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

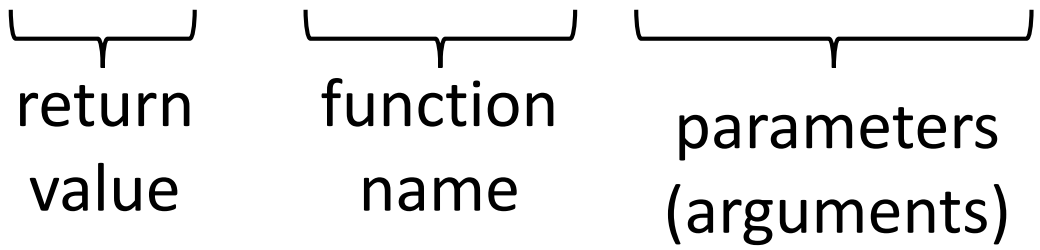
Why do we use functions?

- 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)
```

return values

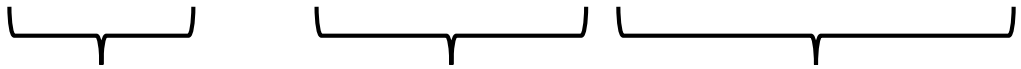
function name

parameters (arguments)

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)
```

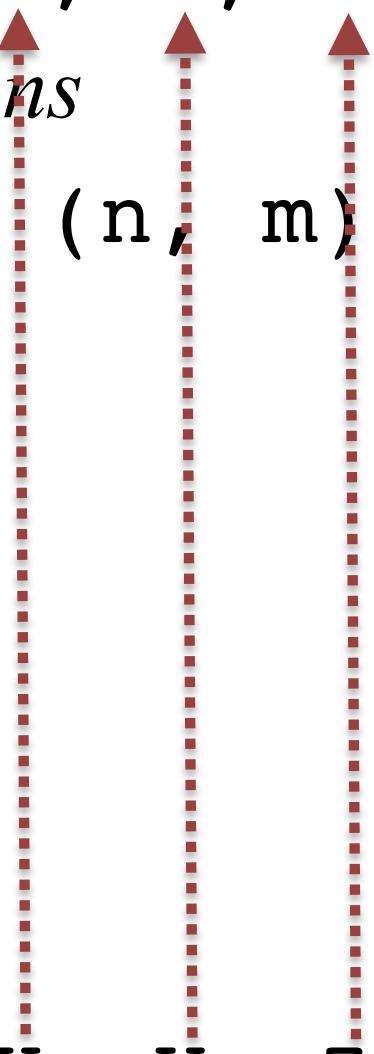
functions in Python

order matters (for arguments without names)

```
def myfun(d, e, f):
    expressions
    return (n, m)

.
.
.

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

colon required

indenting



functions in Python

order matters

```
def myfun(d, e, f):
    expressions
    return (n, m)
```

if returning more than one variable, returns as a tuple

return required if returning values

```
.
.
.
```

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

functions in Python

order matters

```
def myfun(d, e, f):
    expressions
    return (n, m)
```

if returning more than one variable, returns as a tuple

return required if returning values

.
.
.

```
(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 ...

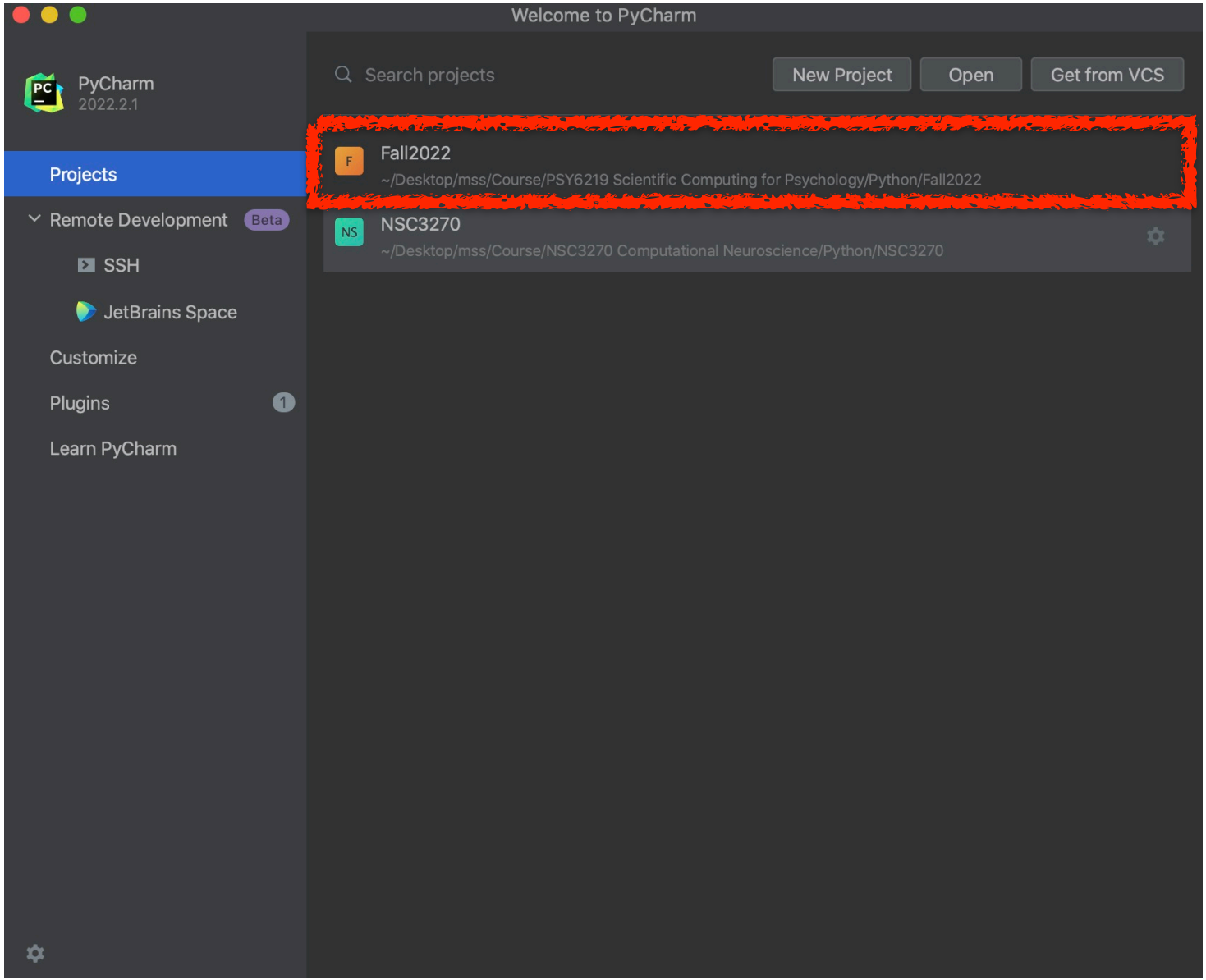
- Hard to debug code (`.py` files in fully-functional IDEs are way better) - `print` statements and weak debugging magics (we did not talk about) only go so far
- Harder to share code (copy and paste from one Notebook to another)
- Git/Github are possible (but less natural, more cumbersome, more as backups 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)

