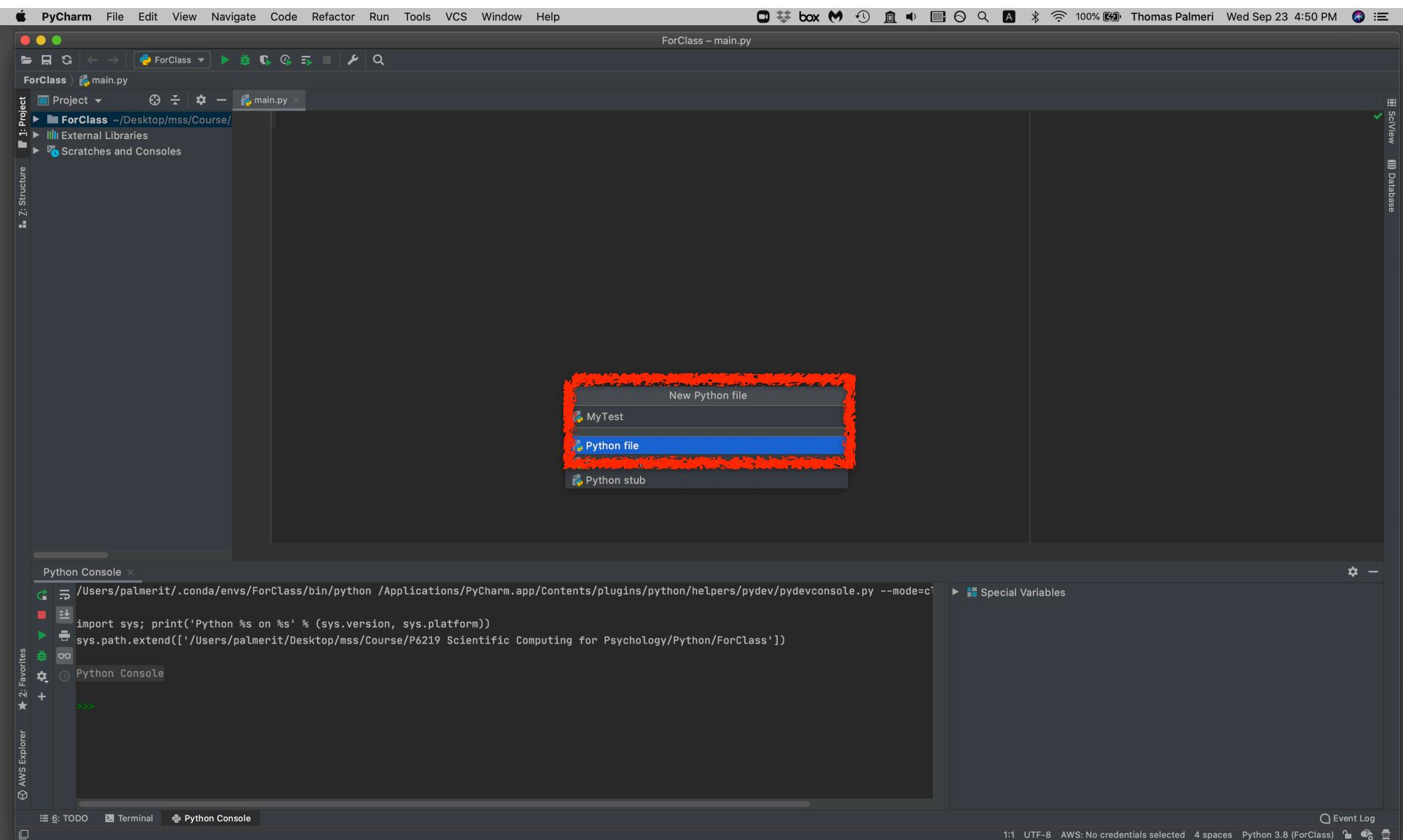


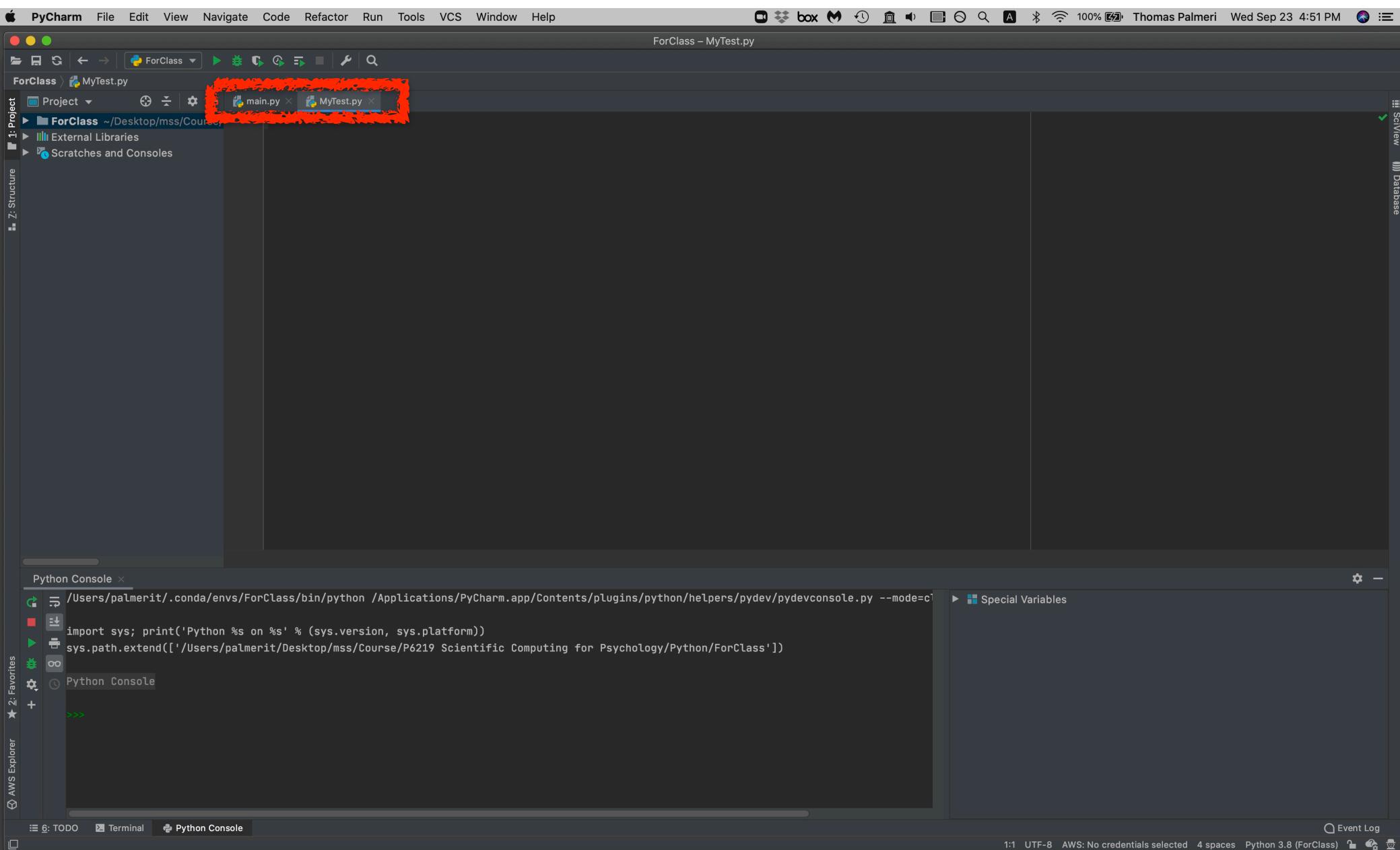
this is the folder that contains my Python files for this project

I created multiple folders, one for each week where we will use Python .py files

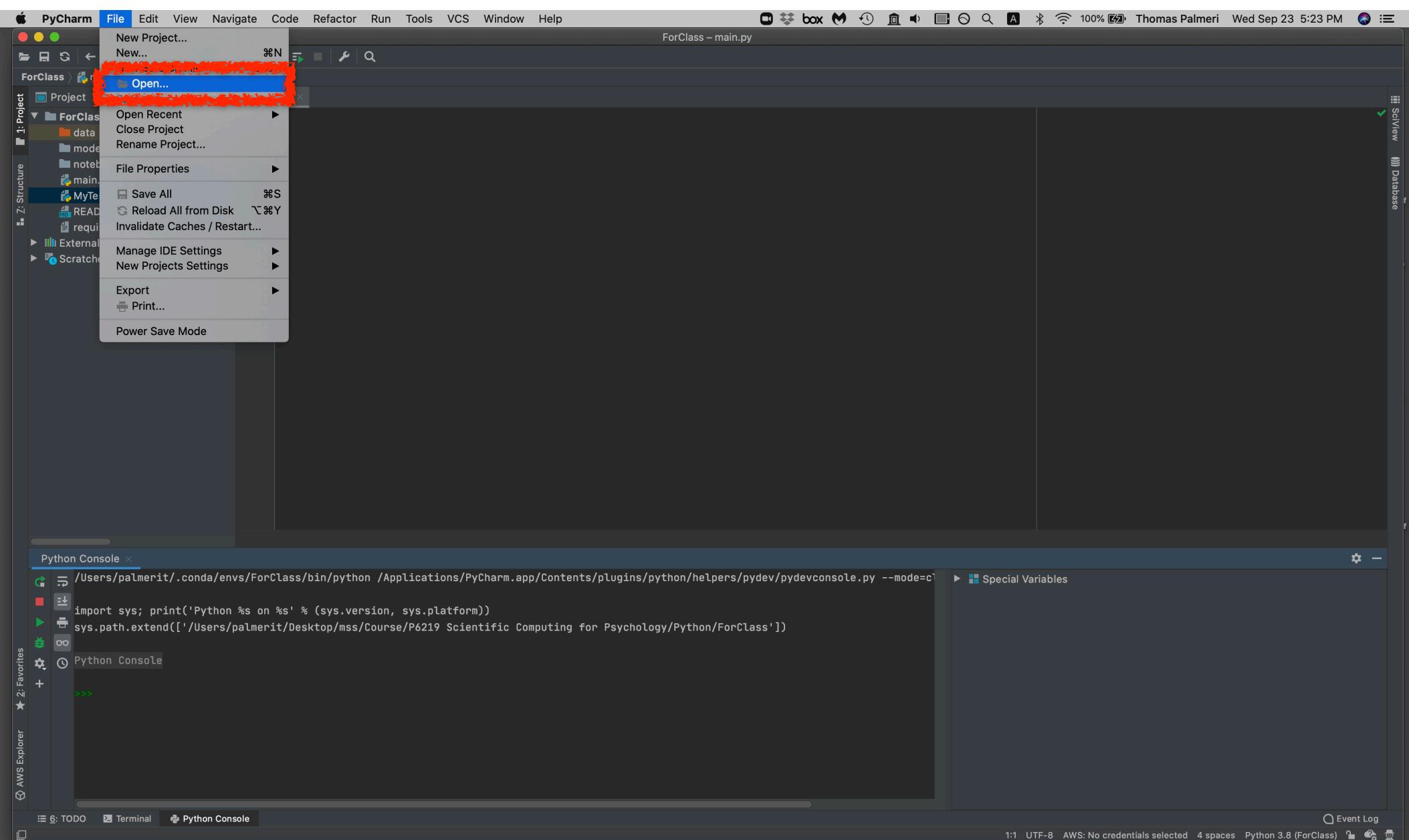








make sure you the file is in the project folder



The screenshot shows the PyCharm IDE interface. The top bar includes the PyCharm logo, menu items (File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help), and system status indicators (box, M, 100%, Thomas Palmeri, Wed Sep 23 4:53 PM). The main window displays the file 'ForClass - MyTest.py'. The code defines a function `psiphys` that computes a psychophysical function based on Weibull parameters. The Python Console tab at the bottom shows the execution of the script, including imports and the definition of `psiphys`. The bottom status bar indicates the file is 6: TODO, the encoding is UTF-8, and the Python version is 3.8 (ForClass).

```
ForClass - MyTest.py
1 > import numpy as np
2   import matplotlib.pyplot as plt
3
4 > #%%%
5
6 def psiphys(x, alpha=1., beta=2., gam=.5, lam=.1):
7     """Computes psychophysical function
8
9     Arguments:
10        x (numpy array): point at which to evaluate psi(x)
11        alpha (float): scale parameter of the Weibull
12        beta (float): shape parameter of the Weibull
13        gam (float): chance (floor on performance)
14        lam (float): lapse rate (ceil on performance)
15
16    Returns:
17        psi (numpy array): psychophysical function psi(x)
18
19    """
20    F = 1 - np.exp(-(x / alpha) * beta)
21    psi = gam + (1. - lam - gam) * F
22    return (psi)
23
24 x = np.arange(0, 5, .01)
25 psi1 = psiphys(x)
26 psi2 = psiphys(x, lam = .2, gam = .4)
```

Python Console ×

```
<ipython> /Users/palmerit/.conda/envs/ForClass/bin/python /Applications/PyCharm.app/Contents/plugins/python/helpers/pydev/pydevconsole.py --mode=cl
import sys; print('Python %s on %s' % (sys.version, sys.platform))
sys.path.append(['/Users/palmerit/Desktop/mss/Course/P6219 Scientific Computing for Psychology/Python/ForClass'])

Python Console
>>>
```

6: TODO Terminal Python Console Event Log

Python Console : place to try out code independent of the program you are writing

The screenshot shows the PyCharm IDE interface. The top navigation bar includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, and Help. The status bar at the bottom right shows the date as Wednesday, September 23, 2023, and the time as 4:53 PM.

The main window displays a Python file named `MyTest.py` under the project `ForClass`. The code defines a function `psiphys` that computes a psychophysical function. The Python Console tab is active, showing the execution of the script and its output. A red box highlights the Python Console area.

```
ForClass - MyTest.py
ForClass > MyTest.py
Project > ForClass ~/Desktop/mss/Course/
External Libraries
Scratches and Consoles
1: Structure
SciView
Database

ForClass - MyTest.py
1 > import numpy as np
2   import matplotlib.pyplot as plt
3
4 > #%%%
5
6 def psiphys(x, alpha=1., beta=2., gam=.5, lam=.1):
7     """Computes psychophysical function
8
9     Arguments:
10        x (numpy array): point at which to evaluate psi(x)
11        alpha (float): scale parameter of the Weibull
12        beta (float): shape parameter of the Weibull
13        gam (float): chance (floor on performance)
14        lam (float): lapse rate (ceil on performance)
15
16    Returns:
17        psi (numpy array): psychophysical function psi(x)
18    """
19    F = 1 - np.exp(-(x / alpha) * beta)
20    psi = gam + (1. - lam - gam) * F
21    return (psi)
22
23 x = np.arange(0, 5, .01)
24 psi1 = psiphys(x)
25 psi2 = psiphys(x, lam = .2, gam = .4)

Python Console >
/Users/palmerit/.conda/envs/ForClass/bin/python /Applications/PyCharm.app/Contents/plugins/python/helpers/pydev/pydevconsole.py --mode=client
import sys; print('Python %s on %s' % (sys.version, sys.platform))
sys.path.append(['/Users/palmerit/Desktop/mss/Course/P6219 Scientific Computing for Psychology/Python/ForClass'])
>>>
```

Python Console : place to try out code independent of the program you are writing

PyCharm File Edit View Navigate Code Refactor Run Tools VCS Window Help

ForClass – MyTest.py

ForClass > MyTest.py

Project

ForClass ~/Desktop/mss/Course/P

- data
- models
- notebooks
- main.py
- MyTest.py
- README.md
- requirements.txt

External Libraries

Scratches and Consoles

SciView

Database

```
1> import numpy as np
2> import matplotlib.pyplot as plt
3>
4> #%%
5>
6> def psiphys(x, alpha=1., beta=.2, gam=.5, lam=.1):
7>     """Computes psychophysical function
8>
9>     Arguments:
10>         x (numpy array): point at which to evaluate psi(x)
11>         alpha (float): scale parameter of the Weibull
12>         beta (float): shape parameter of the Weibull
13>         gam (float): chance (floor on performance)
14>         lam (float): lapse rate (ceil on performance)
15>
16>     Returns:
17>         psi (numpy array): psychophysical function psi(x)
18>     """
19>     F = 1 - np.exp(-(x / alpha) * beta)
20>     psi = gam + (1. - lam - gam) * F
21>     return psi
22>
23> x = np.arange(0, 5, .01)
24> psi1 = psiphys(x)
25> psi2 = psiphys(x, lam = .2, gam = .4)
26>
27> #%%
```

Terminal: Local +

```
The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
(ForClass) Tom-MacBook-Pro-2020:ForClass palmerit$ pwd
/Users/palmerit/Desktop/mss/Course/P6219 Scientific Computing for Psychology/Python/ForClass
(ForClass) Tom-MacBook-Pro-2020:ForClass palmerit$ ls
MyTest.py          README.md        data           main.py       models      notebooks    requirements.txt
(ForClass) Tom-MacBook-Pro-2020:ForClass palmerit$
```

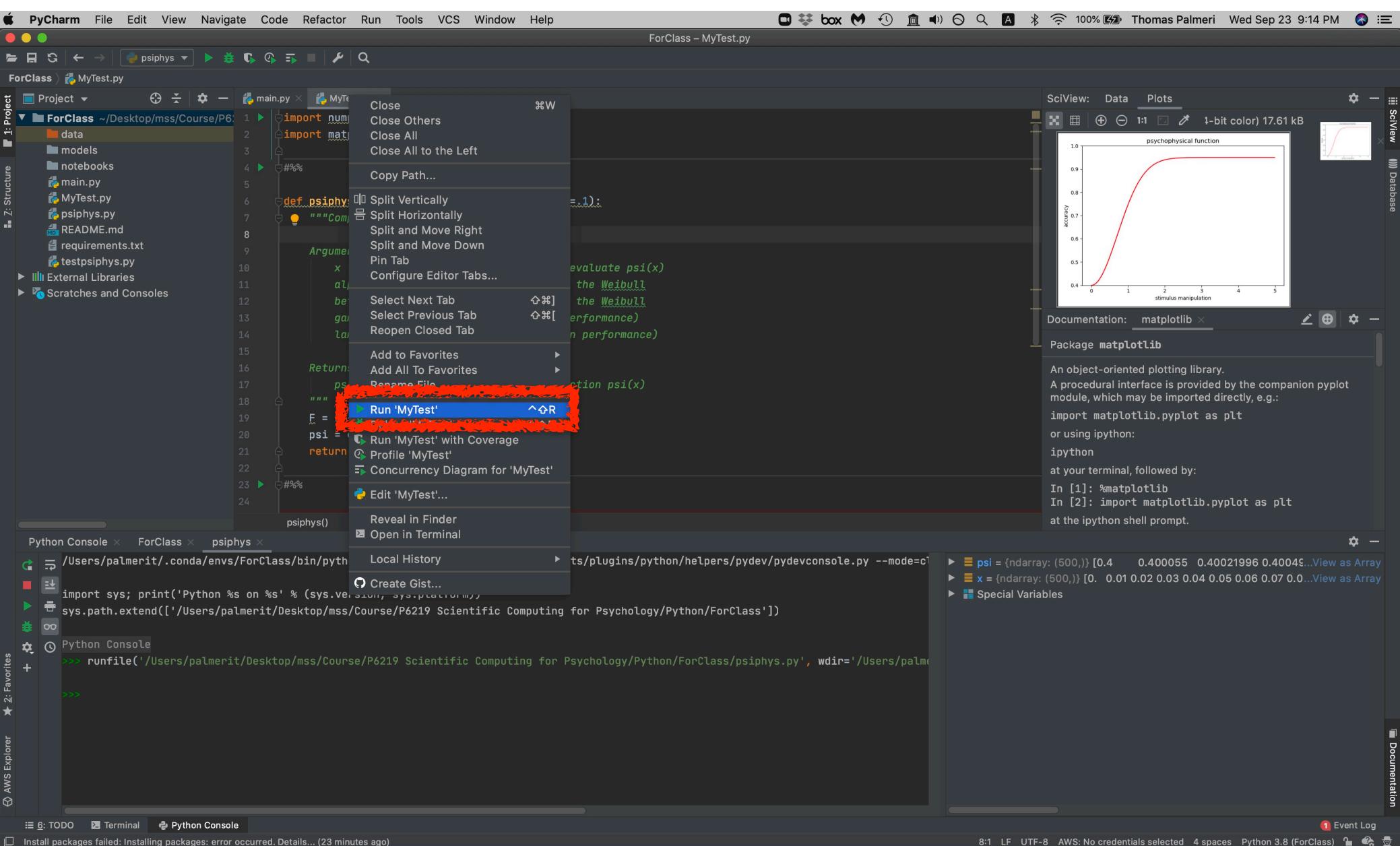
AWS

TOD Terminal Python Console Event Log

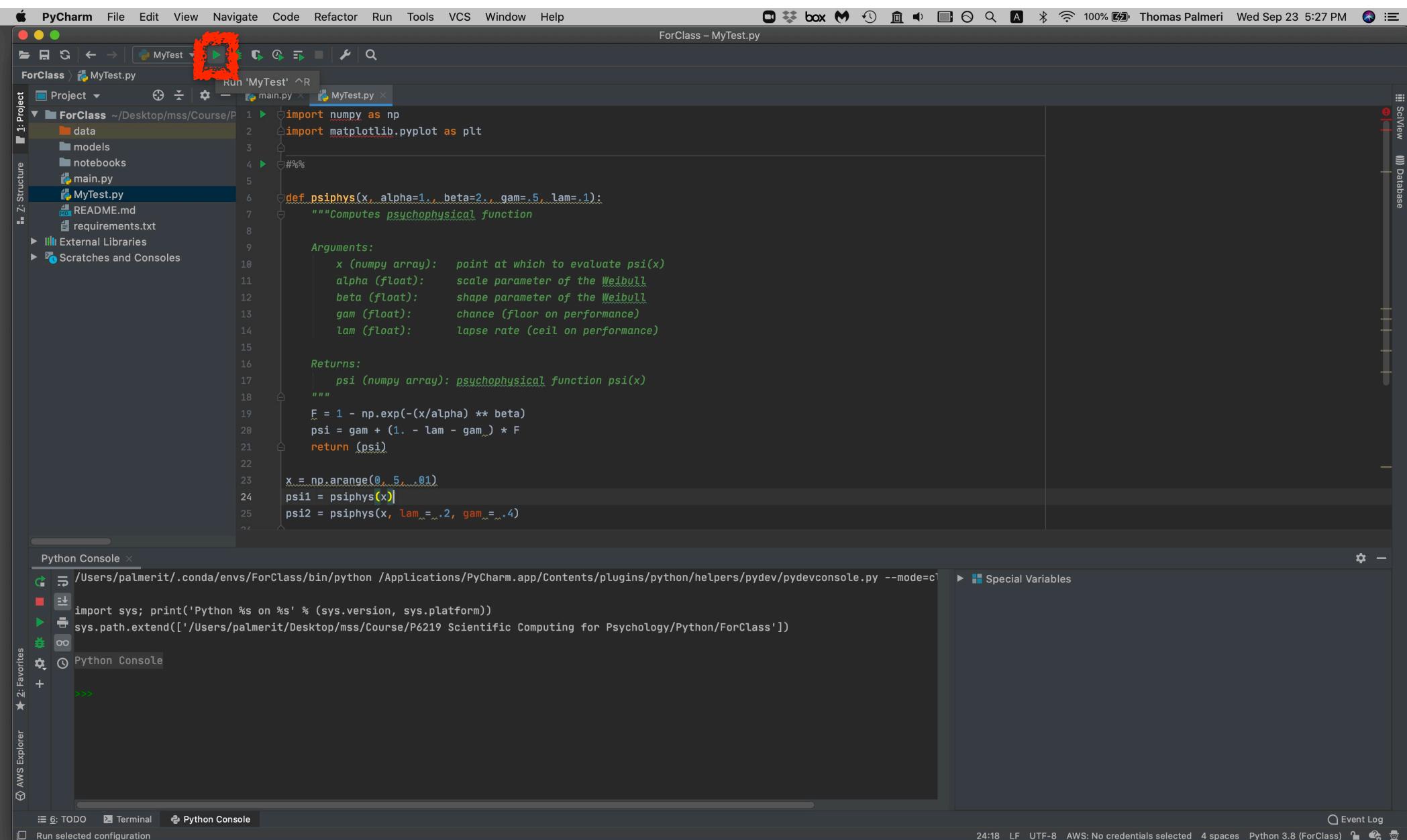
1:1 LF UTF-8 AWS: No credentials selected 4 spaces Python 3.8 (ForClass)

Terminal : work in Linux or Windows (enter in current folder)

right-click on the area containing the file (pulls up menu) and select Run 'MyTest'



green arrow to run MyTest.py



PyCharm File Edit View Navigate Code Refactor Run Tools VCS Window Help

ForClass – MyTest.py

Project : 1: Project ForClass ~/Desktop/mss/Course/P

ForClass data models notebooks main.py MyTest.py README.md requirements.txt External Libraries Scratches and Consoles

```
1 > import numpy as np
2 > import matplotlib.pyplot as plt
3
4 > #%%
5
6 def psiphys(x, alpha=1., beta=2., gam=.5, lam=.1):
7     """Computes psychophysical function
8
9     Arguments:
10        x (numpy array): point at which to evaluate psi(x)
11        alpha (float): scale parameter of the Weibull
12        beta (float): shape parameter of the Weibull
13        gam (float): chance (floor on performance)
14        lam (float): lapse rate (ceil on performance)
15
16    Returns:
17        psi (numpy array): psychophysical function psi(x)
18    """
19    F = 1 - np.exp(-(x/alpha) ** beta)
20    psi = gam + (1. - lam - gam) * F
21    return (psi)
22
23 x = np.arange(0, 5, .01)
24 psi1 = psiphys(x)
25 psi2 = psiphys(x, lam=.2, gam=.4)
```