MyPsiPhys.py

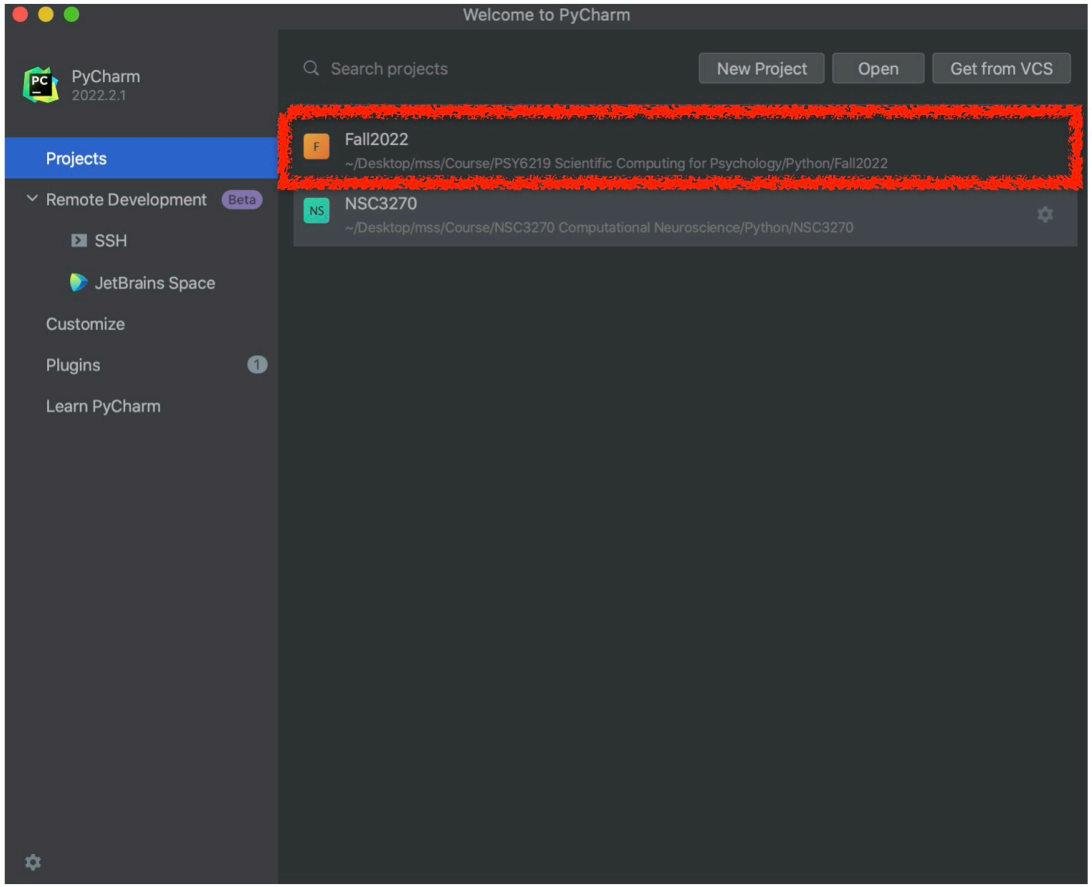
```
import numpy as np
import matplotlib.pyplot as plt


# %%

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

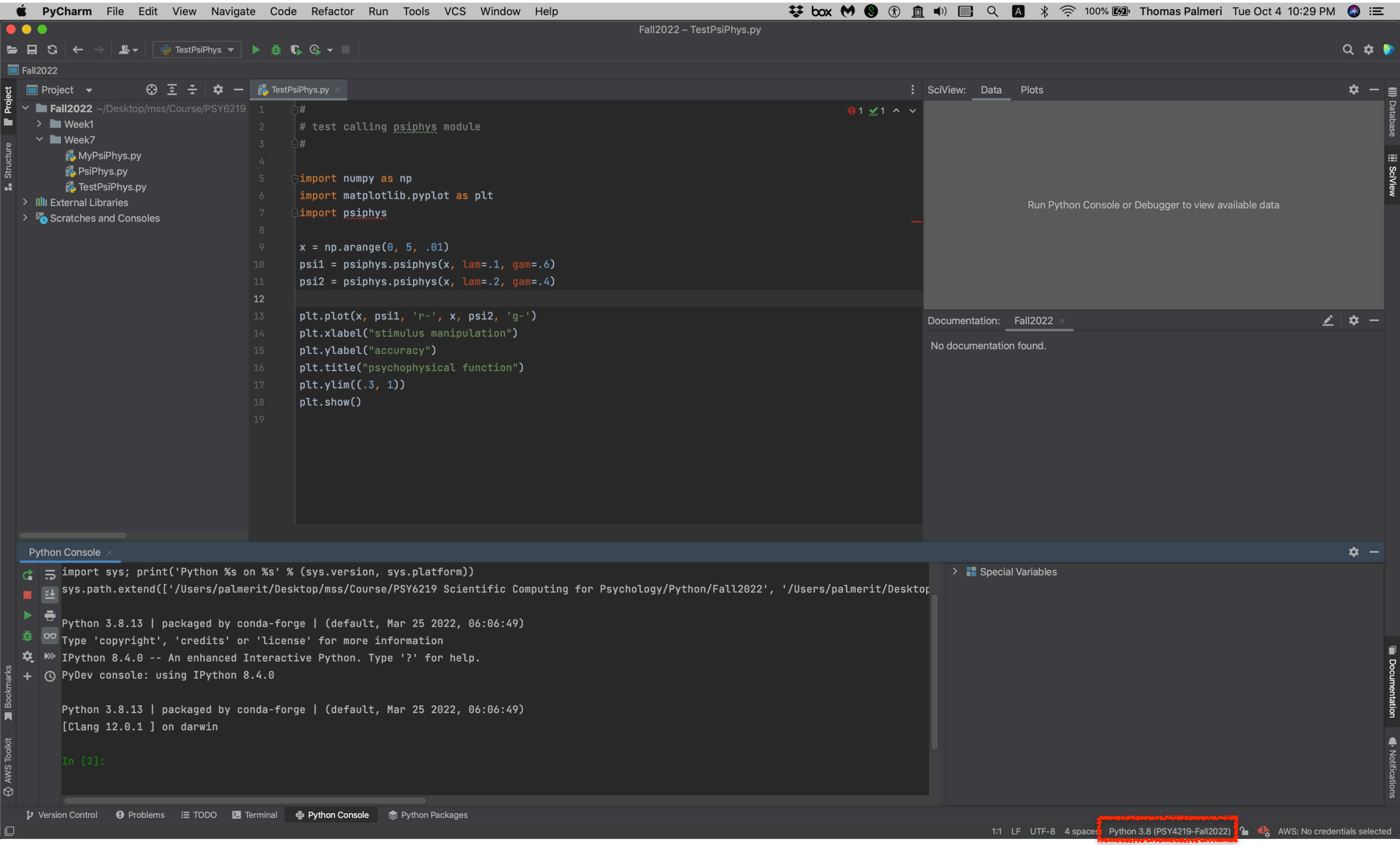


it may just open right up to your project
(you get this when a project was closed)

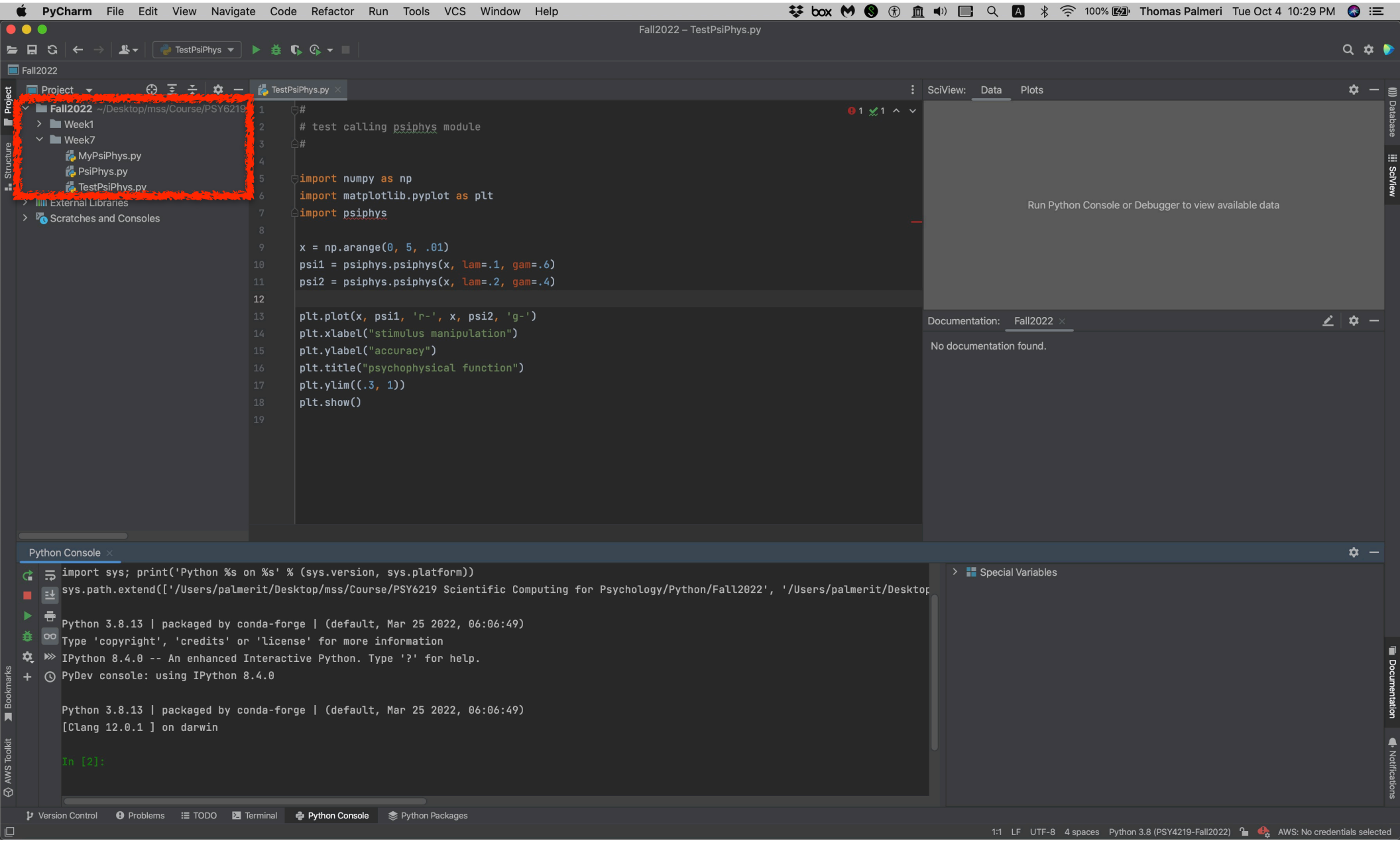
starting PyCharm

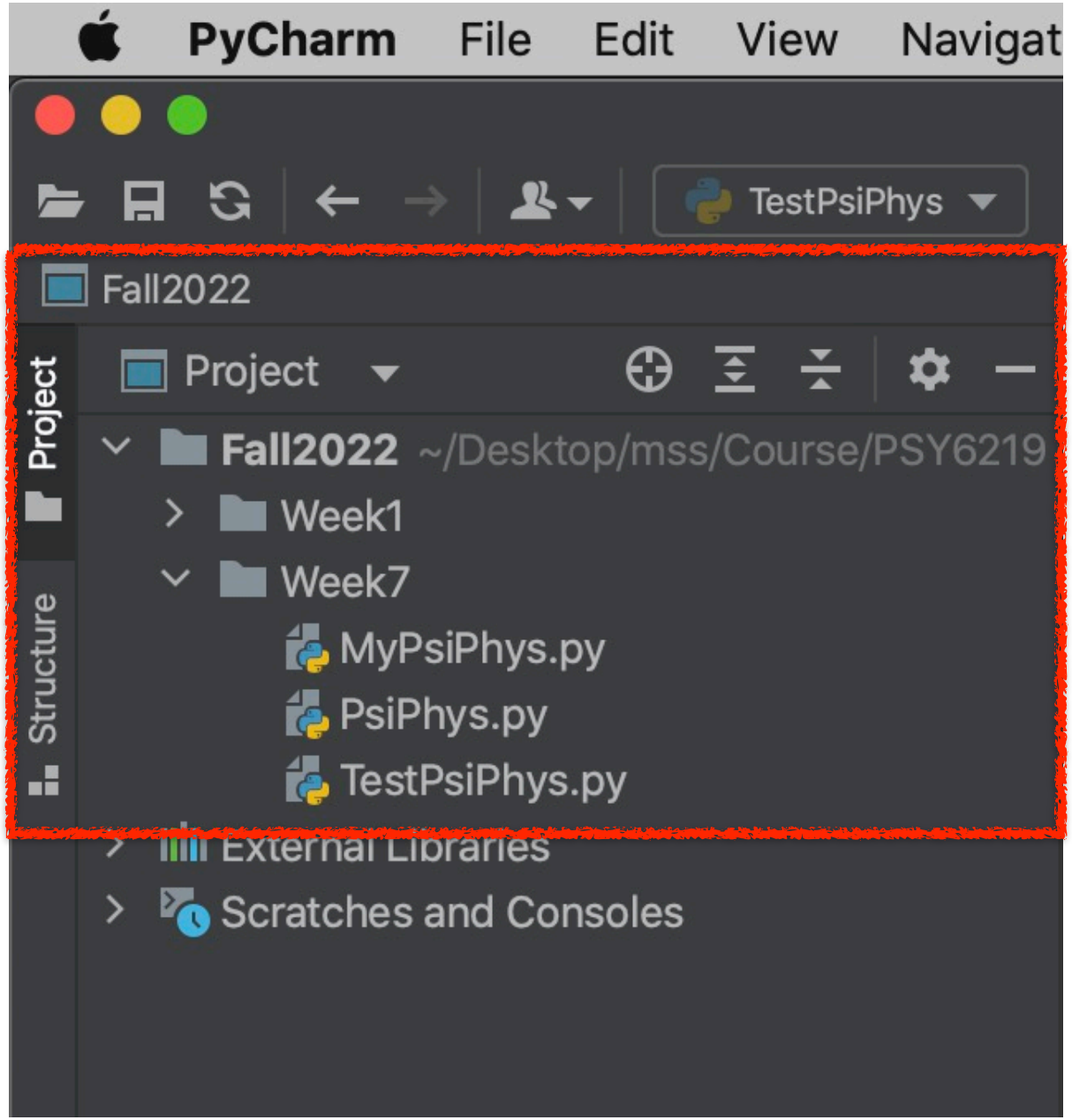


- a "Project" resides in a folder and can include many .py files
- it can be one interconnected set of code (many .py files)
- or a bunch of independent folders and .py files sharing the same environment (which is what I do for much of the course code)



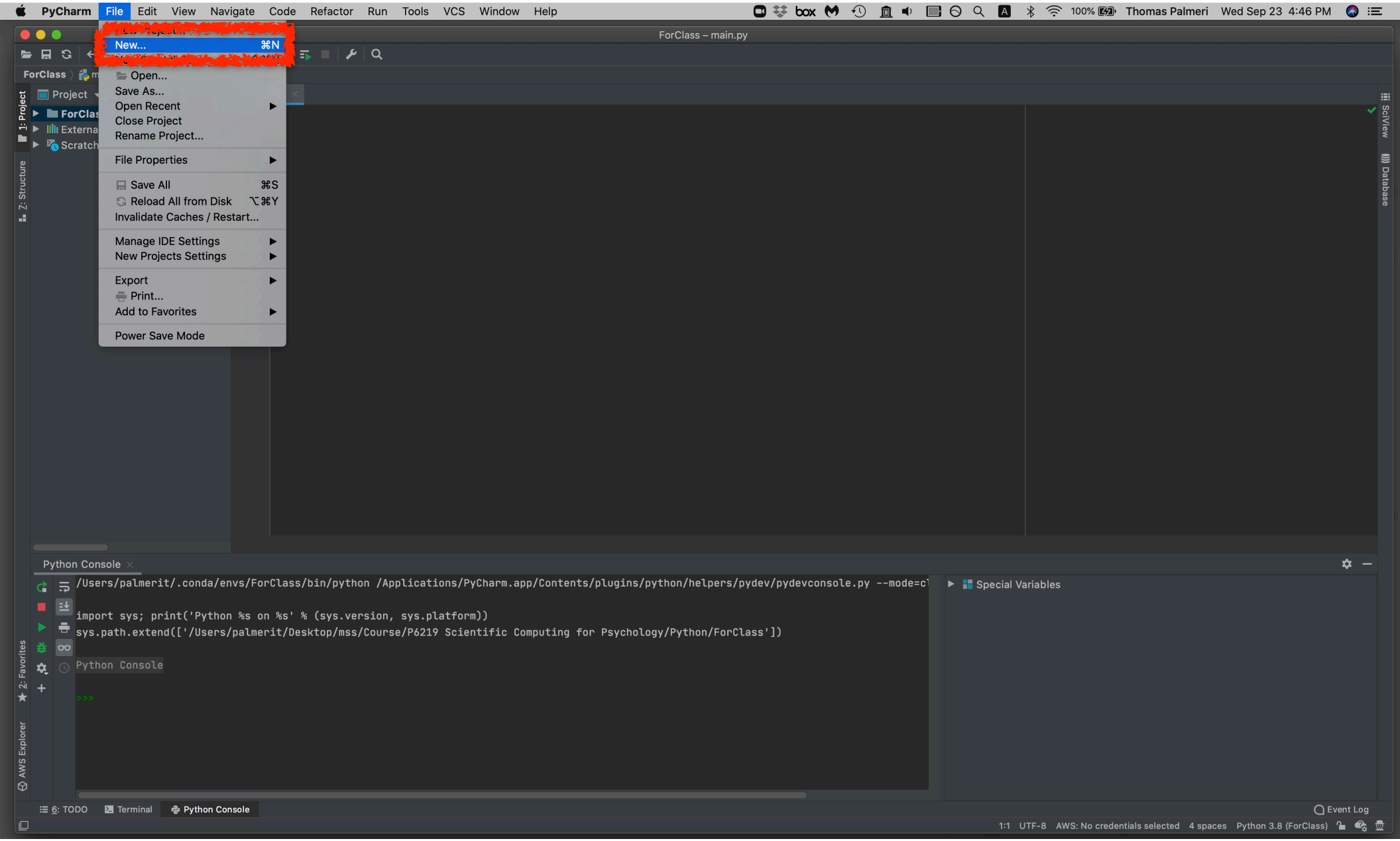
this should be the environment you created before

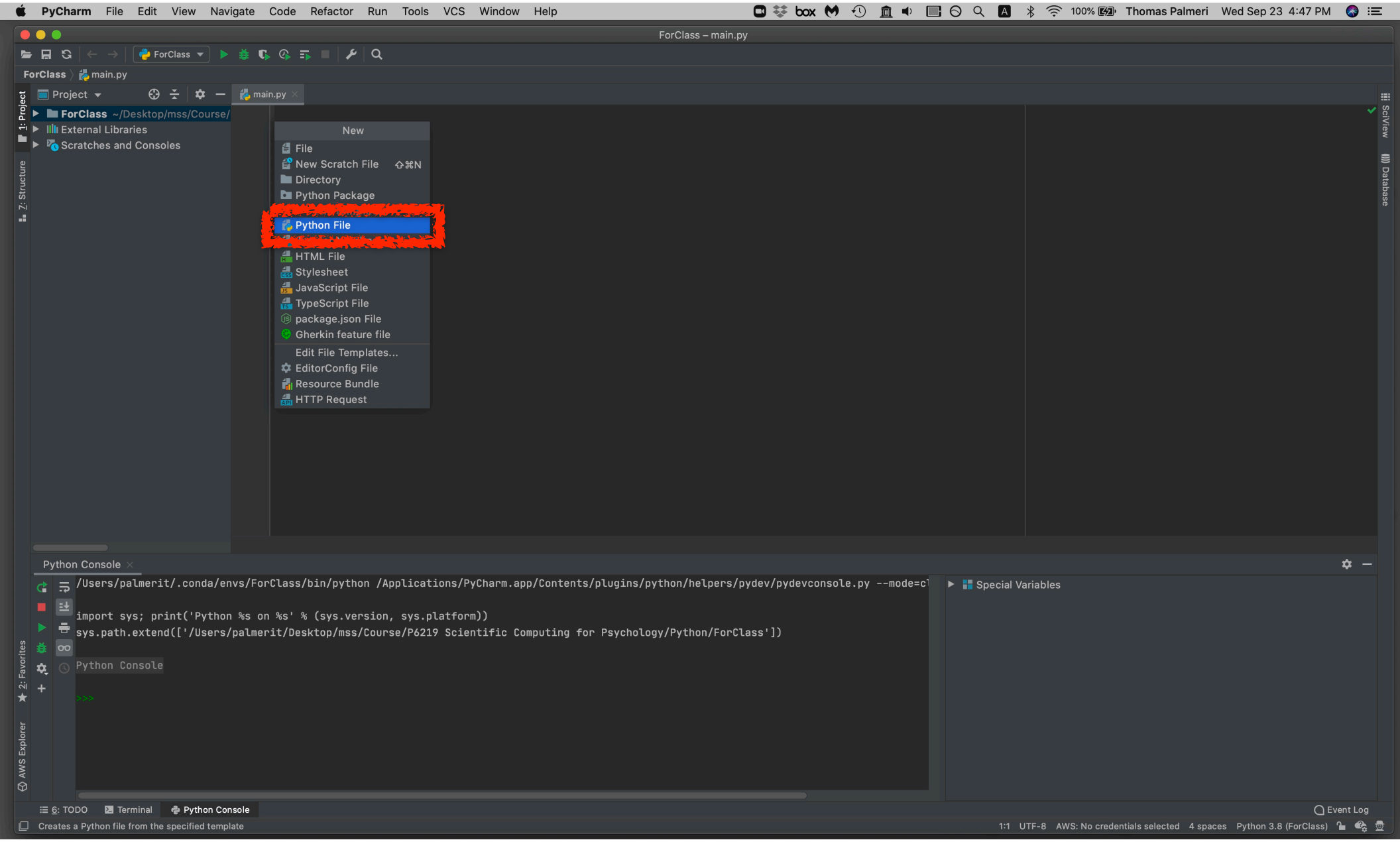

PyCharm File Edit View Navigate Code Refactor Run Tools VCS Window Help
Thomas Palmeri Tue Oct 4 10:29 PM
Fall2022 – TestPsiPhys.py
Project
Fall2022 ~/Desktop/mss/Course/PSY6219
Week1
Week7
MyPsiPhys.py
PsiPhys.py
TestPsiPhys.py
External Libraries
Scratches and Consoles
TestPsiPhys.py
#
test calling 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()
SciView: Data Plots
Run Python Console or Debugger to view available data
Documentation: Fall2022
No documentation found.
Python Console
import sys; print('Python %s on %s' % (sys.version, sys.platform))
sys.path.extend(['/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]:
Special Variables
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

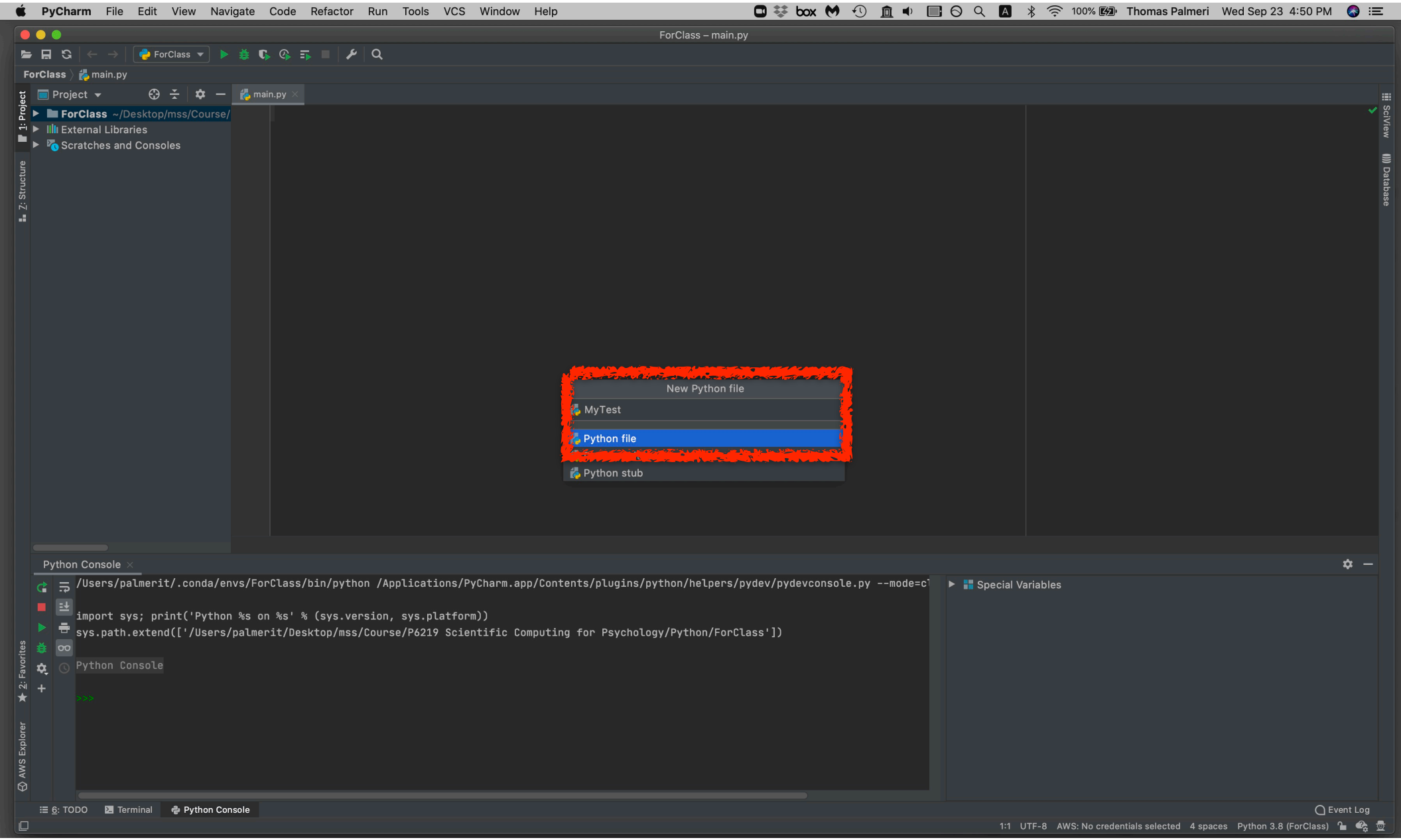