Python Console x

```
/Users/palmerit/.conda/envs/ForClass/bin/python /Applications/PyCharm.app/Contents/plugins/python/helpers/pydev/pydevconsole.py --mode=c
```

Special Variables

import sys; print('Python %s on %s' % (sys.version, sys.platform))
sys.path.append(['/Users/palmerit/Desktop/mss/Course/P6219 Scientific Computing for Psychology/Python/ForClass'])

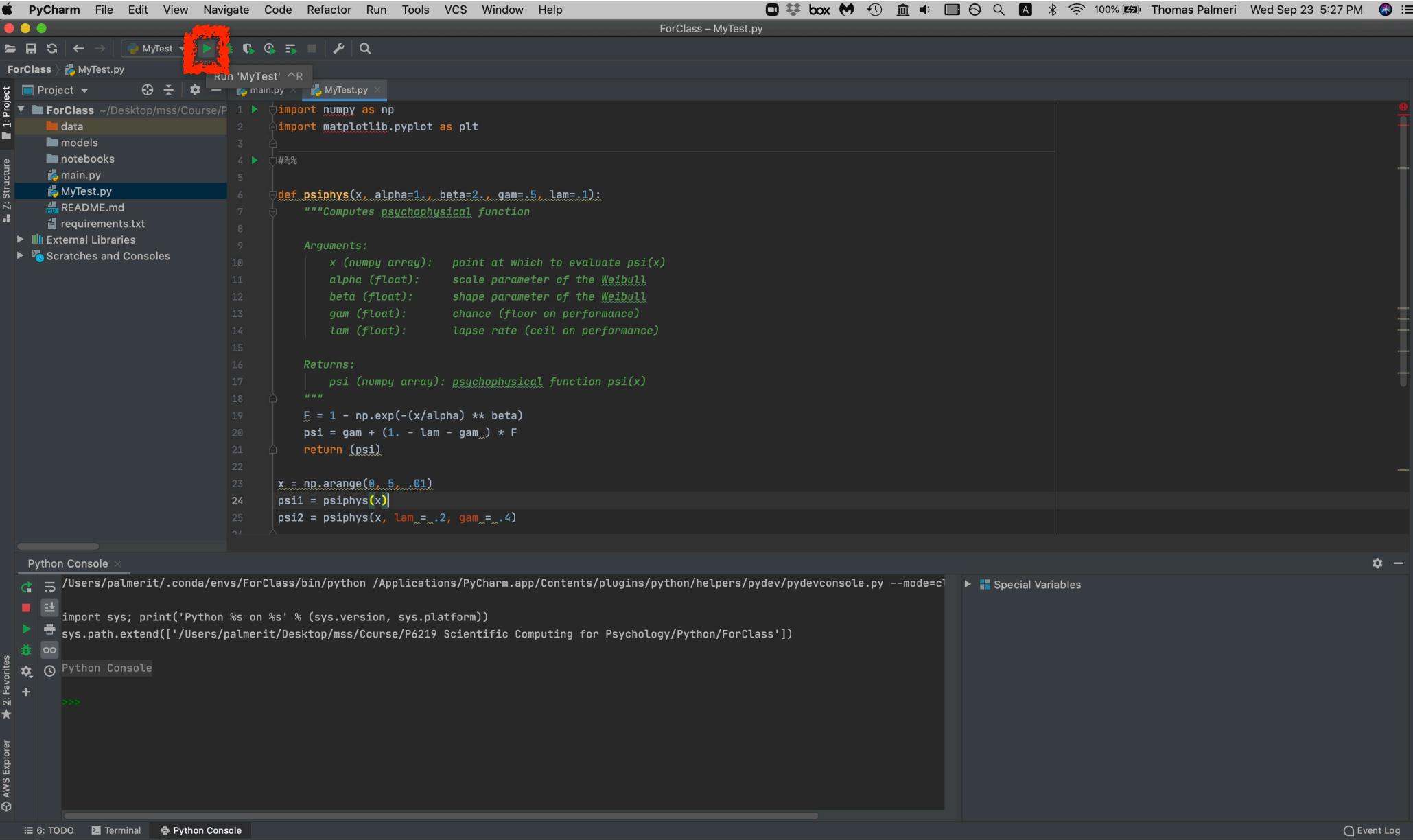
Python Console

>>>

Event Log

green arrow to run MyTest.py

the file that's visible is not necessarily the one that will run when you click the green arrow



PyCharm File Edit View Navigate Code Refactor Run Tools VCS Window Help

ForClass – MyTest.py

Project Structure

ForClass ~/Desktop/mss/Course/P

MyTest.py

main.py

MyTest.py

README.md

requirements.txt

External Libraries

Scratches and Consoles

```
1 > import numpy as np
2 > import matplotlib.pyplot as plt
3 >
4 > #%%
5 >
6 def psiphys(x, alpha=1., beta=2., gam=.5, lam=.1):
7     """Computes psychophysical function
8
9     Arguments:
10        x (numpy array): point at which to evaluate psi(x)
11        alpha (float): scale parameter of the Weibull
12        beta (float): shape parameter of the Weibull
13        gam (float): chance (floor on performance)
14        lam (float): lapse rate (ceil on performance)
15
16    Returns:
17        psi (numpy array): psychophysical function psi(x)
18    """
19    F = 1 - np.exp(-(x/alpha) ** beta)
20    psi = gam + (1. - lam - gam) * F
21    return (psi)
22
23 x = np.arange(0, 5, .01)
24 psi1 = psiphys(x)
25 psi2 = psiphys(x, lam=.2, gam=.4)
```

Python Console

```
/Users/palmerit/.conda/envs/ForClass/bin/python /Applications/PyCharm.app/Contents/plugins/python/helpers/pydev/pydevconsole.py --mode=c
```

Special Variables

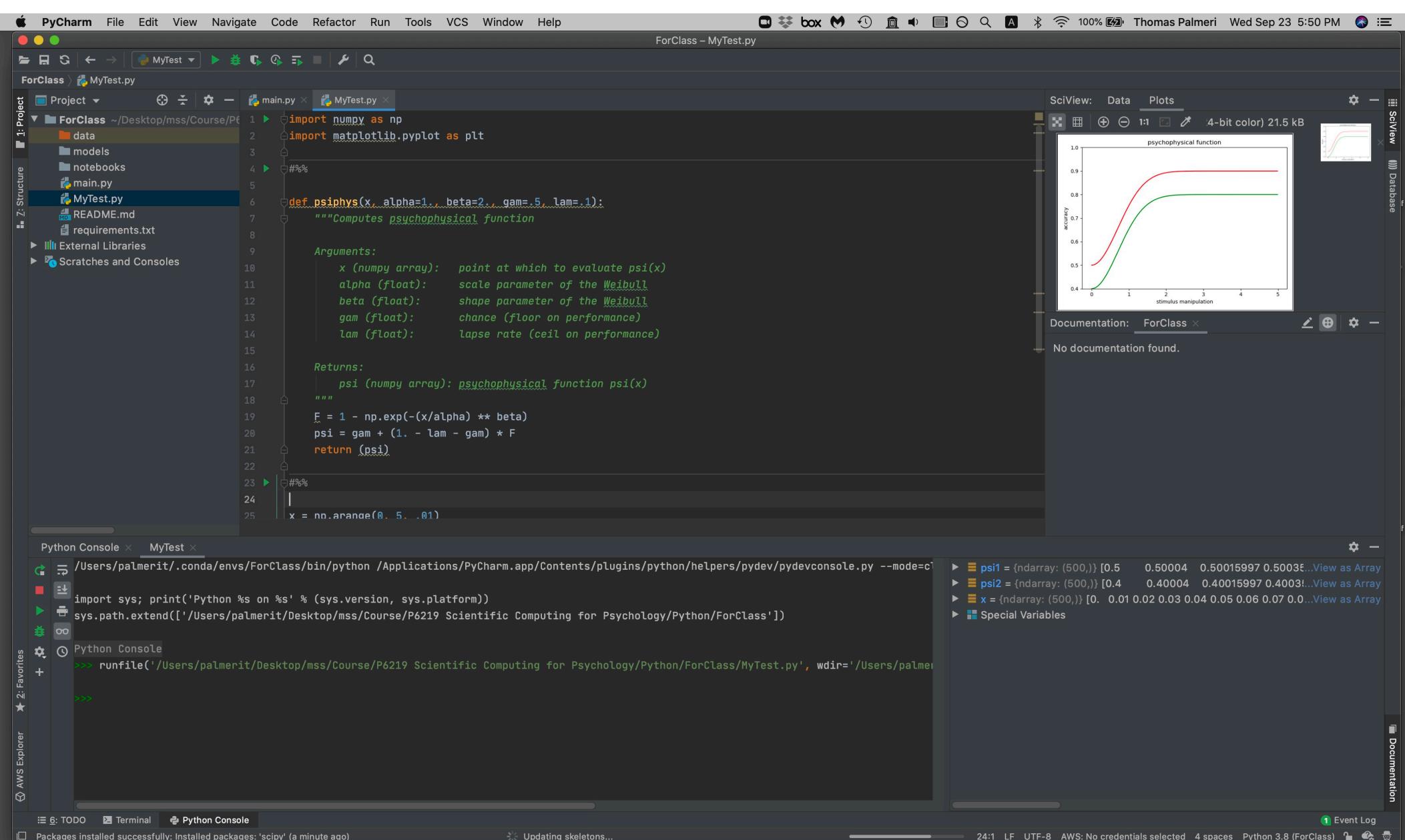
import sys; print('Python %s on %s' % (sys.version, sys.platform))

sys.path.append(['/Users/palmerit/Desktop/mss/Course/P6219 Scientific Computing for Psychology/Python/ForClass'])

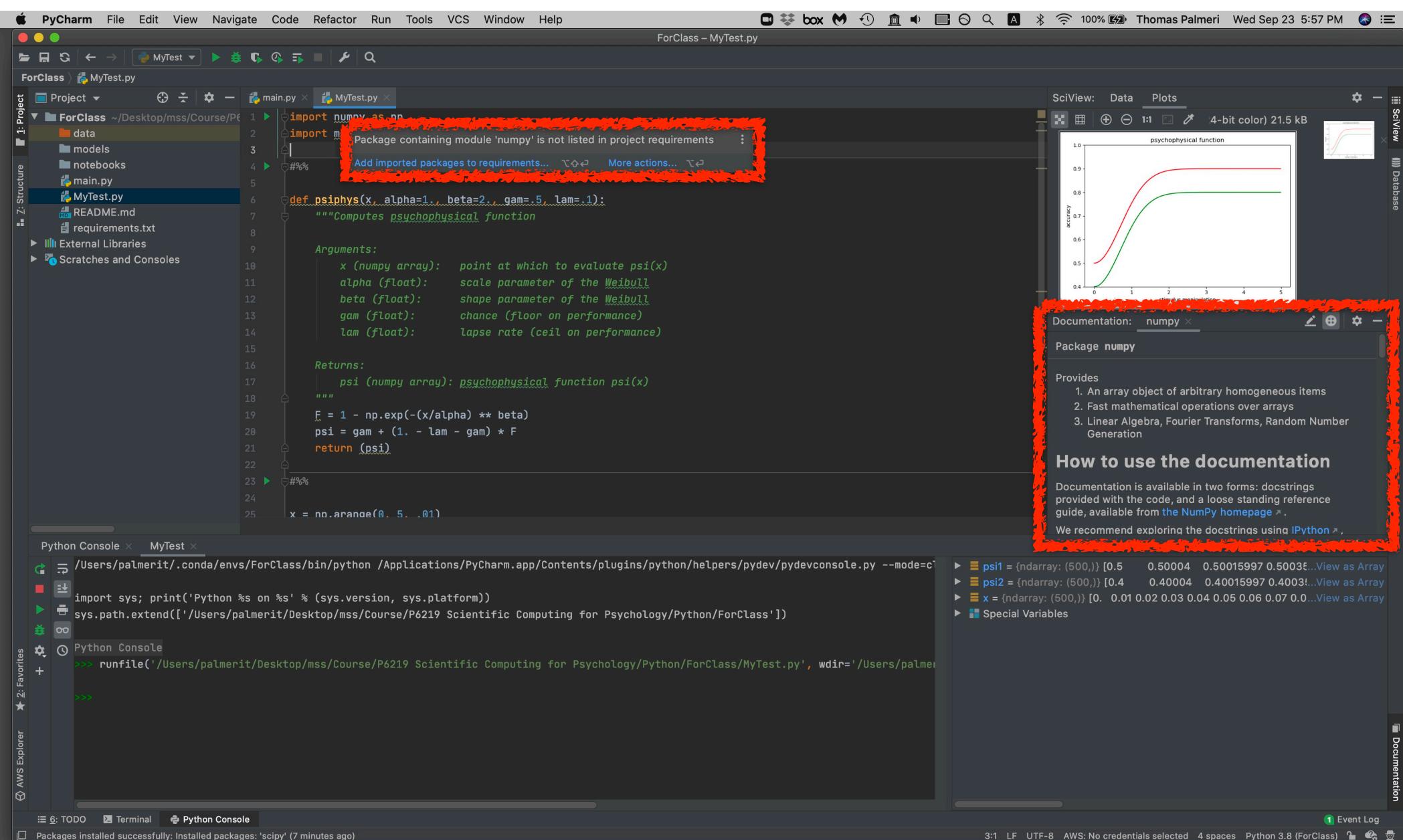
Python Console

>>>

Event Log



hovering mouse reveals errors, warning, and opens documentation



including documentation we created ourselves (for our functions)

The screenshot shows the PyCharm IDE interface with the following components:

- Project:** Shows the file structure of the "ForClass" project.
- Code Editor:** Displays the file `MyTest.py` containing Python code. A red box highlights a PEP 8 warning: "PEP 8: E302 expected 2 blank lines, found 1". Below the code, a tooltip provides reformatting options.
- SciView:** A plot titled "psychophysical function" showing accuracy versus a variable (likely x). The plot has two curves: a red one starting at ~0.45 and a green one starting at ~0.4.
- Documentation:** A panel on the right showing the docstring for the `psiphys` function, its parameters, and its purpose.
- Python Console:** Shows the output of running the script, including imports and the execution of `runfile`.
- Bottom Status Bar:** Shows the file encoding (3:1 LF), character set (UTF-8), AWS status, and Python version (3.8).

some of the warnings are PEP style guidelines (nothing more)

(in scientific mode)

#%% identifies code cells (not part of Python, but understood by many IDEs)

The screenshot shows the PyCharm IDE interface with the following components:

- Top Bar:** PyCharm, File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help.
- Toolbar:** Standard file operations (New, Open, Save, Find, etc.).
- Project View:** Shows the project structure under "ForClass".
- Code Editor:** The file "MyTest.py" is open, containing the following code:

```
import numpy as np
import matplotlib.pyplot as plt
#%%
def psiphys(x, alpha=1., beta=2., gam=.5, lam=.1):
    """Computes psychophysical function

    Arguments:
        x (numpy array): point at which to evaluate psi(x)
        alpha (float): scale parameter of the Weibull
        beta (float): shape parameter of the Weibull
        gam (float): chance (floor on performance)
        lam (float): lapse rate (ceil on performance)

    Returns:
        psi (numpy array): psychophysical function psi(x)
    """
    F = 1 - np.exp(-(x/alpha) ** beta)
    psi = gam + (1. - lam - gam) * F
    return (psi)
#%%
x = np.arange(0., 5., .01)
```

- SciView:** A plot titled "psychophysical function" showing accuracy versus stimulus manipulation. The red curve starts at approximately (0, 0.45) and rises to about 0.9 at x=5. The green curve starts at (0, 0.4) and rises more gradually to about 0.8 at x=5.
- Documentation:** A detailed docstring for the `psiphys` function, including parameters and returns.
- Python Console:** Shows the command runfile and its output.
- Bottom Status Bar:** Event Log, TODO, Terminal, Packages installed successfully: Installed packages: 'scipy' (15 minutes ago), 3:1 LF UTF-8 AWS: No credentials selected 4 spaces Python 3.8 (ForClass).

press green arrow to run the code cell (and only that code cell)

ForClass – MyTest.py

```
import numpy as np
import matplotlib.pyplot as plt

def psiphys(x, alpha=1., beta=2., gam=.5, lam=.1):
    """Computes psychophysical function

    Arguments:
        x (numpy array): point at which to evaluate psi(x)
        alpha (float): scale parameter of the Weibull
        beta (float): shape parameter of the Weibull
        gam (float): chance (floor on performance)
        lam (float): lapse rate (ceil on performance)

    Returns:
        psi (numpy array): psychophysical function psi(x)
    """
    F = 1 - np.exp(-(x/alpha)**beta)
    psi = gam + (1. - lam - gam) * F
    return (psi)

x = np.arange(0., 5., .01)
```

SciView: Data Plots 4-bit color 21.5 kB

psychophysical function

accuracy

stimulus manipulation

Documentation: psiphys(x, alpha=1., beta=2., gam=.5, l...

MyTest

```
def psiphys(x: Any,
            alpha: float = 1.,
            beta: float = 2.,
            gam: float = .5,
            lam: float = .1) -> Any
```

Computes psychophysical function

Params: x – point at which to evaluate psi(x)
alpha – scale parameter of the Weibull
beta – shape parameter of the Weibull
gam – chance (floor on performance)
lam – lapse rate (ceil on performance)

Returns: psychophysical function psi(x)

Python Console x MyTest x

```
/Users/palmerit/.conda/envs/ForClass/bin/python /Applications/PyCharm.app/Contents/plugins/python/helpers/pydev/pydevconsole.py --mode=client
import sys; print('Python %s on %s' % (sys.version, sys.platform))
sys.path.append(['/Users/palmerit/Desktop/mss/Course/P6219 Scientific Computing for Psychology/Python/ForClass'])

```

Python Console

```
>>> runfile('/Users/palmerit/Desktop/mss/Course/P6219 Scientific Computing for Psychology/Python/ForClass/MyTest.py', wdir='/Users/palmerit/Desktop/mss/Course/P6219 Scientific Computing for Psychology/Python/ForClass')
```

psi1 = {ndarray: (500,)} [0.5 0.50004 0.50015997 0.5003...View as Array

psi2 = {ndarray: (500,)} [0.4 0.40004 0.40015997 0.4003...View as Array

x = {ndarray: (500,)} [0. 0.01 0.02 0.03 0.04 0.05 0.06 0.07 0.0...View as Array

Special Variables

AWS Explorer

Event Log

Packages installed successfully: Installed packages: 'scipy' (15 minutes ago)

PyCharm File Edit View Navigate Code Refactor Run Tools VCS Window Help

ForClass – MyTest.py

ForClass > MyTest.py

Project

1: Project ForClass ~/Desktop/mss/Course/P6

2: Structure

3: SciView: Data Plots

4-bit color) 21.5 kB

SciView Database

ForClass.py

```

x (numpy array): point at which to evaluate psi(x)
alpha (float): scale parameter of the Weibull
beta (float): shape parameter of the Weibull
gam (float): chance (floor or performance)
lam (float): lapse rate (ceil on performance)

>Returns:
psi (numpy array): psychophysical function psi(x)
"""
F = 1 - np.exp(-(x/alpha) ** beta)
psi = gam + (1. - lam - gam) * F
return (psi)

#%%
x = np.arange(0, 5, .01)
psi1 = psiphys(x)
psi2 = psiphys(x, lam=.2, gam=.4)

#%%
plt.plot(x, psi1, 'r-', x, psi2, 'g-')
plt.xlabel("stimulus manipulation")
plt.ylabel("accuracy")

```

Documentation: psiphys(x, alpha=1., beta=2., gam=.5, l...

MyTest

```

def psiphys(x: Any,
            alpha: float = 1.,
            beta: float = 2.,
            gam: float = .5,
            lam: float = .1) -> Any

Computes psychophysical function

Params: x – point at which to evaluate psi(x)
        alpha – scale parameter of the Weibull
        beta – shape parameter of the Weibull
        gam – chance (floor on performance)
        lam – lapse rate (ceil on performance)

Returns: psychophysical function psi(x)

```

Python Console x MyTest x

```

...     Returns:
...         psi (numpy array): psychophysical function psi(x)
...
...     """
...     F = 1 - np.exp(-(x/alpha) ** beta)
...     psi = gam + (1. - lam - gam) * F
...     return (psi)
...
...
>>> x = np.arange(0, 5, .01)
... psi1 = psiphys(x)
... psi2 = psiphys(x, lam=.2, gam=.4)
...
...
>>>

```

psi1 = {ndarray: (500,)} [0.5 0.50004 0.50015997 0.5003...View as Array

psi2 = {ndarray: (500,)} [0.4 0.40004 0.40015997 0.4003...View as Array

x = {ndarray: (500,)} [0. 0.01 0.02 0.03 0.04 0.05 0.06 0.07 0.0...View as Array

Special Variables

AWS Explorer

Event Log

Todo Terminal Python Console

Packages installed successfully: Installed packages: 'scipy' (18 minutes ago)

29:1 LF UTF-8 AWS: No credentials selected 4 spaces Python 3.8 (ForClass)

see the code from that cell executed in the MyTest Console

PyCharm File Edit View Navigate Code Refactor Run Tools VCS Window Help

ForClass – MyTest.py

Project: ForClass ~/Desktop/mss/Course/600.641/ForClass

Structure: main.py MyTest.py

```

18 F = 1 - np.exp(-(x/alpha) ** beta)
19 psi = gam + (1. - lam - gam) * F
20 return (psi)
21
22 #%%
23 x = np.arange(0, 5, .01)
24 psi1 = psiphys(x)
25 psi2 = psiphys(x, lam=.2, gam=.4)
26
27 #%%
28
29 plt.plot(x, psi1, 'r-', x, psi2, 'g-')
30 plt.xlabel("stimulus manipulation")
31 plt.ylabel("accuracy")
32 plt.title("psychophysical function")
33 plt.ylim(.4, 1)
34 plt.show()
35
36
37

```

SciView: Data Plots 4-bit color 21.5 kB

psychophysical function

Documentation: psiphys(x, alpha=1., beta=2., gam=.5, lam=.1) → Any

MyTest

```

def psiphys(x: Any,
            alpha: float = 1.,
            beta: float = 2.,
            gam: float = .5,
            lam: float = .1) -> Any

```

Computes psychophysical function

Params: x – point at which to evaluate psi(x)
alpha – scale parameter of the Weibull
beta – shape parameter of the Weibull
gam – chance (floor on performance)
lam – lapse rate (ceil on performance)

Returns: psychophysical function psi(x)

Python Console: MyTest

```

In [1]: ...
        F = 1 - np.exp(-(x/alpha) ** beta)
        psi = gam + (1. - lam - gam) * F
        return (psi)

In [2]: ...
        x = np.arange(0, 5, .01)
        psi1 = psiphys(x)
        psi2 = psiphys(x, lam=.2, gam=.4)

In [3]: ...
        x[0:10]
        array([0. , 0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09])

```

AWS Explorer: 2: Favorites

Event Log: 1 Event

Packages installed successfully: Installed packages: 'scipy' (22 minutes ago)

1:1 LF UTF-8 AWS: No credentials selected 4 spaces Python 3.8 (ForClass)

can view, manipulate, set variables from program in the MyTest Console

PyCharm File Edit View Navigate Code Refactor Run Tools VCS Window Help

ForClass – MyTest.py

ForClass > MyTest.py

Project 1: Project ~Desktop/mss/Course/P6219

ForClass data models notebooks main.py MyTest.py README.md requirements.txt External Libraries Scratches and Consoles

SciView: Data Plots 1:1 4-bit color 21.5 kB

psychophysical function

accuracy stimulus manipulation

Documentation: psiphys(x, alpha=1., beta=2., gam=.5, lam=.1) → Any

MyTest

```
def psiphys(x: Any,
            alpha: float = 1.,
            beta: float = 2.,
            gam: float = .5,
            lam: float = .1) -> Any
```

Computes psychophysical function

Params: x – point at which to evaluate psi(x)
alpha – scale parameter of the Weibull
beta – shape parameter of the Weibull
gam – chance (floor on performance)
lam – lapse rate (ceil on performance)

Returns: psychophysical function psi(x)

Python Console x MyTest x

```
Python %s on %s' % (sys.version, sys.platform)
sys.path.extend(['/Users/palmerit/Desktop/mss/Course/P6219 Scientific Computing for Psychology/Python/ForClass'])
PyDev console: starting.
Python 3.8.5 (default, Sep 4 2020, 02:22:02)
+ x[0:10]
>>> x[0:10]
Traceback (most recent call last):
  File "<input>", line 1, in <module>
    NameError: name 'x' is not defined
>>>
```

Special Variables

AWS Explorer

Event Log

Packages installed successfully: Installed packages: 'scipy' (23 minutes ago)

1:1 LF UTF-8 AWS: No credentials selected 4 spaces Python 3.8 (ForClass)

```
F = 1 - np.exp(-(x/alpha) ** beta)
psi = gam + (1. - lam - gam) * F
return (psi)

x = np.arange(0, 5, .01)
psi1 = psiphys(x)
psi2 = psiphys(x, lam=.2, gam=.4)

plt.plot(x, psi1, 'r-', x, psi2, 'g-')
plt.xlabel("stimulus manipulation")
plt.ylabel("accuracy")
plt.title("psychophysical function")
plt.ylim(.4, 1)
plt.show()
```

the Python Console is independent (doesn't see variables running in MyTest)

PyCharm File Edit View Navigate Code Refactor Run Tools VCS Window Help

ForClass – MyTest.py

Project: ForClass ~/Desktop/mss/Course/ForClass

Structure: ForClass, data, models, notebooks, main.py, MyTest.py, README.md, requirements.txt, External Libraries, Scratches and Consoles