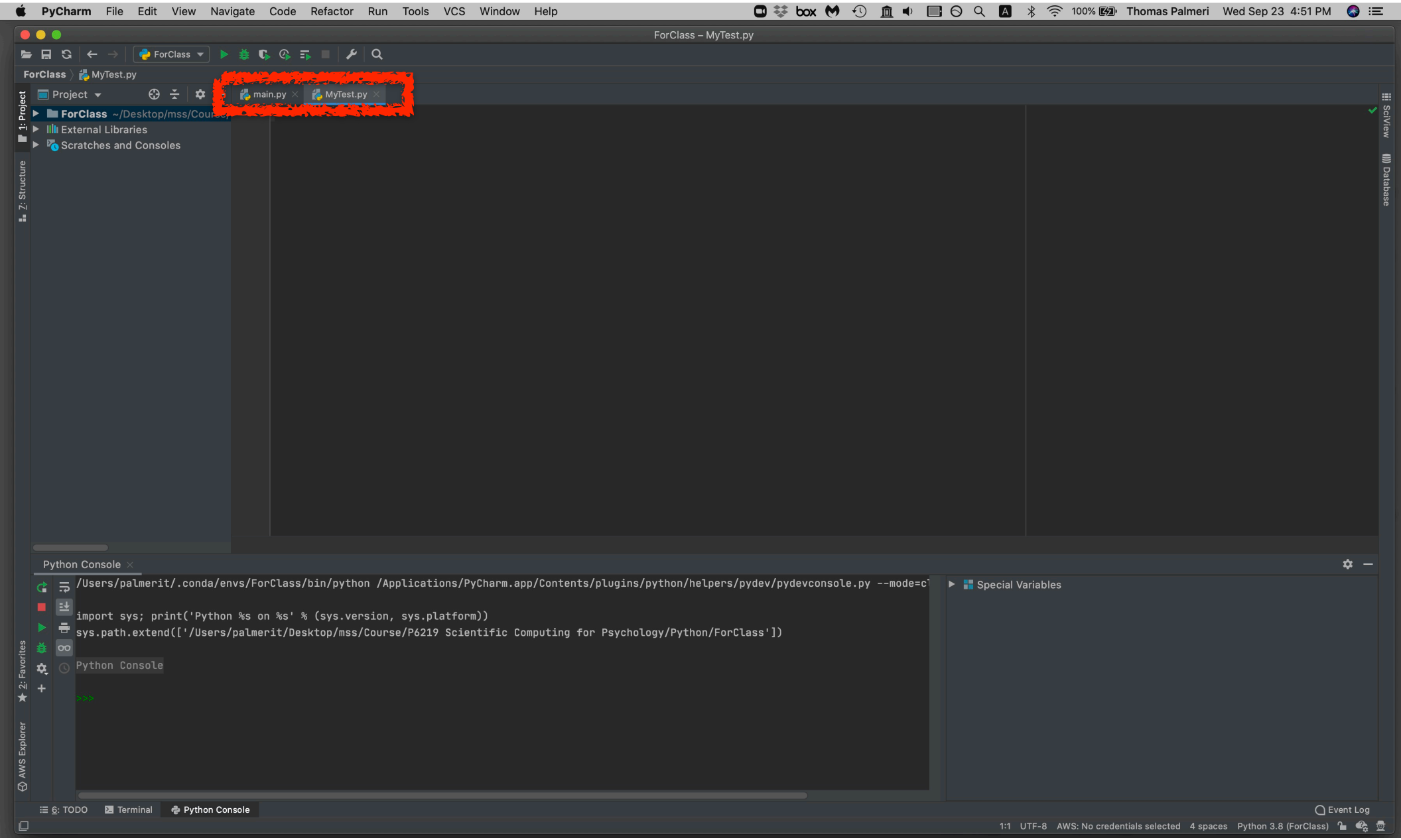
this is the folder that contains my Python files for this project

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

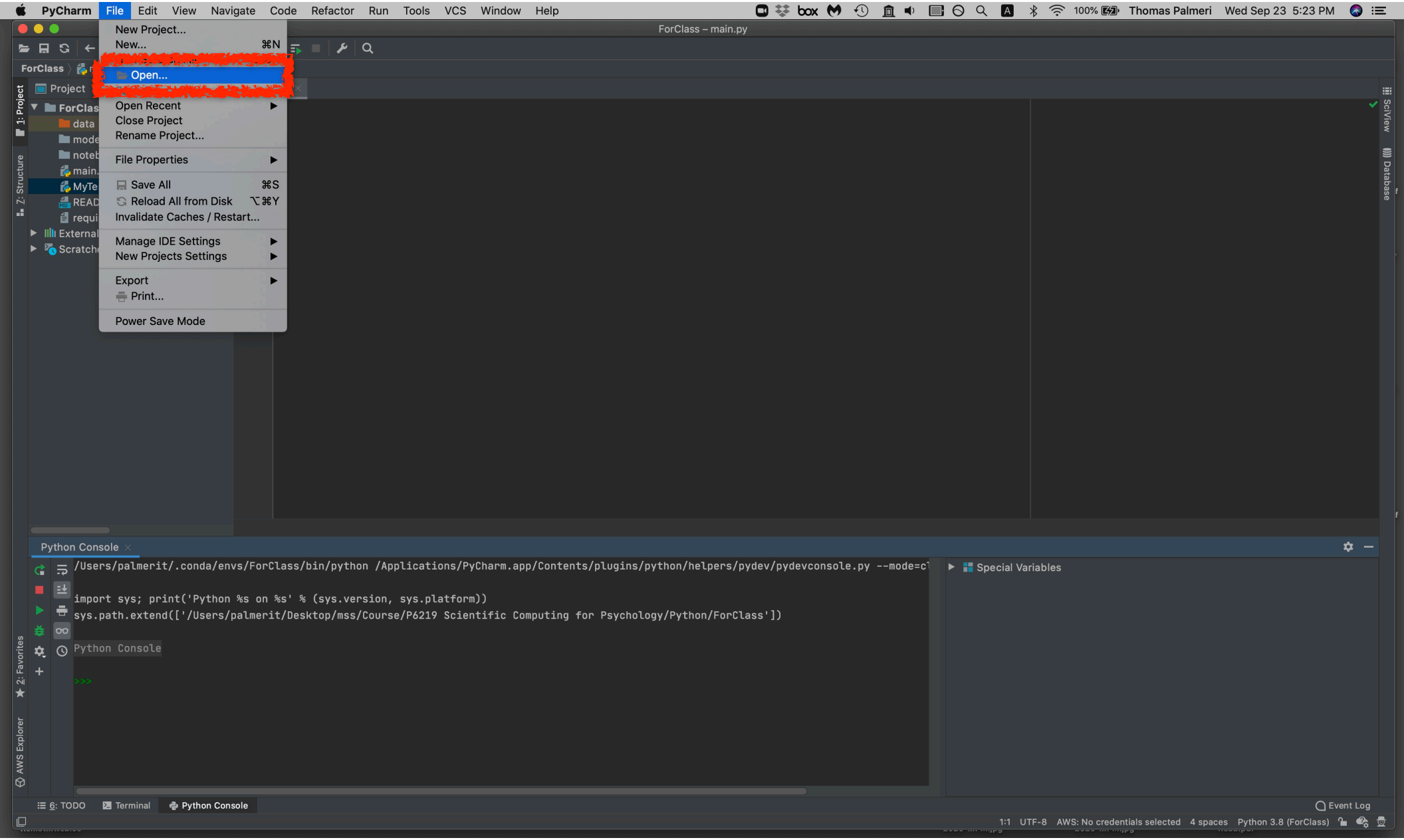

PyCharm File Edit View Navigate Code Refactor Run Tools VCS Window Help
Thomas Palmeri Wed Sep 23 4:46 PM
ForClass – main.py
New... ⌘N
Open...
Save As...
Open Recent
Close Project
Rename Project...
File Properties
Save All ⌘S
Reload All from Disk ⌥⌘Y
Invalidate Caches / Restart...
Manage IDE Settings
New Projects Settings
Export
Print...
Add to Favorites
Power Save Mode
Project
ForClass
Python Console
/Users/palmerit/.conda/envs/ForClass/bin/python /Applications/PyCharm.app/Contents/plugins/python/helpers/pydev/pydevconsole.py --mode=c
import sys; print('Python %s on %s' % (sys.version, sys.platform))
sys.path.extend(['/Users/palmerit/Desktop/mss/Course/P6219 Scientific Computing for Psychology/Python/ForClass'])
Python Console
>>>
Special Variables
6: TODO
Terminal
Python Console
Event Log
1:1 UTF-8 AWS: No credentials selected 4 spaces Python 3.8 (ForClass)

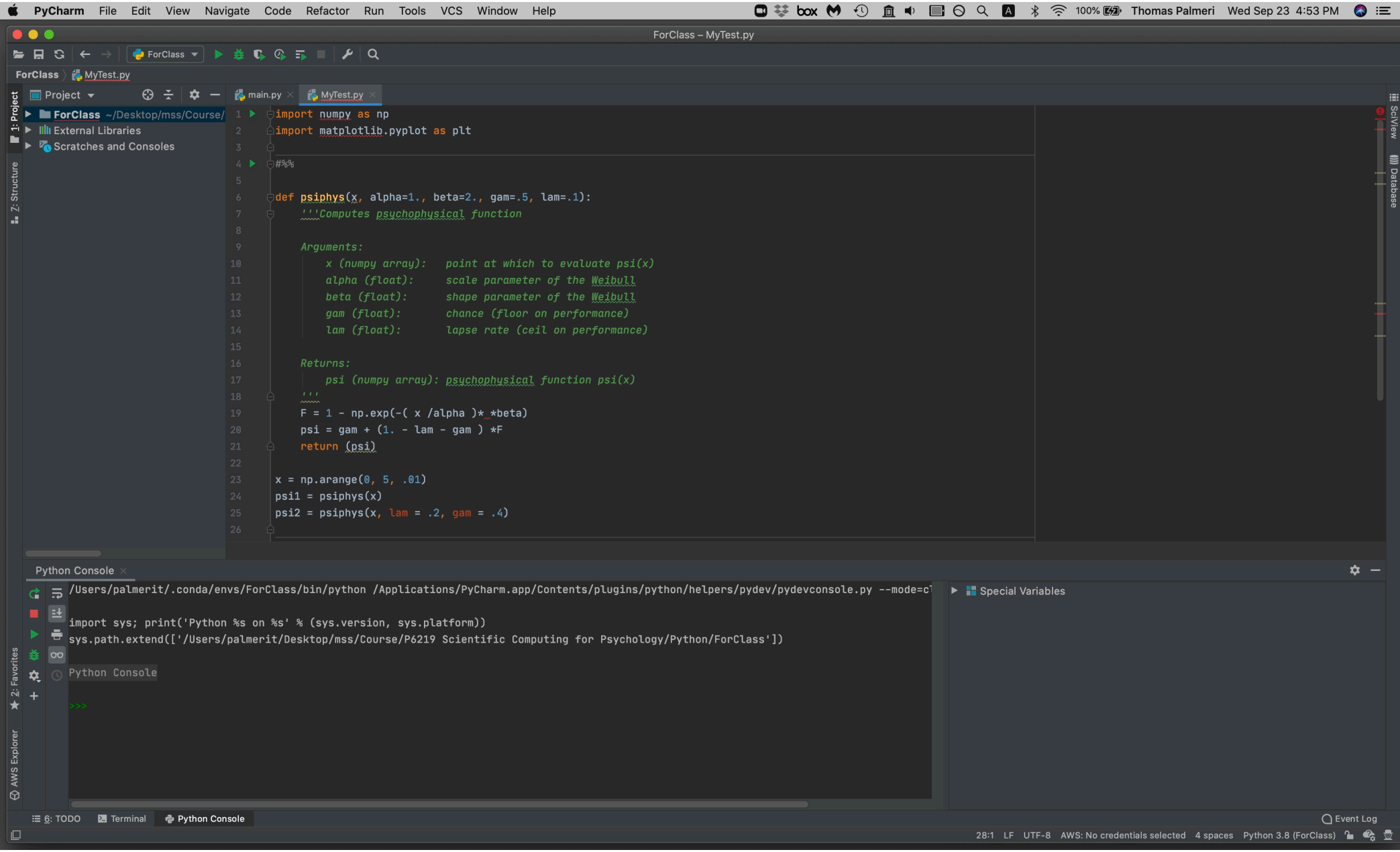

PyCharm File Edit View Navigate Code Refactor Run Tools VCS Window Help
Thomas Palmeri Wed Sep 23 4:47 PM
ForClass – main.py
ForClass main.py
Project
ForClass ~/Desktop/mss/Course/
External Libraries
Scratches and Consoles
main.py
New
File
New Scratch File ⇧⌘N
Directory
Python Package
Python File
HTML File
Stylesheet
JavaScript File
TypeScript File
package.json File
Gherkin feature file
Edit File Templates...
EditorConfig File
Resource Bundle