SciView: Data Plots 4-bit color 21.5 kB

psychophysical function

Documentation: psiphys(x, alpha=1., beta=2., gam=.5, lam=.1) → Any

MyTest

```
def psiphys(x: Any,
            alpha: float = 1.,
            beta: float = 2.,
            gam: float = .5,
            lam: float = .1) -> Any
```

Computes psychophysical function

Params: x – point at which to evaluate psi(x)
alpha – scale parameter of the Weibull
beta – shape parameter of the Weibull
gam – chance (floor on performance)
lam – lapse rate (ceil on performance)

Returns: psychophysical function psi(x)

Python Console x MyTest x

```
... F = 1 - np.exp(-(x/alpha) ** beta)
... psi = gam + (1. - lam - gam) * F
... return (psi)

>>> x = np.arange(0, 5, .01)
... psi1 = psiphys(x)
... psi2 = psiphys(x, lam=.2, gam=.4)
...
>>> x[0:10]
array([0. , 0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09])

>>>
```

AWS Explorer

Todo Terminal Python Console Event Log

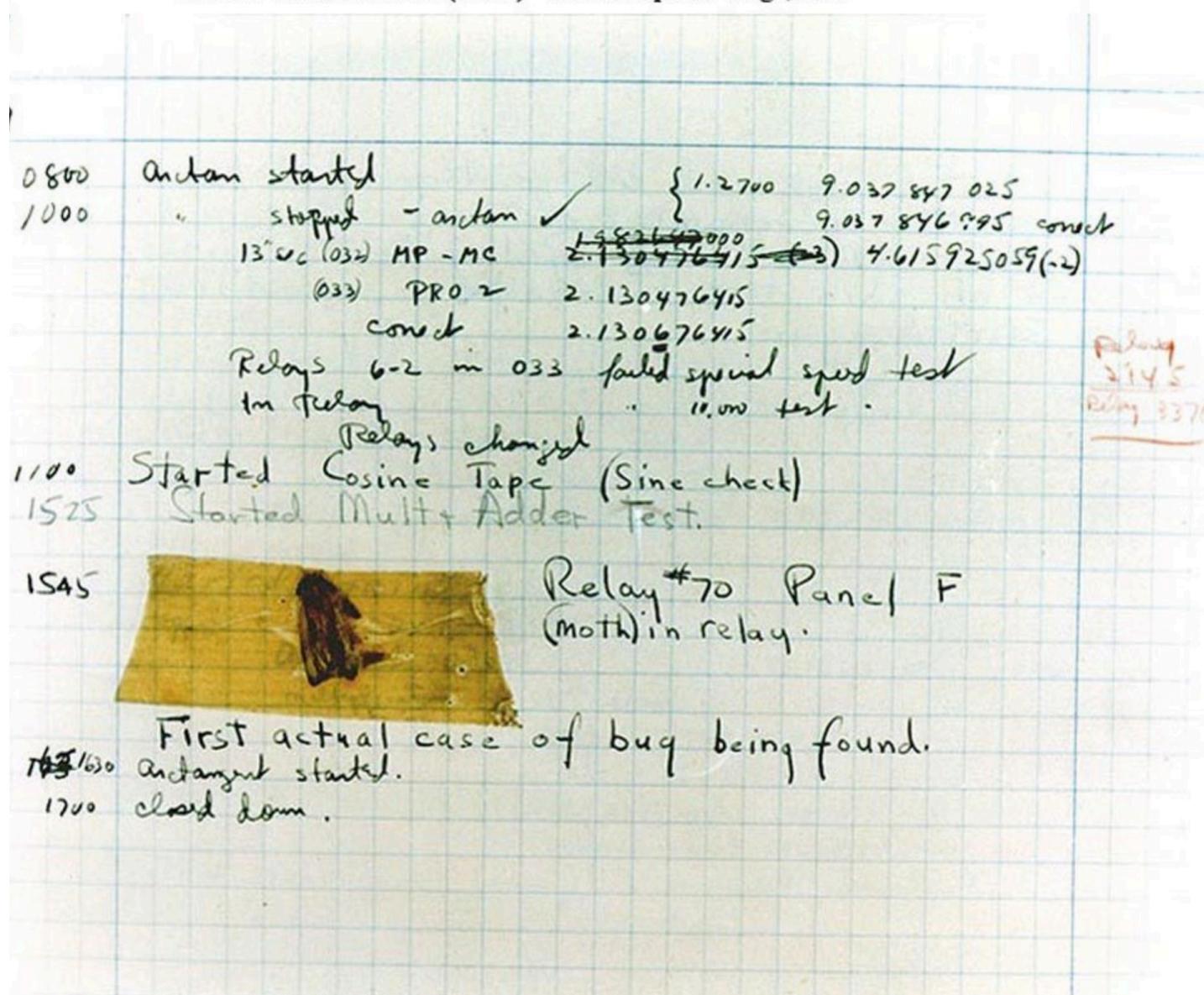
Packages installed successfully: Installed packages: 'scipy' (22 minutes ago)

1:1 LF UTF-8 AWS: No credentials selected 4 spaces Python 3.8 (ForClass)

also notice that variables (and their values and types) are shown here

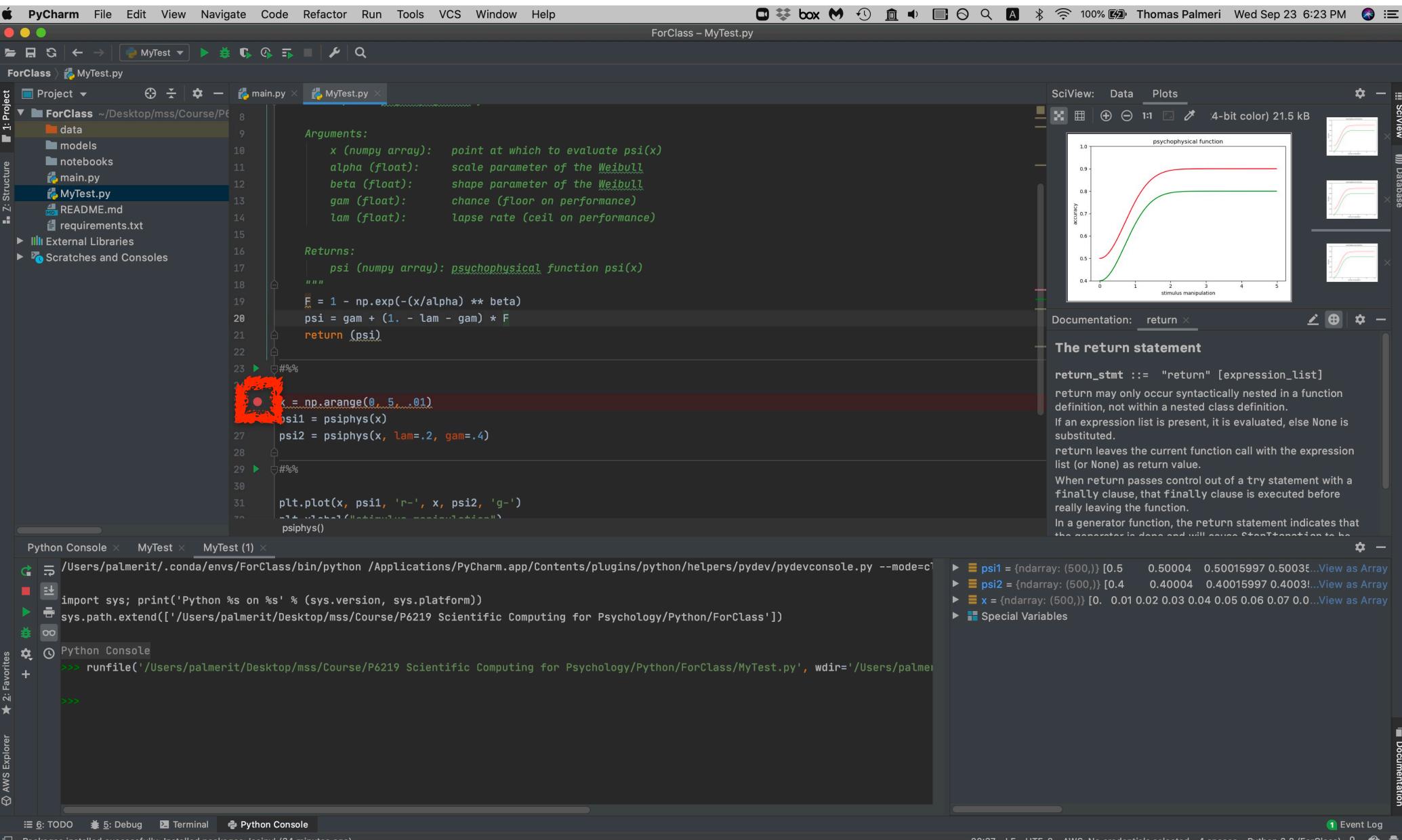
debugging in PyCharm

Photo # NH 96566-KN (Color) First Computer "Bug", 1947



Edison actually coined the term bug (in a system) before computers were around

click in the "gutter" (in same column as green arrows) to set a **breakpoint** (can have many)



The screenshot shows the PyCharm IDE interface with the following details:

- File Menu:** PyCharm, File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help.
- Toolbar:** Includes icons for file operations like Open, Save, and Run.
- Project View:** Shows the project structure under "ForClass".
- Code Editor:** The file "MyTest.py" is open. A red circle highlights the gutter area where a breakpoint is being set. The code defines a function "psiphys" with arguments x, alpha, beta, gamma, and lam, and returns a psychophysical function psi(x). It also contains a plot command.
- SciView:** A panel titled "SciView" displays a graph of "psychophysical function" accuracy versus stimulus manipulation. The graph shows two curves: a red curve starting at (0,0) and a green curve starting at approximately (0,0.4).
- Documentation:** A sidebar provides documentation for the "return" statement, including examples and syntax rules.
- Python Console:** The console output shows the execution of "runfile" command, which runs the script and displays the SciView plot.
- Bottom Status Bar:** Shows the current time (20:37), encoding (LF), file encoding (UTF-8), AWS status, number of spaces (4), Python version (3.8), and event log.

click on that to debug (it's supposed to look like a bug, but with my old eyes looks like a turtle)

PyCharm File Edit View Navigate Code Refactor Run Tools VCS Window Help

ForClass – MyTest.py

ForClass MyTest.py Debug 'MyTest' ^D

Project 1: Project ForClass ~Desktop/mss/Course/P6219 Scientific Computing for Psychology/Python/ForClass

Structure 7: Structure

SciView: Data Plots 4-bit color 21.5 kB SciView Database

Arguments:

```
x (numpy array): point at which to evaluate psi(x)
alpha (float): scale parameter of the Weibull
beta (float): shape parameter of the Weibull
gam (float): chance (floor on performance)
lam (float): lapse rate (ceil on performance)
```

Returns:

```
psi (numpy array): psychophysical function psi(x)
"""
F = 1 - np.exp(-(x/alpha) ** beta)
psi = gam + (1. - lam - gam) * F
return (psi)
```

Documentation: return

The return statement

```
return_stmt ::= "return" [expression_list]
return may only occur syntactically nested in a function definition, not within a nested class definition.
If an expression list is present, it is evaluated, else None is substituted.
return leaves the current function call with the expression list (or None) as return value.
When return passes control out of a try statement with a finally clause, that finally clause is executed before really leaving the function.
In a generator function, the return statement indicates that the generator is done and will cause StopIteration to be
```

PyCharm Console x MyTest x MyTest (1) x

```
/Users/palmerit/.conda/envs/ForClass/bin/python /Applications/PyCharm.app/Contents/plugins/python/helpers/pydev/pydevconsole.py --mode=cl
import sys; print('Python %s on %s' % (sys.version, sys.platform))
sys.path.append(['/Users/palmerit/Desktop/mss/Course/P6219 Scientific Computing for Psychology/Python/ForClass'])

```

Python Console

```
>>> runfile('/Users/palmerit/Desktop/mss/Course/P6219 Scientific Computing for Psychology/Python/ForClass/MyTest.py', wdir='/Users/palmerit/Desktop/mss/Course/P6219 Scientific Computing for Psychology/Python/ForClass')
```

Variables

- psi1 = {ndarray: (500,)} [0.5 0.50004 0.50015997 0.5003...View as Array
- psi2 = {ndarray: (500,)} [0.4 0.40004 0.40015997 0.4003!...View as Array
- x = {ndarray: (500,)} [0. 0.01 0.02 0.03 0.04 0.05 0.06 0.07 0.0...View as Array
- Special Variables

AWS Explorer

Event Log

Debug selected configuration

20:37 LF UTF-8 AWS: No credentials selected 4 spaces Python 3.8 (ForClass)

code suspended at that line, but has not executed that line yet

Screenshot of PyCharm IDE showing a Python script named MyTest.py. The code defines a function psi and uses it to plot two psychophysical functions.

```
psi (numpy array): psychophysical function psi(x)
    """
    F = 1 - np.exp(-(x/alpha) ** beta)
    psi = gam + (1. - lam - gam) * F
    return (psi)

X = np.arange(0, 5, .01)

psi1 = psiphys(X, lam=.2, gam=.4)
```

The line `X = np.arange(0, 5, .01)` is highlighted with a red box and is the current line of execution, indicated by a red dot in the gutter. The SciView panel shows two plots: one for the psychophysical function and another for the accuracy.

SciView: Data Plots
psychophysical function
accuracy
stimulus manipulation

Documentation: return

The return statement

```
return_stmt ::= "return" [expression_list]
return may only occur syntactically nested in a function definition, not within a nested class definition.
If an expression list is present, it is evaluated, else None is substituted.
return leaves the current function call with the expression list (or None) as return value.
When return passes control out of a try statement with a finally clause, that finally clause is executed before really leaving the function.
In a generator function, the return statement indicates that
```

Debug: MyTest

Debugger, Console, Frames, Variables

MainThread, <module>, MyTest.py:25

Event Log

Packages installed successfully: Installed packages: 'scipy' (45 minutes ago)

PyCharm File Edit View Navigate Code Refactor Run Tools VCS Window Help

ForClass – MyTest.py

ForClass > MyTest.py

Project 1: Project ~Desktop/mss/Course/P6 17

Structure

1: Project

ForClass

- data
- models
- notebooks
- main.py
- MyTest.py
- README.md
- requirements.txt

External Libraries

Scratches and Consoles

SciView: Data Plots 4-bit color 21.5 kB

psychophysical function

accuracy stimulus manipulation

Documentation: return

The return statement

return_stmt ::= "return" [expression_list]

return may only occur syntactically nested in a function definition, not within a nested class definition.

If an expression list is present, it is evaluated, else None is substituted.

return leaves the current function call with the expression list (or None) as return value.

When return passes control out of a try statement with a finally clause, that finally clause is executed before really leaving the function.

Debug: MyTest

Debugger

Frames

MainThread

<module>, MyTest.py:25

Variables

Special Variables

AWS Explorer

TODO Debug Terminal Python Console Event Log

The screenshot shows the PyCharm IDE interface. The top navigation bar includes PyCharm, File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, and Help. The title bar says 'ForClass – MyTest.py'. The left sidebar has a 'Project' view with a tree structure for the 'ForClass' project, including subfolders like 'data', 'models', 'notebooks', 'main.py', 'MyTest.py', 'README.md', and 'requirements.txt'. Below this is an 'External Libraries' section and a 'Scratches and Consoles' section. The main code editor window shows 'MyTest.py' with code related to a psychophysical function. A red highlight is on line 25: 'x = np.arange(0, 5, .01)'. To the right of the code editor is a 'SciView' panel titled 'psychophysical function' showing a plot of accuracy versus stimulus manipulation. The bottom of the screen features a 'Debug' toolbar with various icons for debugging, and the status bar at the bottom right shows 'Event Log'.

notice that the bottom has switched into Debug mode

PyCharm File Edit View Navigate Code Refactor Run Tools VCS Window Help

ForClass – MyTest.py

ForClass > MyTest.py

Project 1: Project ~Desktop/mss/Course/P6 17

1: Structure

1: SciView: Data Plots

1: SciView

1: Database

1: Documentation

1: The return statement

1: return_stmt ::= "return" [expression_list]

1: return may only occur syntactically nested in a function definition, not within a nested class definition.

1: If an expression list is present, it is evaluated, else None is substituted.

1: return leaves the current function call with the expression list (or None) as return value.

1: When return passes control out of a try statement with a finally clause, that finally clause is executed before really leaving the function.

1: In a generator function, the return statement indicates that

1: Documentation

1: Psi (numpy array): psychophysical function psi(x)

1: """

1: F = 1 - np.exp(-(x/alpha) ** beta)

1: psi = gam + (1. - lam - gam) * F

1: return (psi)

1: %%

1: X = np.arange(0, 5, .01)

1: psi1 = psiphys(X)

1: psi2 = psiphys(X, lam=.2, gam=.4)

1: %%

1: plt.plot(X, psi1, 'r-', X, psi2, 'g-')

1: plt.xlabel("stimulus manipulation")

1: plt.ylabel("accuracy")

1: plt.title("psychophysical function")

1: plt.ylim(.4, 1)

1: plt.show()

1: Documentation: return

1: The return statement

1: return_stmt ::= "return" [expression_list]

1: return may only occur syntactically nested in a function definition, not within a nested class definition.

1: If an expression list is present, it is evaluated, else None is substituted.

1: return leaves the current function call with the expression list (or None) as return value.

1: When return passes control out of a try statement with a finally clause, that finally clause is executed before really leaving the function.

1: In a generator function, the return statement indicates that

1: Documentation

Debug: MyTest

Debugger Console

Frames

MainThread <module>, MyTest.py:25

Variables

2: Favorites

2: AWS Explorer

2: Terminal

2: Python Console

2: Event Log

2: Packages installed successfully: 'scipy' (45 minutes ago)

25:1 LF UTF-8 AWS: No credentials selected 4 spaces Python 3.8 (ForClass)

can still access the Terminal

PyCharm File Edit View Navigate Code Refactor Run Tools VCS Window Help

ForClass – MyTest.py

ForClass > MyTest.py

Project 1: Project ~Desktop/mss/Course/P6 17

1: Structure

1: SciView: Data Plots

1: SciView

1: Database

1: Documentation

1: The return statement

1: return_stmt ::= "return" [expression_list]

1: return may only occur syntactically nested in a function definition, not within a nested class definition.

1: If an expression list is present, it is evaluated, else None is substituted.

1: return leaves the current function call with the expression list (or None) as return value.

1: When return passes control out of a try statement with a finally clause, that finally clause is executed before really leaving the function.

1: In a generator function, the return statement indicates that

1: Documentation

1: TODO 5: Debug Terminal Python Console

1: Packages installed successfully: Installed pack... (2 days ago)

1: Event Log

psi (numpy array): psychophysical function psi(x)

```
18     """
19     F = 1 - np.exp(-(x/alpha) ** beta)
20     psi = gam + (1. - lam - gam) * F
21     return (psi)
22
23 #%%
24
25 X = np.arange(0, 5, .01)
26 psi1 = psiphys(X)
27 psi2 = psiphys(X, lam=.2, gam=.4)
28
29 #%%
30
31 plt.plot(X, psi1, 'r-', X, psi2, 'g-')
32 plt.xlabel("stimulus manipulation")
33 plt.ylabel("accuracy")
34 plt.title("psychophysical function")
35 plt.ylim(.4, 1)
36 plt.show()
```

Debug: MyTest

Debugger Console

Frames

MainThread <module>, MyTest.py:25

Variables

2: Favorites

2: AWS Explorer

2: Documentation

and the independent Python Console

PyCharm File Edit View Navigate Code Refactor Run Tools VCS Window Help

ForClass – MyTest.py

ForClass > MyTest.py

Project 1: Project ~Desktop/mss/Course/P6 17

1: Structure

1: SciView: Data Plots

1: SciView

1: Database

1: Documentation

1: The return statement

1: return_stmt ::= "return" [expression_list]

1: return may only occur syntactically nested in a function definition, not within a nested class definition.

1: If an expression list is present, it is evaluated, else None is substituted.

1: return leaves the current function call with the expression list (or None) as return value.

1: When return passes control out of a try statement with a finally clause, that finally clause is executed before really leaving the function.

1: In a generator function, the return statement indicates that

```
psi (numpy array): psychophysical function psi(x)
"""
F = 1 - np.exp(-(x/alpha) ** beta)
psi = gam + (1. - lam - gam) * F
return (psi)

#%
x = np.arange(0, 5, .01)
psi1 = psiphys(x)
psi2 = psiphys(x, lam=.2, gam=.4)

#%
plt.plot(x, psi1, 'r-', x, psi2, 'g-')
plt.xlabel("stimulus manipulation")
plt.ylabel("accuracy")
plt.title("psychophysical function")
plt.ylim(.4, 1)
plt.show()
```

Debug: MyTest

Debugger Console

Frames

MainThread

<module>, MyTest.py:25

Variables

Special Variables

2: Favorites

2: AWS Explorer

2: Documentation

6: TODO 5: Debug 4: Terminal Python Console

Packages installed successfully: Installed packages: 'scipy' (45 minutes ago)

25:1 LF UTF-8 AWS: No credentials selected 4 spaces Python 3.8 (ForClass) Event Log

The screenshot shows the PyCharm IDE interface. The top navigation bar includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, and Help. The main window displays a project named 'ForClass' with files like 'main.py' and 'MyTest.py'. The code editor shows Python code for calculating psychophysical functions and plotting them. A SciView panel on the right displays a plot titled 'psychophysical function' showing accuracy versus stimulus manipulation for two conditions. The Debug panel at the bottom shows the current stack trace, and the AWS Explorer panel shows favorite AWS services. The bottom status bar indicates the file is 25:1, uses LF line endings, is in UTF-8 encoding, and is using Python 3.8.

the Console tied to MyTest.py is here

PyCharm File Edit View Navigate Code Refactor Run Tools VCS Window Help

Thomas Palmeri Wed Sep 23 6:37 PM

ForClass – MyTest.py

ForClass > MyTest.py

Project Structure

1: Project ForClass ~/Desktop/mss/Course/ForClass

2: Structure

MyTest.py

```
19 F = 1 - np.exp(-(x/alpha) ** beta)
20 psi = gam + (1. - lam - gam) * F
21 return (psi)
22
23 #%%
24
25 x = np.arange(0, 5, .01)
26 psi1 = psiphys(x)
27 psi2 = psiphys(x, lam=.2, gam=.4)
28
29 #%%
30
31 plt.plot(x, psi1, 'r-', x, psi2, 'g-')
32 plt.xlabel("stimulus manipulation")
33 plt.ylabel("accuracy")
34 plt.title("psychophysical function")
35 plt.ylim((.4, 1))
36 plt.show()
37
```

SciView: Data Plots

4-bit color 21.5 kB

psychophysical function

accuracy

stimulus manipulation

Documentation: return

The return statement

```
return_stmt ::= "return" [expression_list]
return may only occur syntactically nested in a function definition, not within a nested class definition.
If an expression list is present, it is evaluated, else None is substituted.
return leaves the current function call with the expression list (or None) as return value.
When return passes control out of a try statement with a finally clause, that finally clause is executed before really leaving the function.
In a generator function, the return statement indicates that
```

Debug: MyTest

Debugger Console

```
/Users/palmerit/.conda/envs/ForClass/bin/python /Applications/PyCharm.app/Contents/plugins/python/helpers/pydev/pydevd.py --cmd-line --multiproc --qt-support=auto --client 127.0.0.1 --port 65159 --file "/Users/palmerit/PycharmProjects/ForClass/MyTest.py"
```

pydev debugger: process 85316 is connecting

Connected to pydev debugger (build 201.8743.11)

AWS Explorer

2: Favorites

Event Log

TODO Debug Terminal Python Console

Packages installed successfully: Installed packages: 'scipy' (48 minutes ago)

PyCharm File Edit View Navigate Code Refactor Run Tools VCS Window Help

ForClass – MyTest.py

ForClass > MyTest.py

Project 1: Project ~Desktop/mss/Course/P6 17

Structure

1: Project ForClass ~Desktop/mss/Course/P6 17

data

models

notebooks

main.py

MyTest.py

README.md

requirements.txt

External Libraries

Scratches and Consoles

SciView: Data Plots 4-bit color 21.5 kB

psychophysical function

accuracy

stimulus manipulation

Documentation: return

The return statement

return_stmt ::= "return" [expression_list]

return may only occur syntactically nested in a function definition, not within a nested class definition.

If an expression list is present, it is evaluated, else None is substituted.

return leaves the current function call with the expression list (or None) as return value.

When return passes control out of a try statement with a finally clause, that finally clause is executed before really leaving the function.