HTTP Request
Python Console
/Users/palmerit/.conda/envs/ForClass/bin/python /Applications/PyCharm.app/Contents/plugins/python/helpers/pydev/pydevconsole.py --mode=c
import sys; print('Python %s on %s' % (sys.version, sys.platform))
sys.path.extend(['/Users/palmerit/Desktop/mss/Course/P6219 Scientific Computing for Psychology/Python/ForClass'])
Python Console
>>>
Special Variables
6: TODO
Terminal
Python Console
Event Log
Creates a Python file from the specified template
1:1 UTF-8 AWS: No credentials selected 4 spaces Python 3.8 (ForClass)


PyCharm File Edit View Navigate Code Refactor Run Tools VCS Window Help
Thomas Palmeri Wed Sep 23 4:50 PM
ForClass – main.py
ForClass
main.py
Project
ForClass ~/Desktop/mss/Course/
External Libraries
Scratches and Consoles
New Python file
MyTest
Python file
Python stub
Python Console
/Users/palmerit/.conda/envs/ForClass/bin/python /Applications/PyCharm.app/Contents/plugins/python/helpers/pydev/pydevconsole.py --mode=c
import sys; print('Python %s on %s' % (sys.version, sys.platform))
sys.path.extend(['/Users/palmerit/Desktop/mss/Course/P6219 Scientific Computing for Psychology/Python/ForClass'])
Python Console
>>>
Special Variables
6: TODO
Terminal
Python Console
Event Log
1:1 UTF-8 AWS: No credentials selected 4 spaces Python 3.8 (ForClass)

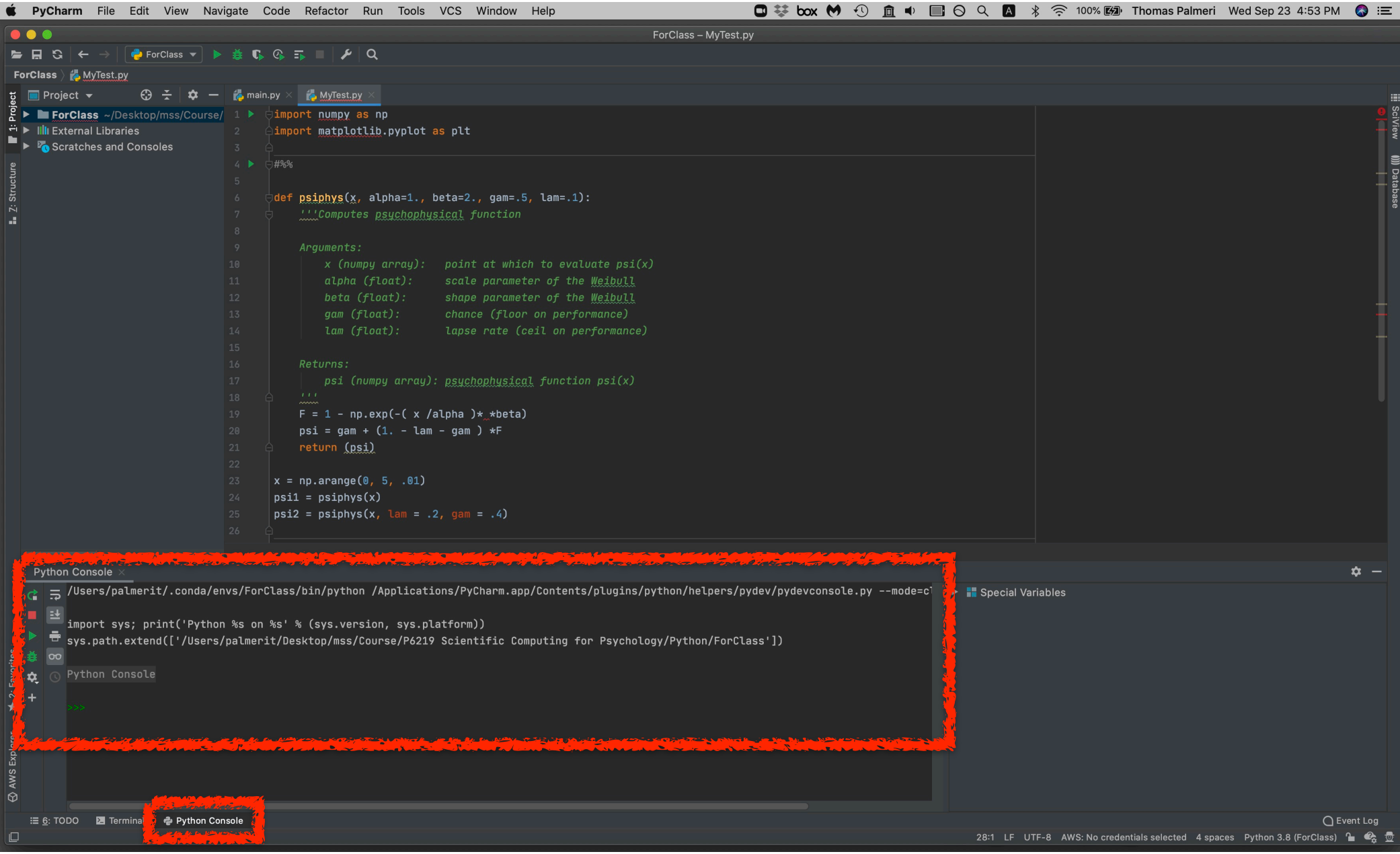

PyCharm File Edit View Navigate Code Refactor Run Tools VCS Window Help
Thomas Palmeri Wed Sep 23 4:51 PM
ForClass – MyTest.py
ForClass
MyTest.py
main.py
MyTest.py
Project
ForClass ~/Desktop/mss/Cour