In a generator function, the return statement indicates that

Debug: MyTest

Debugger Console

Frames

MainThread

<module>, MyTest.py:25

Variables

AVS Explorer

TODO Debug Terminal Python Console

Packages installed successfully: Installed packages: 'scipy' (45 minutes ago)

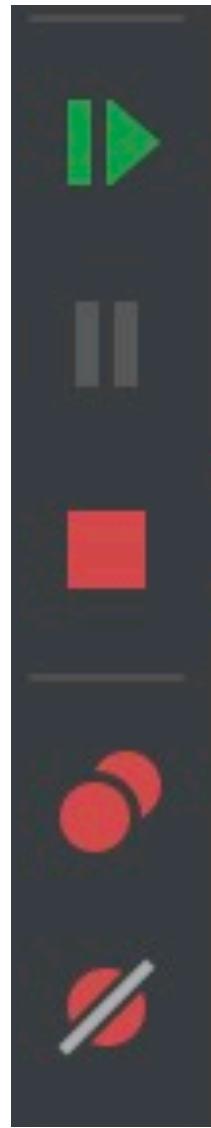
Event Log

25:1 LF UTF-8 AWS: No credentials selected 4 spaces Python 3.8 (ForClass)

This screenshot shows the PyCharm IDE interface. The top navigation bar includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, and Help. The main window displays a project named 'ForClass' with files like main.py and MyTest.py. The code editor shows a Python script with a function 'psi' and a plot of accuracy vs stimulus manipulation. The SciView panel on the right shows three plots of the psychophysical function. The bottom section features the Debug tool window, which is highlighted with a red box, showing the stack trace and variables for the current frame. Other panels like TODO, Terminal, and Python Console are also visible at the bottom.

these are core debug tools (for stepping through the code)

resume
program



pause
program



stop
program



view
breakpoints



mute
breakpoints



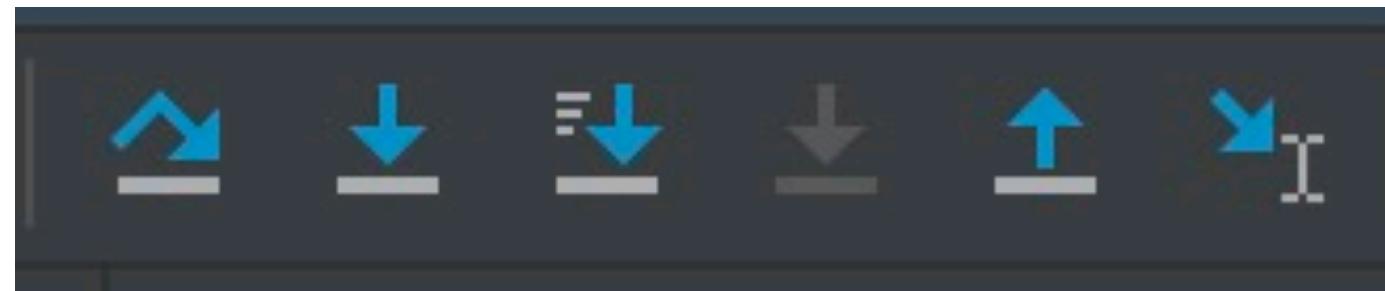
step
over

step
into

step
into
my code

step
out

run
to
cursor



variable values are shown in the code and in the Variables window

The screenshot shows the PyCharm IDE interface with the following components:

- Top Bar:** PyCharm, File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help.
- Toolbar:** Standard file operations (New, Open, Save, Find, etc.).
- Project View:** Shows the project structure with a folder named "ForClass" containing files like "main.py", "MyTest.py", and "README.md".
- Code Editor:** The "MyTest.py" file is open, showing Python code for calculating a psychophysical function and plotting it. A red box highlights the plot command and its output.
- SciView:** A panel showing a plot titled "psychophysical function" with "accuracy" on the y-axis (ranging from 0.4 to 1.0) and "stimulus manipulation" on the x-axis (ranging from 0 to 5). It displays two curves: a red curve peaking at ~0.9 and a green curve peaking at ~0.8. Below the plot, documentation for the "return" statement is visible.
- Variables View:** A panel titled "Variables" showing the current state of variables. It lists "psi1" (ndarray of shape (500,) with values [0.5, 0.50004, ...]), "psi2" (ndarray of shape (500,) with values [0.4, 0.40004, ...]), "x" (ndarray of shape (500,) with values [0., 0.01, 0.02, ..., 0.17]), and "Special Variables".
- Bottom Status Bar:** Shows file status (25:1 LF), encoding (UTF-8), AWS credentials, and Python version (Python 3.8 (ForClass)).

PyCharm File Edit View Navigate Code Refactor Run Tools VCS Window Help

Thomas Palmeri Wed Sep 23 7:51 PM

ForClass – MyTest.py

Project: ForClass ~/Desktop/mss/Course/Python/ForClass

Structure: data, models, notebooks, main.py, MyTest.py, README.md, requirements.txt, External Libraries, Scratches and Consoles

SciView: Data Plots 4-bit color 21.5 kB

psychophysical function

accuracy stimulus manipulation

The return statement

return_stmt ::= "return" [expression_list]

return may only occur syntactically nested in a function definition, not within a nested class definition.

If an expression list is present, it is evaluated, else None is substituted.

return leaves the current function call with the expression list (or None) as return value.

When return passes control out of a try statement with a finally clause, that finally clause is executed before really leaving the function.

In a generator function, the return statement indicates that

Arguments:

```
def psiphys(x, alpha=1., beta=2., gam=.5, lam=.1): x: [0. 0.01 0.02 0.03 0.04 0.05 0.06 0.07 0.08 0.09 0.1 0.11 0.12
```

"""\Computes psychophysical function

Arguments:

```
        x (numpy array): point at which to evaluate psi(x)
```

alpha (float): scale parameter of the Weibull

beta (float): shape parameter of the Weibull

gam (float): chance (floor on performance)

lam (float): lapse rate (ceil on performance)

Returns:

```
        psi (numpy array): psychophysical function psi(x)
```

"""

```
F = 1 - np.exp(-(x/alpha) ** beta)
psi = gam + (1. - lam - gam) * F
return (psi)
```

psiphys()

Variables

MainThread

psiphys, MyTest.py:19

<module>, MyTest.py:26

alpha = {float} 1.0
beta = {float} 2.0
gam = {float} 0.5
lam = {float} 0.1

x = {ndarray: (500,)} [0. 0.01 0.02 0.03 0.04 0.05 0.06 0.07 0.08 0.09 0.1 0.11 0.12 0.13, 0.14 0.15 0.16 0.17 0.18 0.19 0.2 0.21 0.22 0.23 0.24 0.25 0.26 0.27, 0.28 0.29 0.3 0.31 0.32 0.33 0.34 0.35 0.36 0.37 ...View as Array

Event Log

Packages installed successfully: Installed packages: 'scipy' (today 5:49 PM)

25:1 LF UTF-8 AWS: No credentials selected 4 spaces Python 3.8 (ForClass)

stepping into a function, you see the local variables

PyCharm File Edit View Navigate Code Refactor Run Tools VCS Window Help

ForClass – MyTest.py

Project: ForClass ~/Desktop/mss/Course/ForClass

Structure: 1: Project ForClass data notebooks main.py MyTest.py README.md requirements.txt External Libraries Scratches and Consoles

SciView: Data Plots 4-bit color 21.5 kB SciView Database

ForClass > MyTest.py

```

def psiphys(x, alpha=1., beta=2., gam=.5, lam=.1):
    """Computes psychophysical function
    Arguments:
        x (numpy array): point at which to evaluate psi(x)
        alpha (float): scale parameter of the Weibull
        beta (float): shape parameter of the Weibull
        gam (float): chance (floor on performance)
        lam (float): lapse rate (ceil on performance)

    Returns:
        psi (numpy array): psychophysical function psi(x)
    """
    F = 1 - np.exp(-(x/alpha) ** beta)
    psi = gam + (1. - lam - gam) * F
    return psi

```

The return statement

`return_stmt ::= "return" [expression_list]`

return may only occur syntactically nested in a function definition, not within a nested class definition.

If an expression list is present, it is evaluated, else None is substituted.

return leaves the current function call with the expression list (or None) as return value.

When return passes control out of a try statement with a finally clause, that finally clause is executed before really leaving the function.

In a generator function, the return statement indicates that

Debug: MyTest

Variables:

- MainThread
- psiphys, MyTest.py:19
- <module>, MyTest.py:26
- x = [ndarray: (500,)] [0. 0.01 0.02 0.03 0.04 0.05 0.06 0.07 0.08 0.09 0.1 0.11 0.12 0.13, 0.14 0.15 0.16 0.17 0.18 0.19 0.2 0.21 0.22 0.23 0.24 0.25 0.26 0.27, 0.28 0.29 0.3 0.31 0.32 0.33 0.34 0.35 0.36 0.37 ...View as Array

AWS Explorer

Event Log

Packages installed successfully: Installed packages: 'scipy' (today 5:49 PM)

25:1 LF UTF-8 AWS: No credentials selected 4 spaces Python 3.8 (ForClass)

you can move through the "stack", to see variables outside the function

PyCharm File Edit View Navigate Code Refactor Run Tools VCS Window Help

ForClass – MyTest.py

Project: ForClass ~/Desktop/mss/Course/ForClass

Structure: 1: Project, 2: Structure

SciView: Data Plots

Documentation: return

The return statement

```
return_stmt ::= "return" [expression_list]
return may only occur syntactically nested in a function definition, not within a nested class definition.
If an expression list is present, it is evaluated, else None is substituted.
return leaves the current function call with the expression list (or None) as return value.
When return passes control out of a try statement with a finally clause, that finally clause is executed before really leaving the function.
In a generator function, the return statement indicates that
```

Debug: MyTest

Debugger Console

```
/Users/palmeri/.conda/envs/ForClass/bin/python /Applications/PyCharm.app/Contents/plugins/python/helpers/pydev/pydevd.py --cmd-line --multiproc --qt-support=auto --client 127.0.0.1 --port 56427 --file "/Users/palmeri/Desktop/mss/Course/ForClass/MyTest.py"
```

Connected to pydev debugger (build 201.8743.11)

```
>>> alpha
1.0
>>> alpha = 5
>>> alpha
5
>>>
```

Todo: 6: TODO

Debug: 5: Debug

Terminal: Python Console

Packages installed successfully: Installed packages: 'scipy' (today 5:49 PM)

Event Log: 1:1 LF UTF-8 AWS: No credentials selected 4 spaces Python 3.8 (ForClass)

This screenshot shows the PyCharm IDE interface. The top navigation bar includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, and Help. The title bar says 'ForClass – MyTest.py'. The left sidebar has 'Project' and 'Structure' sections. The main code editor shows 'MyTest.py' with code related to psychophysical functions. A SciView panel on the right displays three line graphs titled 'psychophysical function' showing accuracy versus stimulus manipulation. The bottom section shows a debugger console with variable values and a terminal window showing the connection to a pydev debugger.

can view and change values of variables in the Console (tied to MyTest.py)

PyCharm File Edit View Navigate Code Refactor Run Tools VCS Window Help

ForClass – MyTest.py

ForClass > MyTest.py

Project

1: Project

1: Structure

1: SciView: Data Plots

SciView: Data Plots

SciView Database

ForClass ~Desktop/mss/Course/6

data

models

notebooks

main.py

MyTest.py

README.md

requirements.txt

External Libraries

Scratches and Consoles

ForClass – MyTest.py

alpha (float): scale parameter of the Weibull
beta (float): shape parameter of the Weibull
gam (float): chance (floor on performance)
lam (float): lapse rate (ceil on performance)

Returns:
 psi (numpy array): psychophysical function psi(x)

"""

F = 1 - np.exp(-(x/alpha) ** beta)
psi = gam + (1. - lam - gam) * F
return (psi)

#%

x = np.arange(0., 5., .01)
psi1 = psiphys(x)
psi2 = psiphys(x, lam=.2, gam=.4)

#%

plt.plot(x, psi1, 'r-', x, psi2, 'g-')
plt.xlabel("stimulus manipulation")
plt.ylabel("accuracy")
plt.title("psychophysical function")
psiphys()

The return statement

return_stmt ::= "return" [expression_list]
return may only occur syntactically nested in a function definition, not within a nested class definition.
If an expression list is present, it is evaluated, else None is substituted.
return leaves the current function call with the expression list (or None) as return value.
When return passes control out of a try statement with a finally clause, that finally clause is executed before really leaving the function.
In a generator function, the return statement indicates that

Debug: MyTest

Debugger Console

Frames Variables

Connected

Frames are not available

AWS Explorer Favorites

6: TODO 5: Debug 2: Terminal Python Console

Packages installed successfully: Installed packages: 'scipy' (today 5:49 PM)

19:1 LF UTF-8 AWS: No credentials selected 4 spaces Python 3.8 (ForClass) Event Log

This screenshot shows the PyCharm IDE interface. The top navigation bar includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, and Help. The main window displays the code for 'MyTest.py' in the 'ForClass' project. The code defines a function 'psiphys' with parameters alpha, beta, gam, and lam, and returns a numpy array representing a psychophysical function. A break point is set at line 25. The SciView panel on the right shows a plot titled 'psychophysical function' with two sigmoidal curves (green and red) representing accuracy versus stimulus manipulation. The bottom panels show the Debug and AWS Explorer toolbars.

PyCharm File Edit View Navigate Code Refactor Run Tools VCS Window Help

ForClass – MyTest.py

ForClass > MyTest.py

Project: ForClass ~/Desktop/mss/Course/P6219 Scientific Computing for Psychology/Python/ForClass

Structure: main.py > MyTest.py

```

1 import matplotlib.pyplot as plt
2
3 #%%
4
5 def psiphys(x, alpha=1., beta=2., gam=.5, lam=.1):
6     """Computes psychophysical function
7
8     Arguments:
9         x (numpy array): point at which to evaluate psi(x)
10        alpha (float): scale parameter of the Weibull
11        beta (float): shape parameter of the Weibull
12        gam (float): chance (floor on performance)
13        lam (float): lapse rate (ceil on performance)
14
15
16     Returns:
17         psi (numpy array): psychophysical function psi(x)
18     """
19
20     F = 1 - np.exp(-(x/alpha) ** beta)
21     psi = gam + (1. - lam - gam) * F
22
23 #%%
24
25 x = np.arange(0, 5, .01)

```

SciView: Data Plots 4-bit color 21.5 kB

psychophysical function

Documentation: beta

Parameter beta of MyTest.psiphys
beta: float
shape parameter of the Weibull

Python Console

```

MyTest > MyTest (1) > MyTest (2) > MyTest (3) > MyTest (4) >

```

```

/u... /Users/palmerit/Desktop/mss/Course/P6219 Scientific Computing for Psychology/Python/ForClass/Contents/plugins/python/helpers/pydev/pydevconsole.py --mode=cl
import sys; print('Python %s on %s' % (sys.version, sys.platform))
sys.path.append(['/Users/palmerit/Desktop/mss/Course/P6219 Scientific Computing for Psychology/Python/ForClass/MyTest.py', wdir=''/Users/palme...

```

Special Variables

psi1 = {ndarray: (500,)} [0.5 0.50004 0.50015997 0.50036...View as Array

psi2 = {ndarray: (500,)} [0.4 0.40004 0.40015997 0.40036...View as Array

x = {ndarray: (500,)} [0. 0.01 0.02 0.03 0.04 0.05 0.06 0.07 0.0...View as Array

AWS Explorer

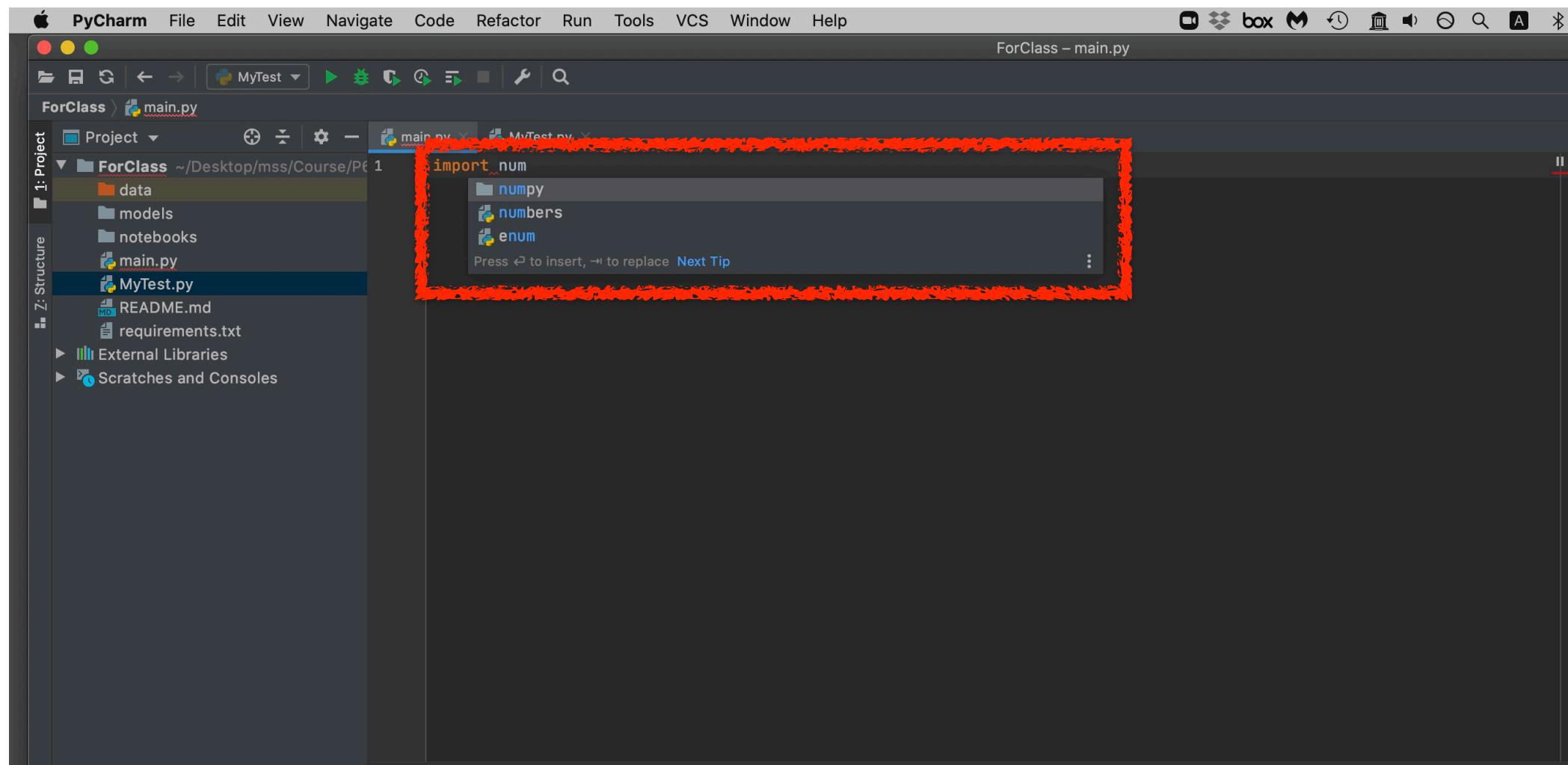
Event Log

Packages installed successfully: Installed packages: 'scipy' (today 5:49 PM)

25:1 LF UTF-8 AWS: No credentials selected 4 spaces Python 3.8 (ForClass)

each time you run the .py file, you will create a new Console (so, you'll want to close them)

autocomplete as you're typing in code (Tab to complete, or use arrows and Enter to select)



The screenshot shows the PyCharm IDE interface with the following details:

- Top Bar:** PyCharm, File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help.
- Toolbar:** Standard icons for file operations like Open, Save, Run, and Find.
- Project Bar:** Shows the project structure under "ForClass".
- Code Editor:** The file "main.py" is open, containing the following code:

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3
4 def psiphys(x, alpha=1., beta=2., gam=.5, lam=.1):
5     """Computes psychophysical function
6
7     Arguments:
8         x (numpy array): point at which to evaluate psi(x)
9         alpha (float): scale parameter of the Weibull
10        beta (float): shape parameter of the Weibull
11        gam (float): chance (floor on performance)
12        lam (float): lapse rate (ceil on performance)
13
14    Returns:
15        psi (numpy array): psychophysical function psi(x)
16    """
17    F = 1 - np.exp(-(x/alpha) ** beta)
18    psi = gam + (1. - lam - gam) * F
19    return (psi)
20
21 x = np.arange(0, 5, .01)
22 psi1 = psiphys
23 f psiphys(x, alpha, beta, gam, lam)
```
- Completion Overlay:** A red box highlights the code "f psiphys(x, alpha, beta, gam, lam)". A tooltip appears below it with the text "Press ^ to choose the selected (or first) suggestion and insert a dot afterwards Next Tip".

autocomplete as you're typing in code (Tab to complete, or use arrows and Enter to select)

PyCharm File Edit View Navigate Code Refactor Run Tools VCS Window Help

ForClass – main.py

ForClass > main.py

Project main.py MyTest.py

1: Project 1: ForClass ~~/Desktop/mss/Course/P 1
2: data
3: models
4: notebooks
5: main.py
6: MyTest.py
7: README.md
8: requirements.txt

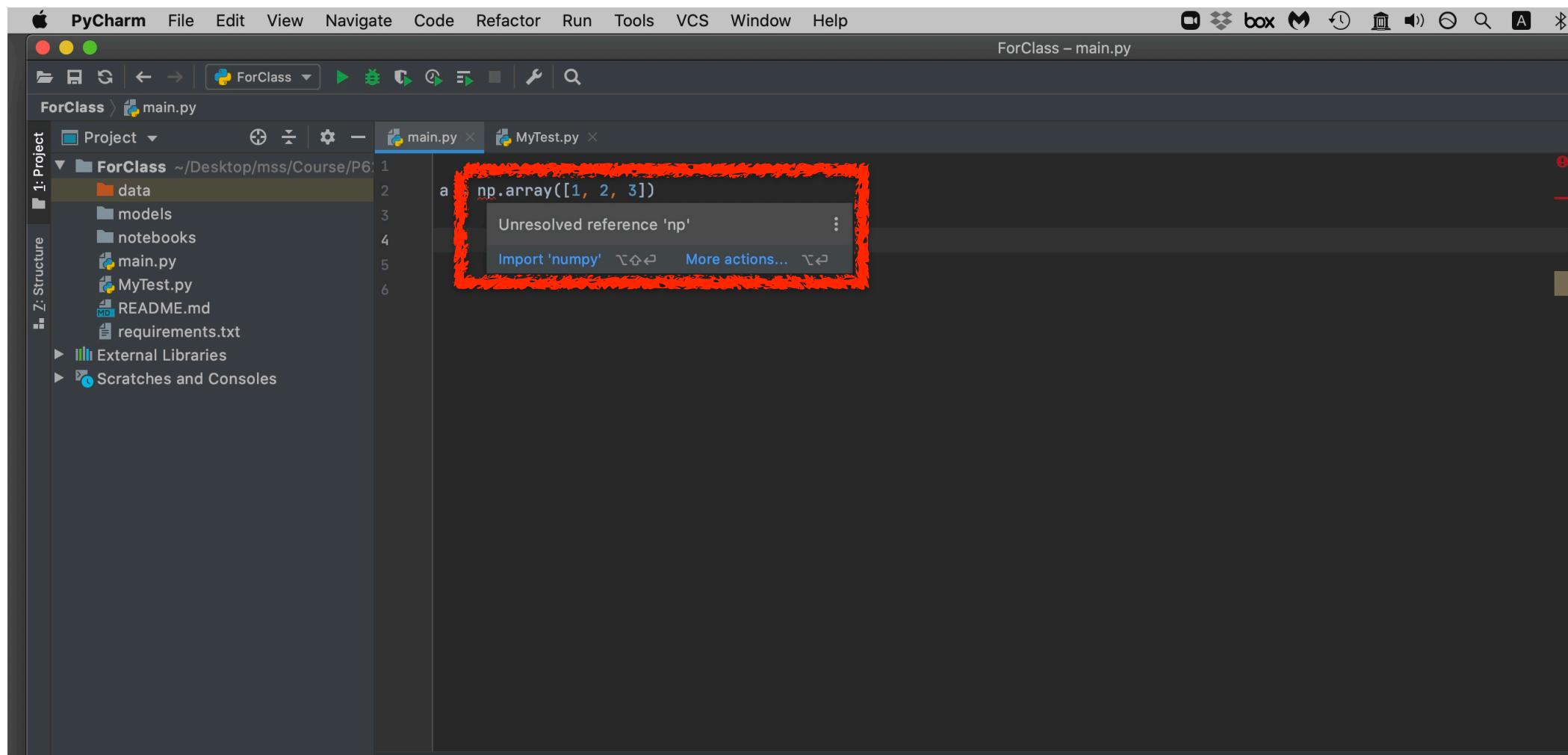
External Libraries
Scratches and Consoles

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5])
print(arr)
```

v arr
bytearray
builtins
Press ⌘ to insert, ⌘ to replace Next Tip

The screenshot shows the PyCharm IDE interface. The top menu bar includes Apple logo, PyCharm, File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, and Help. The title bar says "ForClass – main.py". Below the menu is a toolbar with various icons. The left sidebar has sections for "Project" and "Structure". The "Project" section shows a folder named "ForClass" containing subfolders "data", "models", "notebooks", and files "main.py" and "MyTest.py", along with "README.md" and "requirements.txt". The "Structure" section shows "External Libraries" and "Scratches and Consoles". The main editor area contains Python code: "import numpy as np", "arr = np.array([1, 2, 3, 4, 5])", and "print(arr)". A tooltip is displayed over the line "print(arr)" with the text "v arr", "bytearray", and "builtins". A red box highlights this tooltip. At the bottom of the tooltip, it says "Press ⌘ to insert, ⌘ to replace Next Tip".

can automatically import a (missing) module (in this case numpy)



can automatically import a (missing) module (in this case math)

The screenshot shows the PyCharm IDE interface. The top menu bar includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, and Help. The title bar indicates the file is "ForClass – main.py". The left sidebar has sections for Project, Z: Structure, External Libraries, and Scratches and Consoles. The Project section shows a directory structure for "ForClass" containing "data", "models", "notebooks", "main.py", "MyTest.py", "README.md", and "requirements.txt". The main code editor window displays the following Python code:

```
a = np.array([1, 2, 3])
b = math.sin(2.4)
```

The line "b = math.sin(2.4)" is highlighted with a red border, and a tooltip appears over the "math" reference, stating "Unresolved reference 'math'". Below the tooltip are buttons for "Import 'math'" and "More actions...".



post questions about using PyCharm on Piazza

a module in Python

- a module in Python is most simply a .py file that contains variables and/or functions that can be called from another program using an `import` command
- follow the same rules as importing and using modules like numpy, matplotlib, only here we're talking about a module you might create and use yourself

```
import mymodule
```

```
import mymodule as mm
```

```
from mymodule import myfun, myvar
```

see `PsiPhys.py` and `TestPsiPhys.py`

test/demo within a module (`__main__`)

imports

variable definitions

function definitions

```
if __name__ == '__main__':
```

things to run to test/demo the module

*this is only executed when file run on its own,
not used when imported as a module*