External Libraries
Scratches and Consoles
1: Project
7: Structure
SciView
Database
Python Console
/Users/palmerit/.conda/envs/ForClass/bin/python /Applications/PyCharm.app/Contents/plugins/python/helpers/pydev/pydevconsole.py --mode=c
import sys; print('Python %s on %s' % (sys.version, sys.platform))
sys.path.extend(['/Users/palmerit/Desktop/mss/Course/P6219 Scientific Computing for Psychology/Python/ForClass'])
Python Console
>>>
Special Variables
2: Favorites
AWS Explorer
6: TODO
Terminal
Python Console
Event Log
1:1 UTF-8 AWS: No credentials selected 4 spaces Python 3.8 (ForClass)

make sure you the file is in the project folder

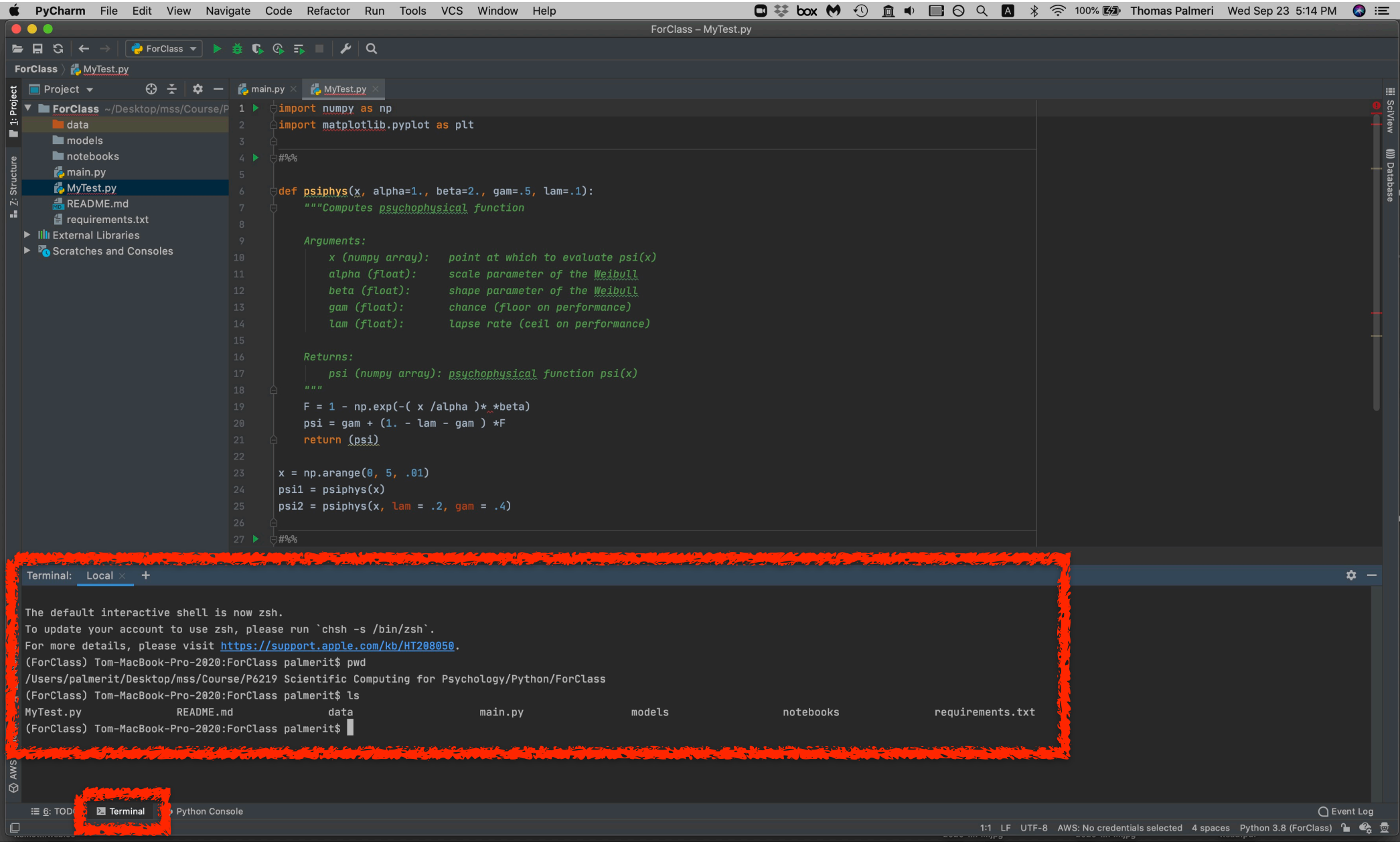




Python Console : place to try out code independent of the program you are writing

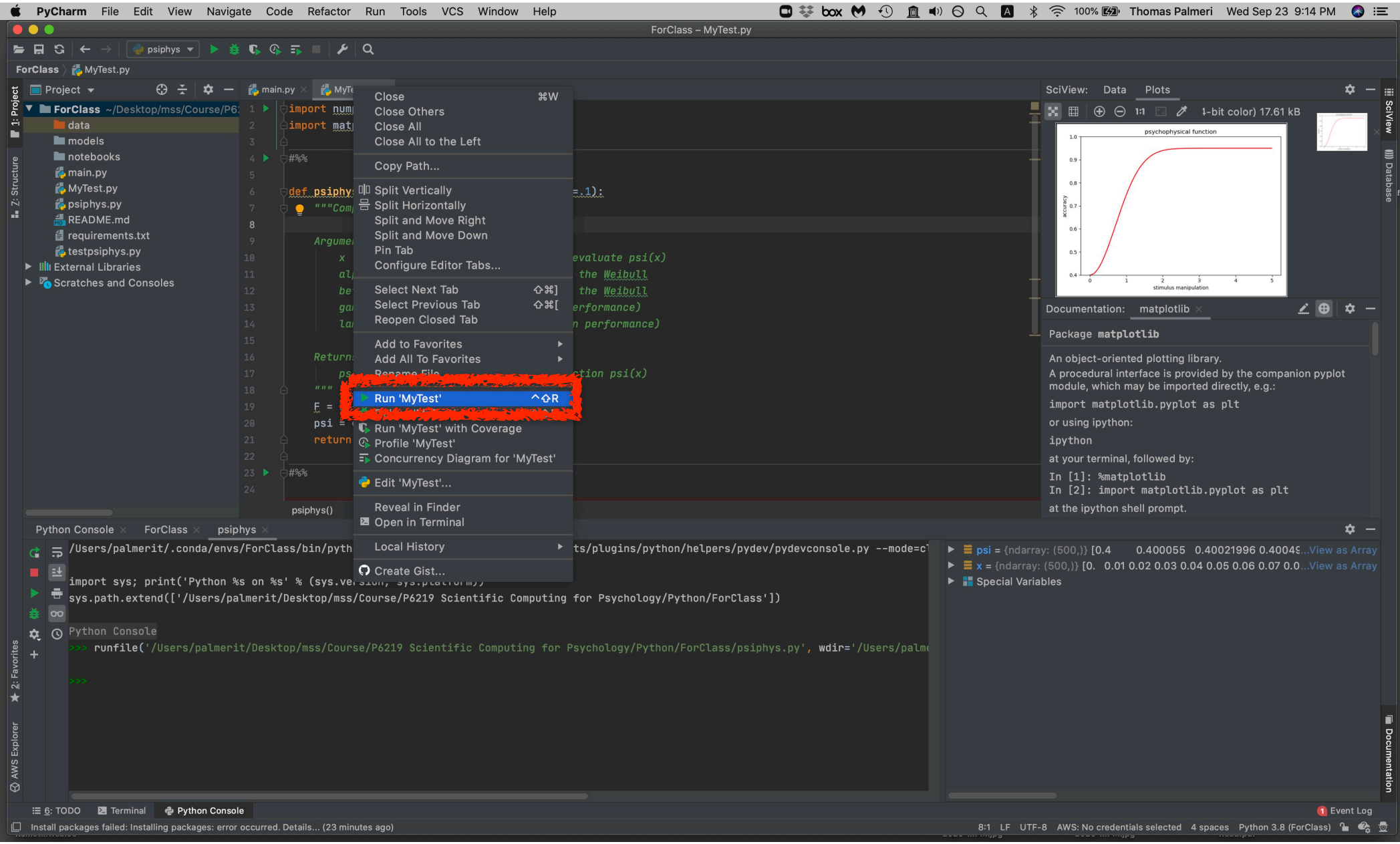


Python Console : place to try out code independent of the program you are writing

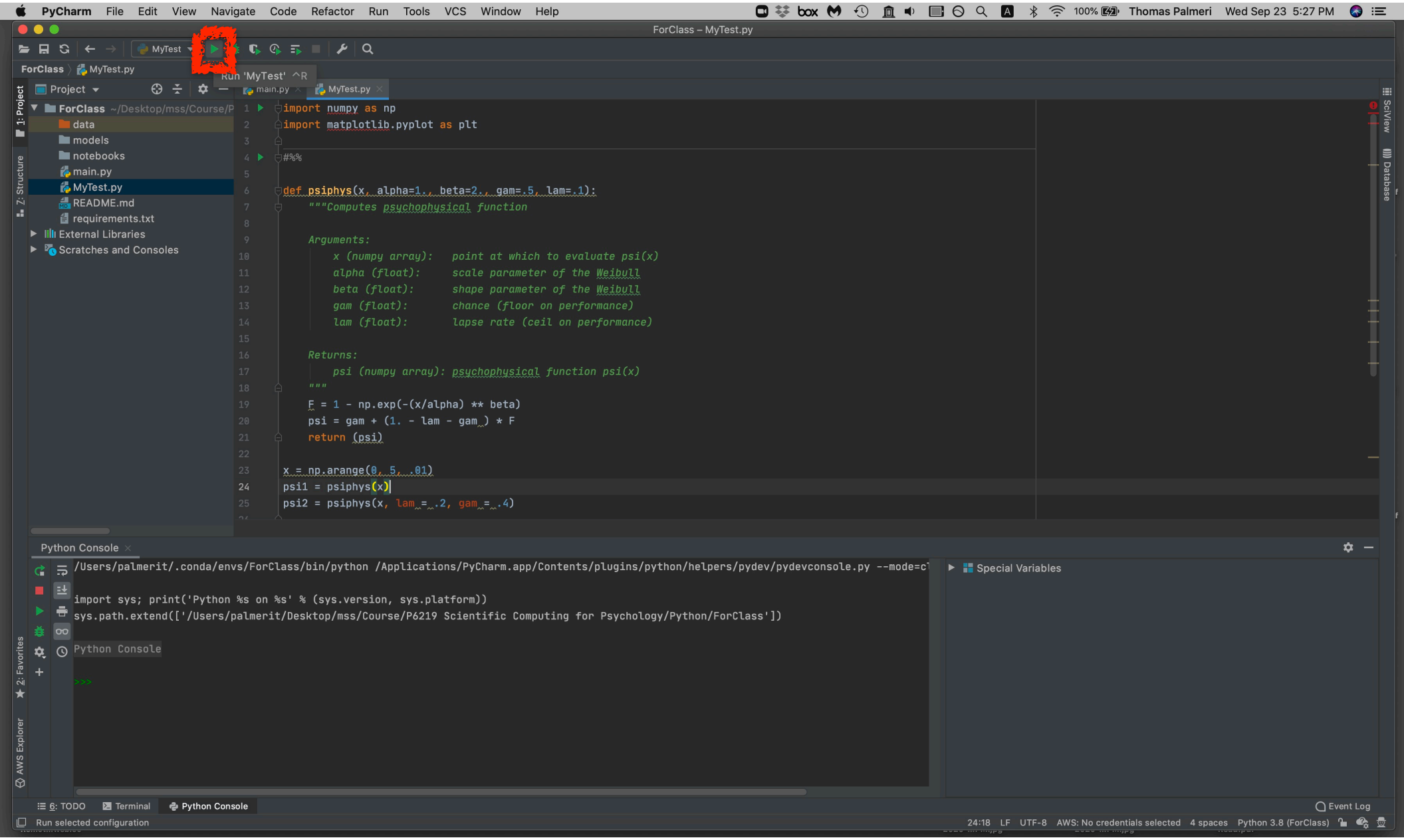


Terminal : work in Linux or Windows (enter in current folder)

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

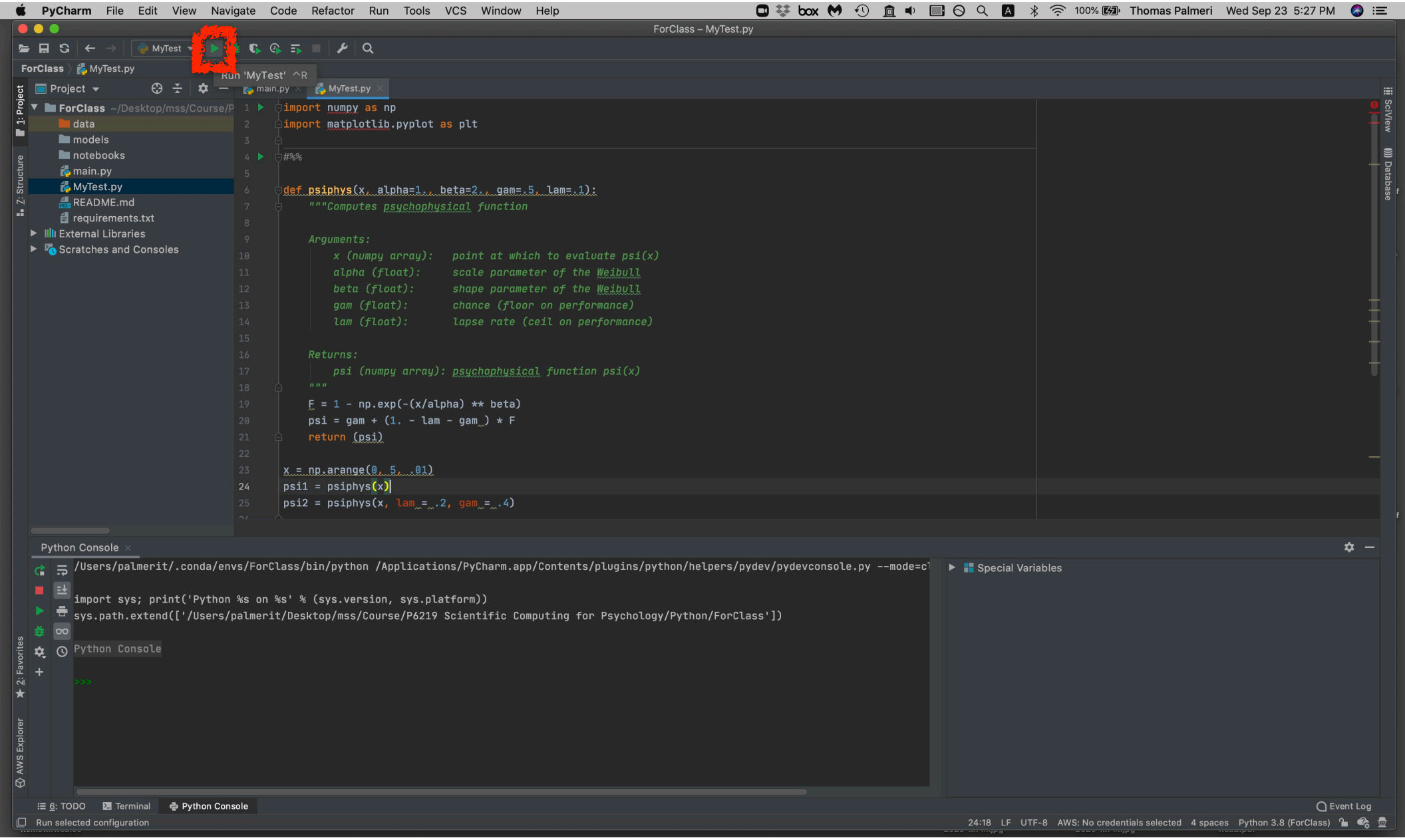


green arrow to run `MyTest.py`



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 — Thomas Palmeri Wed Sep 23 5:50 PM

ForClass – MyTest.py

MyTest

ForClass 〉 MyTest.py

Project

ForClass ~/Desktop/mss/Course/P6

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

External Libraries

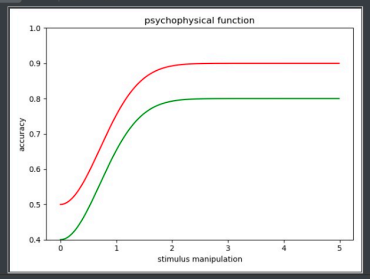
Scratches and Consoles

main.py | 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 #%%
24
25 x = np.arange(0. 5. .01)
```

SciView: Data Plots

4-bit color) 21.5 kB



Documentation: ForClass

No documentation found.

Python Console | MyTest

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

import sys; print('Python %s on %s' % (sys.version, sys.platform))
sys.path.extend(['/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/palmer

>>>
```

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

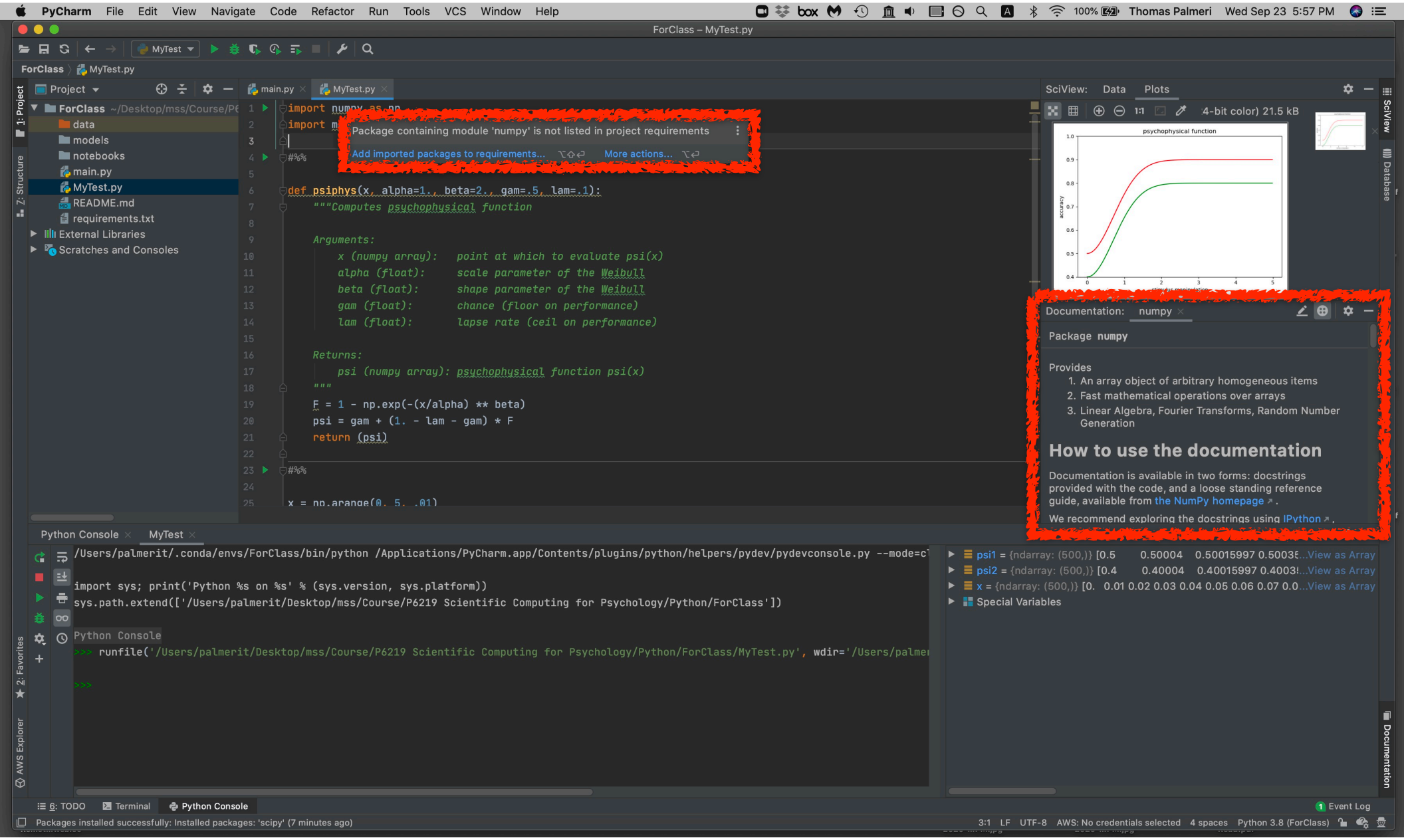
x = {ndarray: (500,)} [0. 0.01 0.02 0.03 0.04 0.05 0.06 0.07 0.0...View as Array

Special Variables

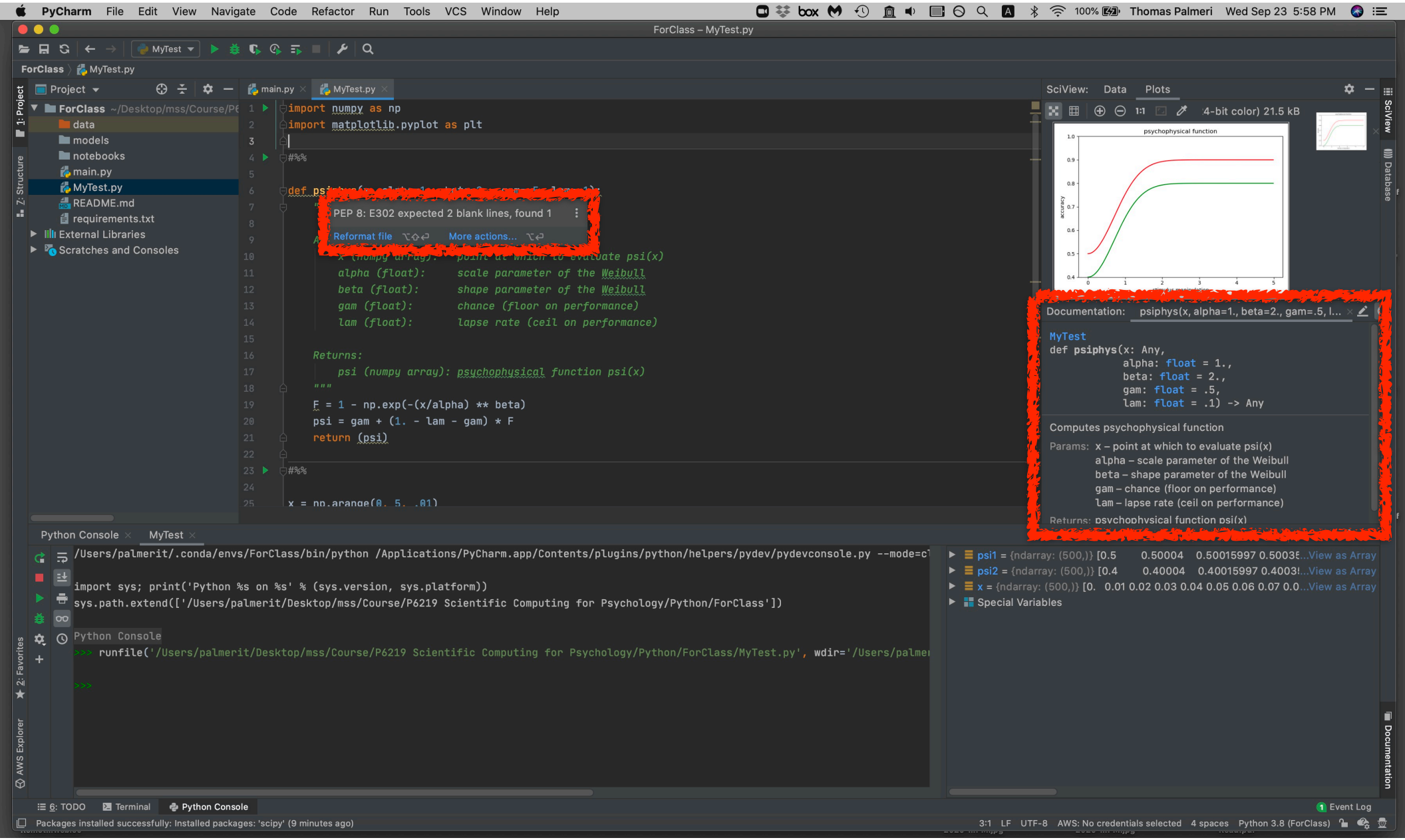
6: TODO | Terminal | Python Console | 1 Event Log

Packages installed successfully: Installed packages: 'scipy' (a minute ago) — Updating skeletons... — 24:1 LF UTF-8 AWS: No credentials selected 4 spaces Python 3.8 (ForClass)

hovering mouse reveals errors, warning, and opens documentation



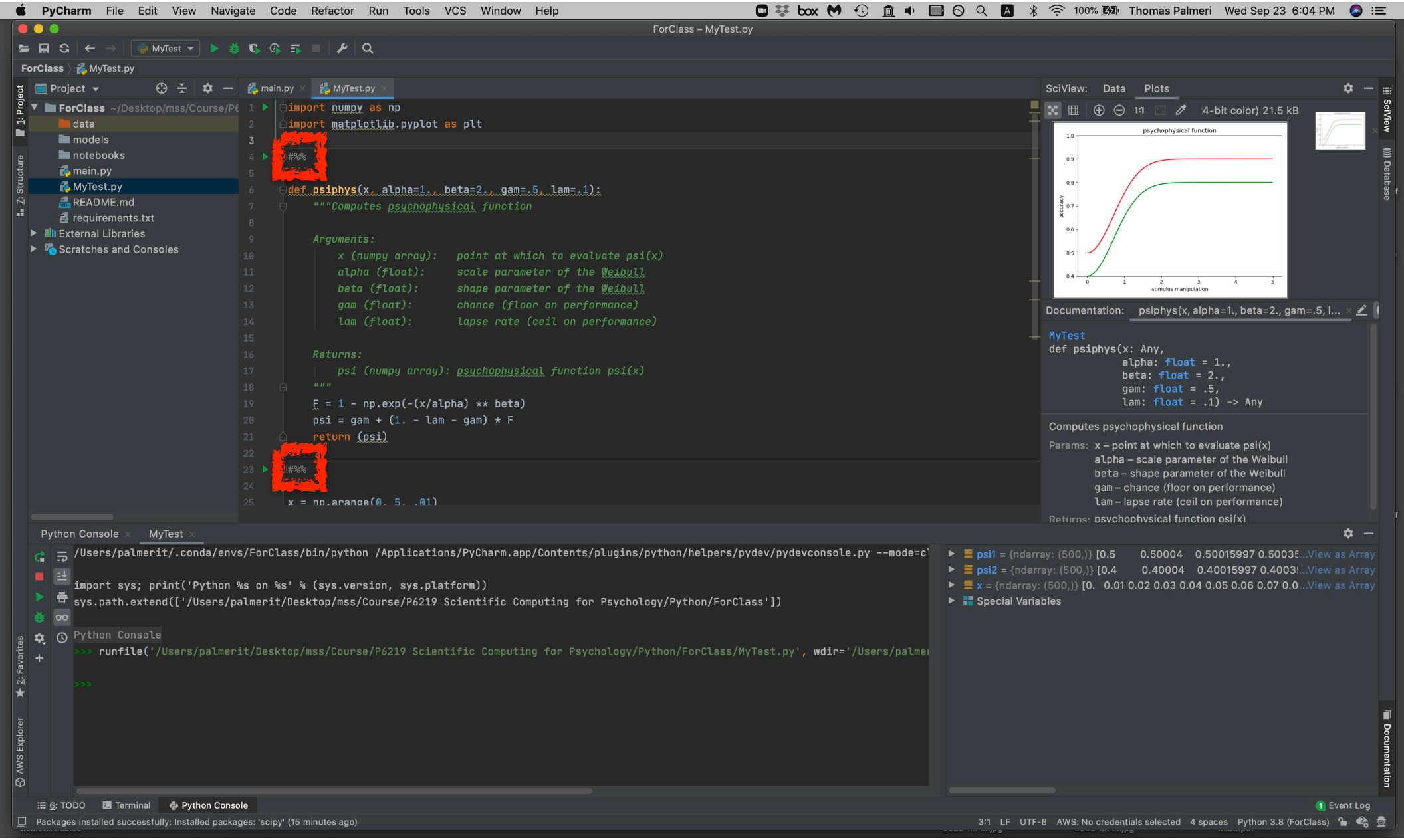
including documentation we created ourselves (for our functions)



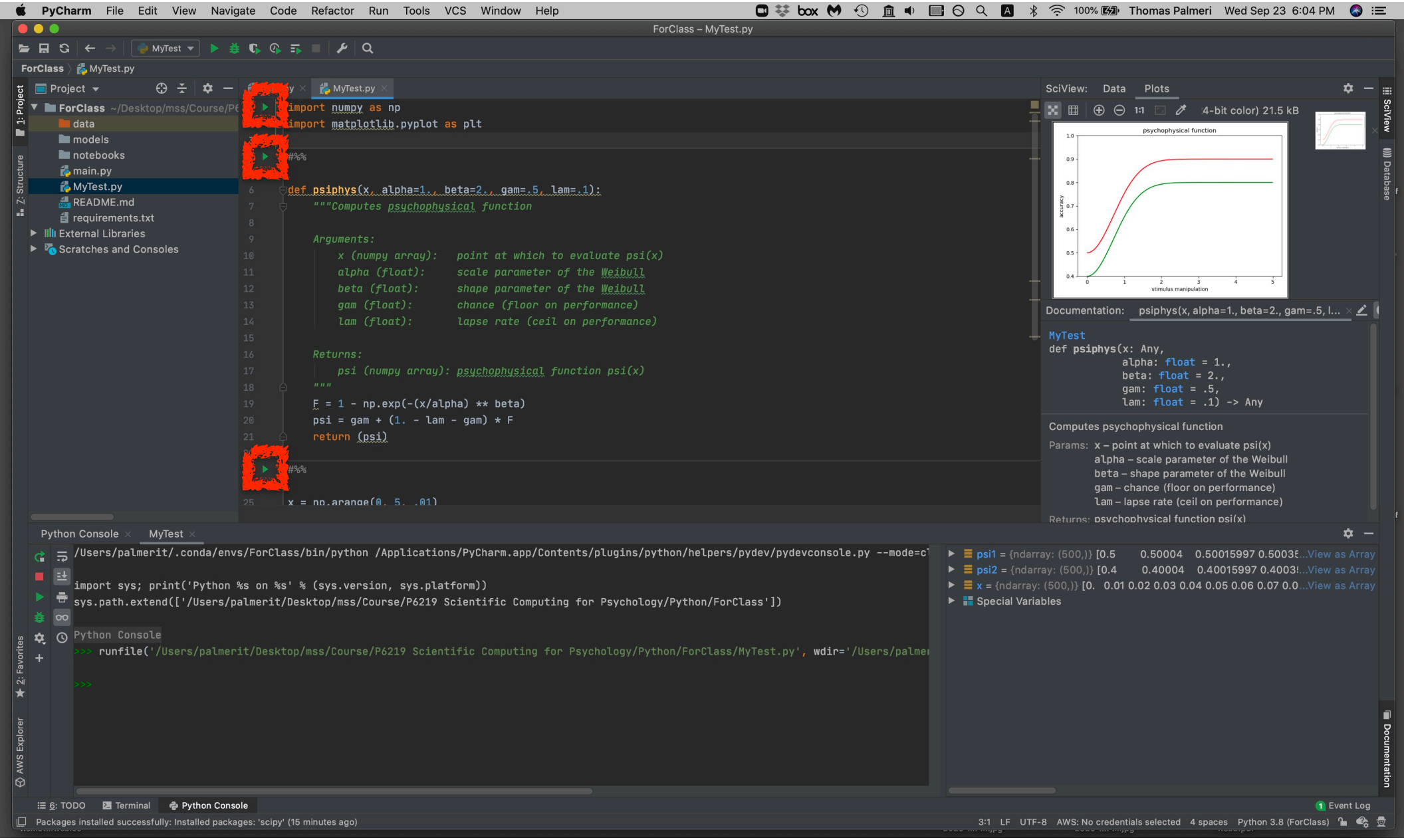
some of the warnings are PEP style guidelines (nothing more)

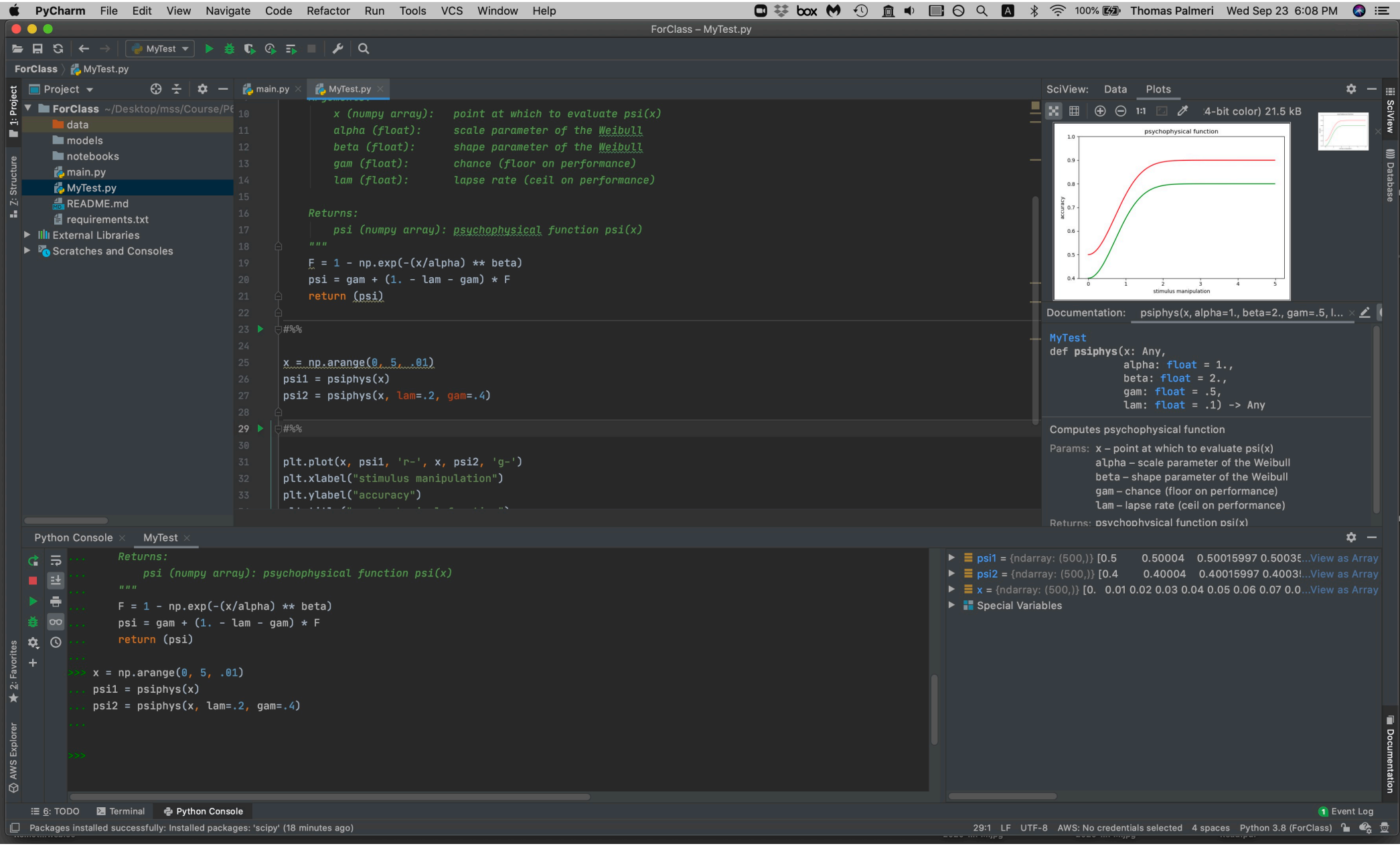
(in scientific mode)

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

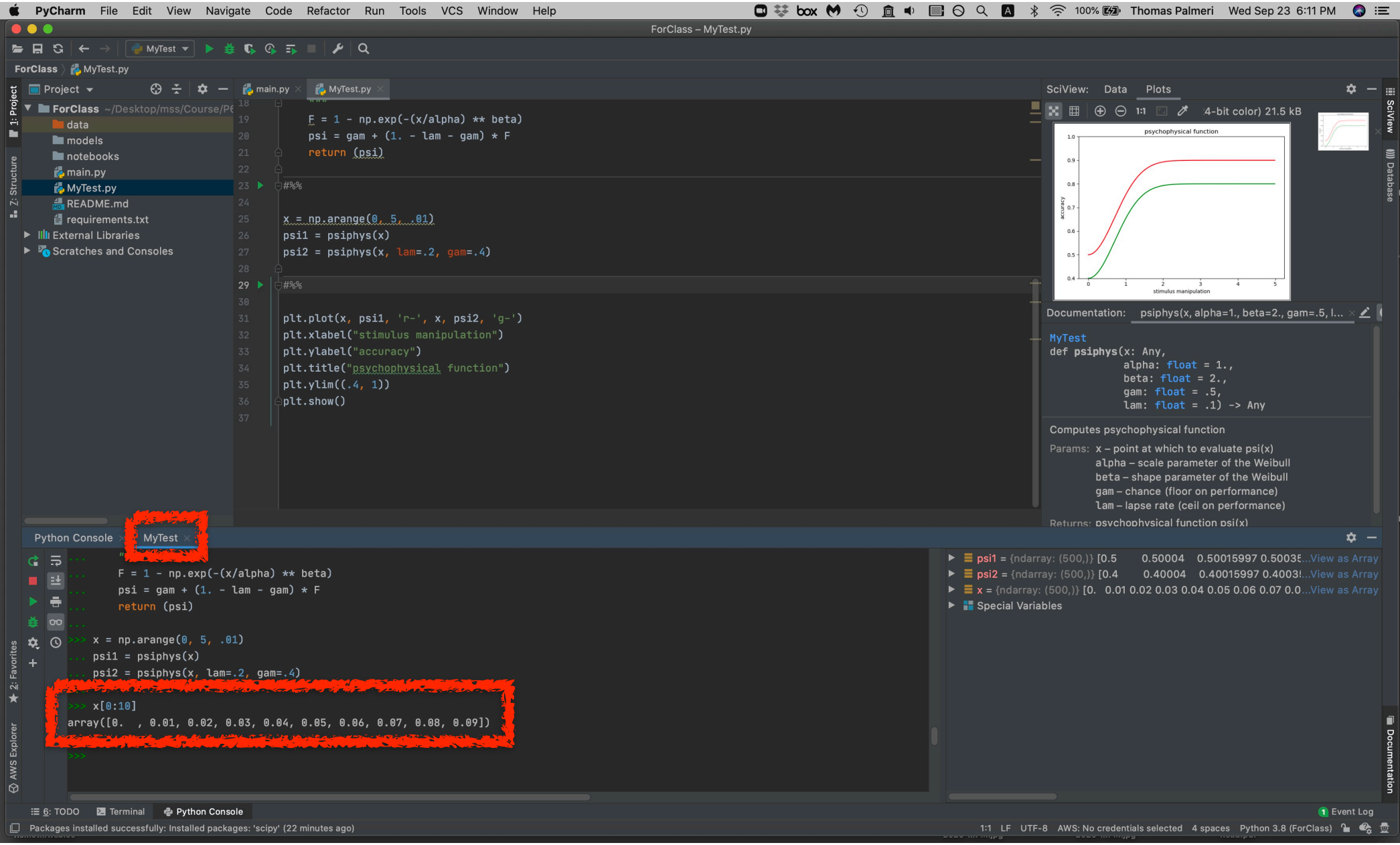


press green arrow to run the code cell (and only that code cell)

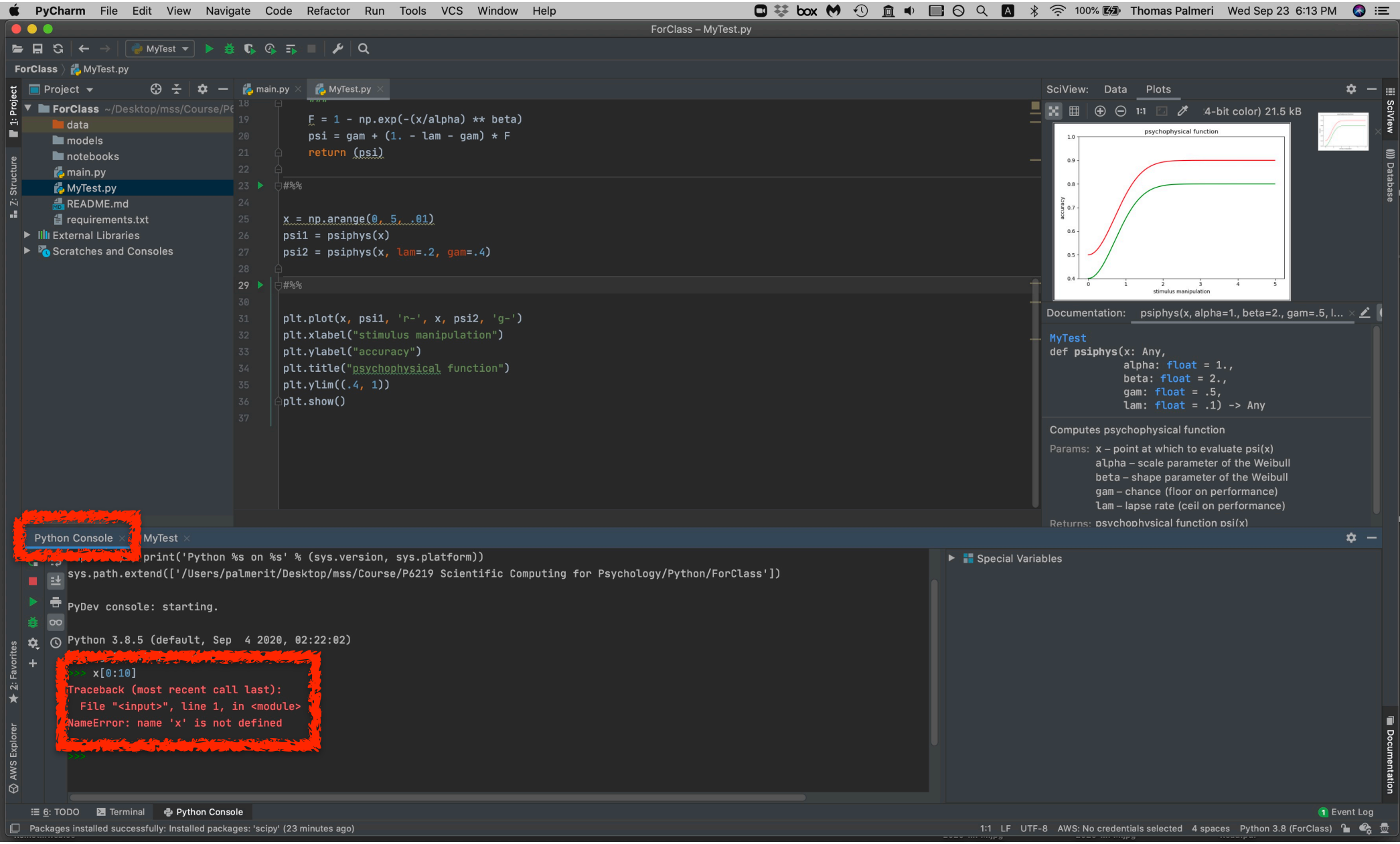




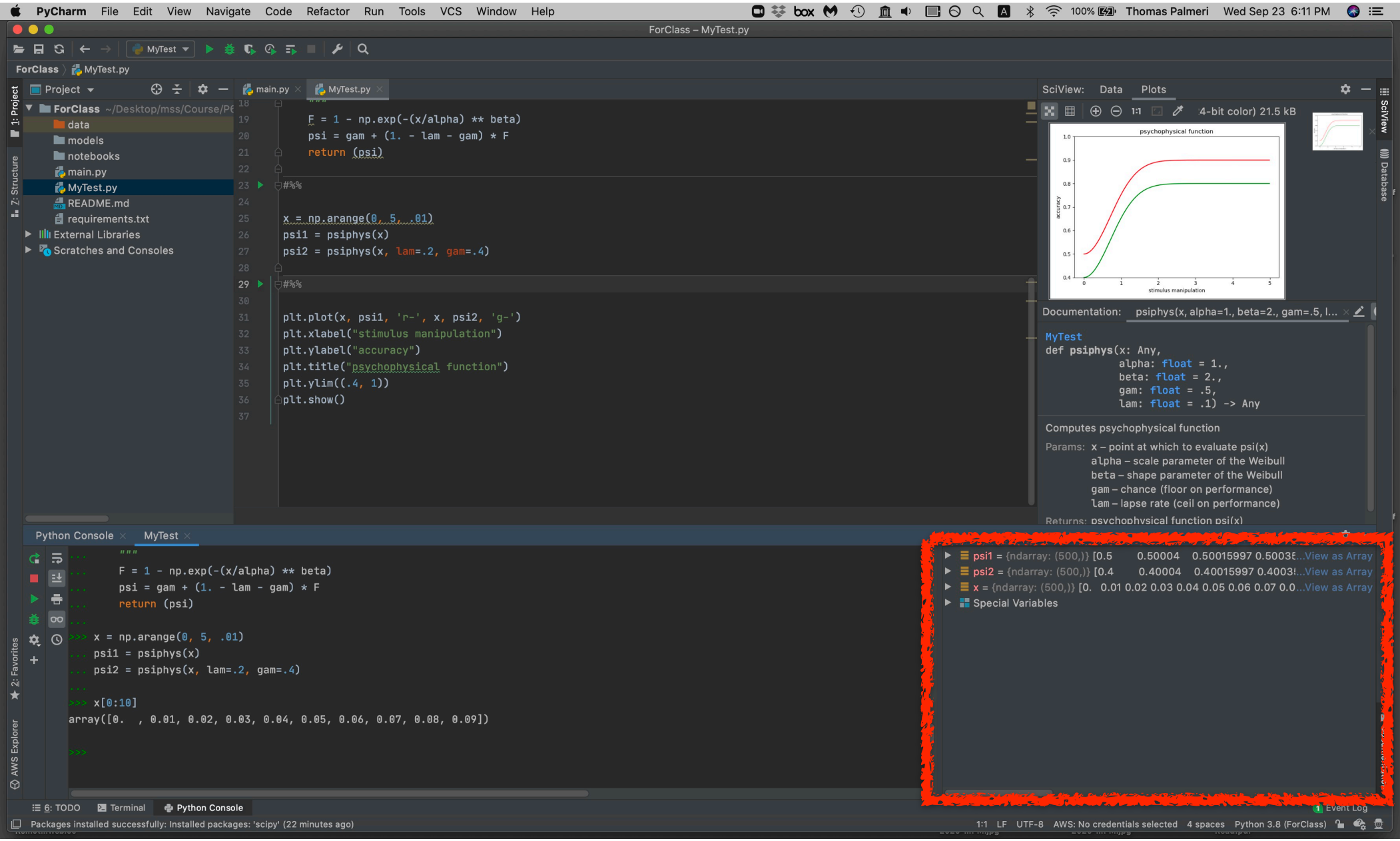
see the code from that cell executed in the `MyTest` Console



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

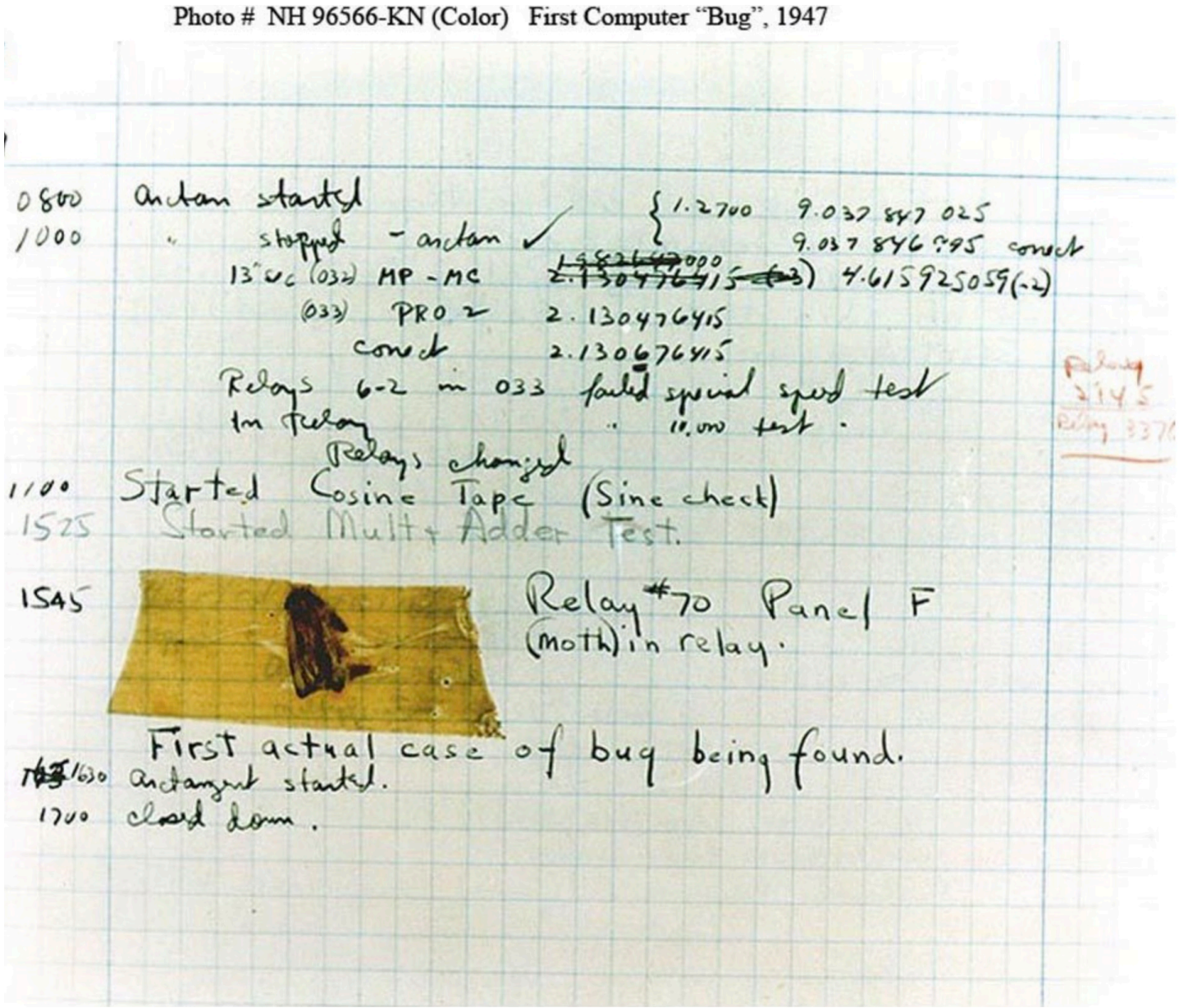


the Python Console is independent (doesn't see variables running in `MyTest`)



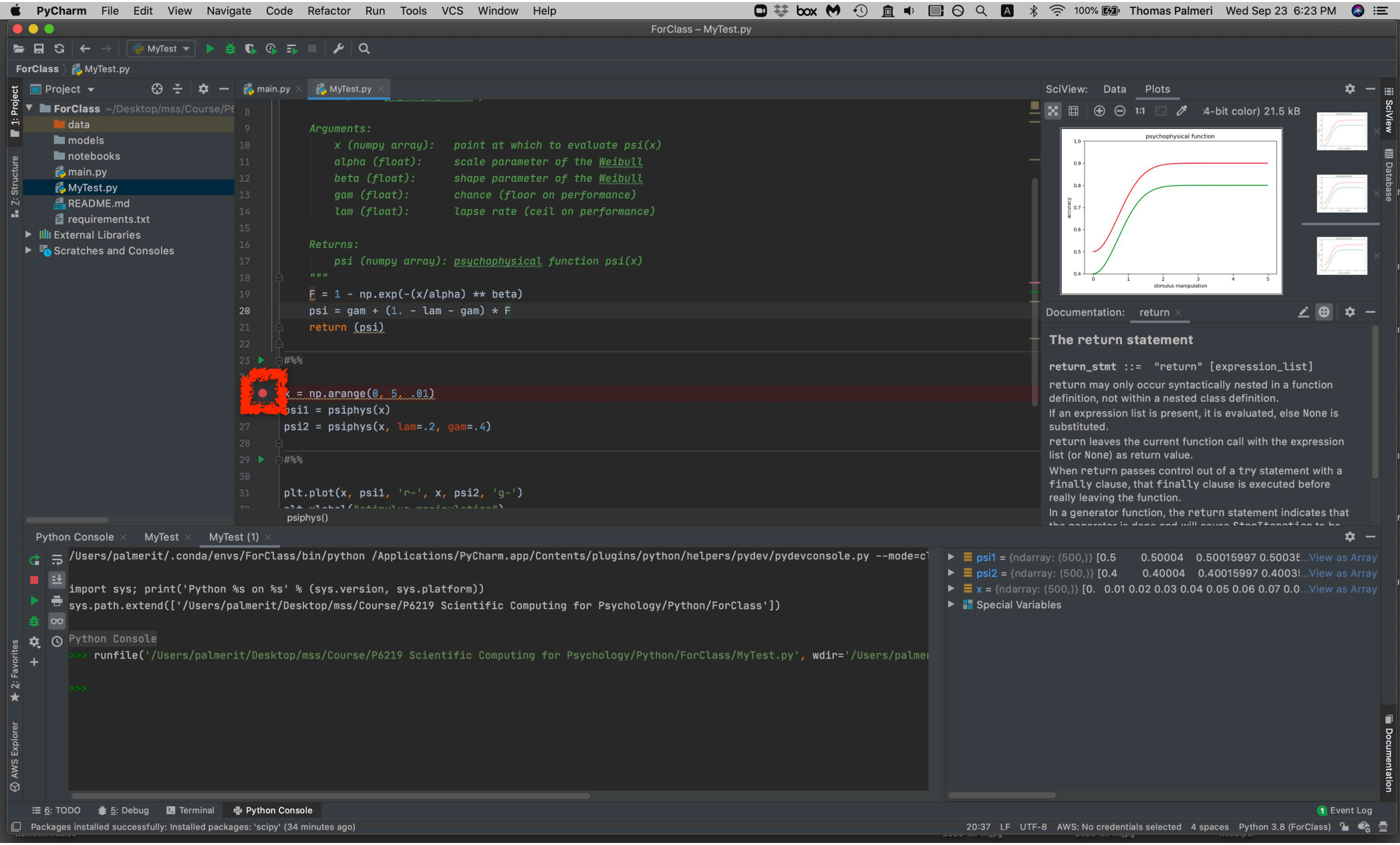
also notice that variables (and their values and types) are shown here

debugging in PyCharm

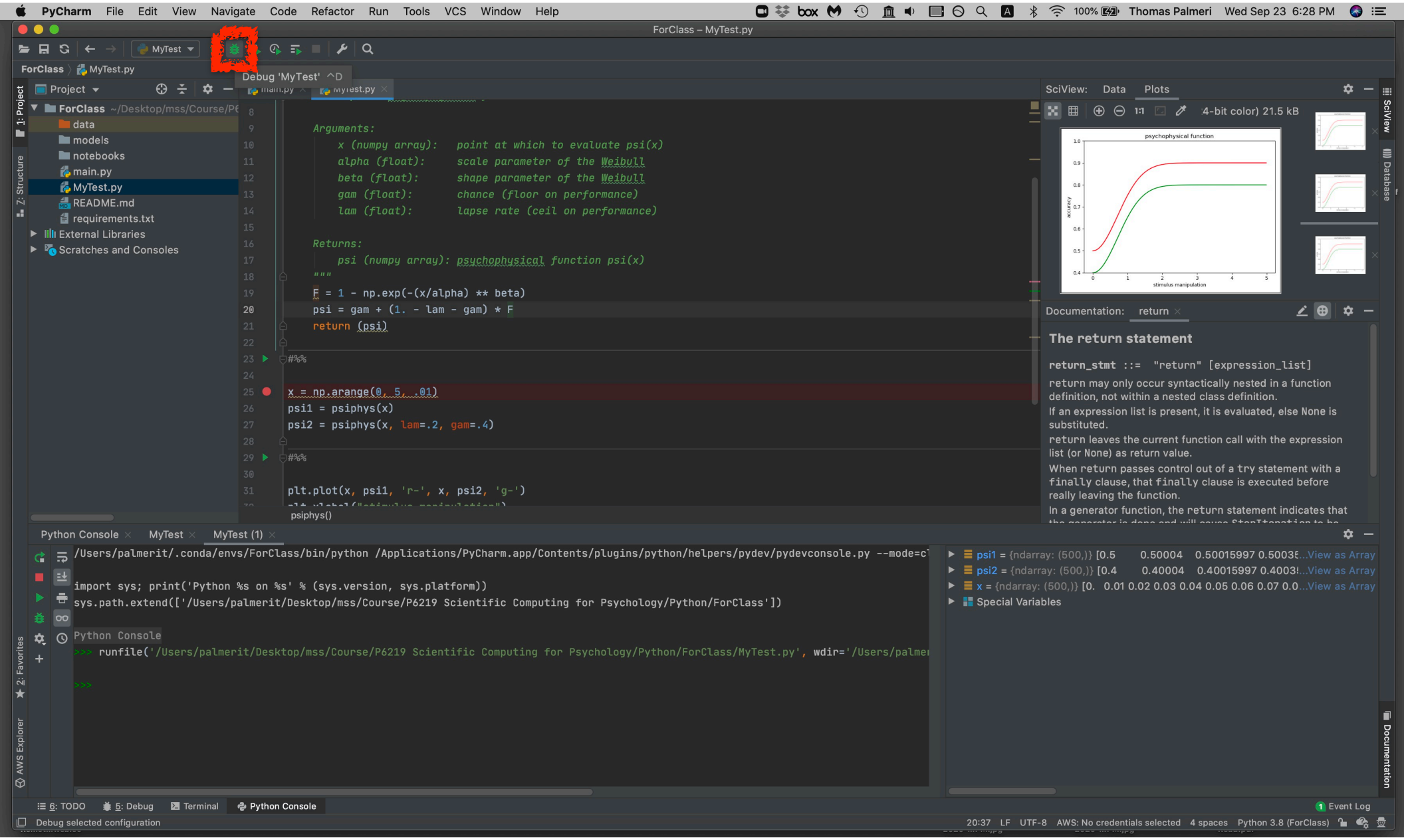


Edison actually coined the term bug (in a system) before computers were around

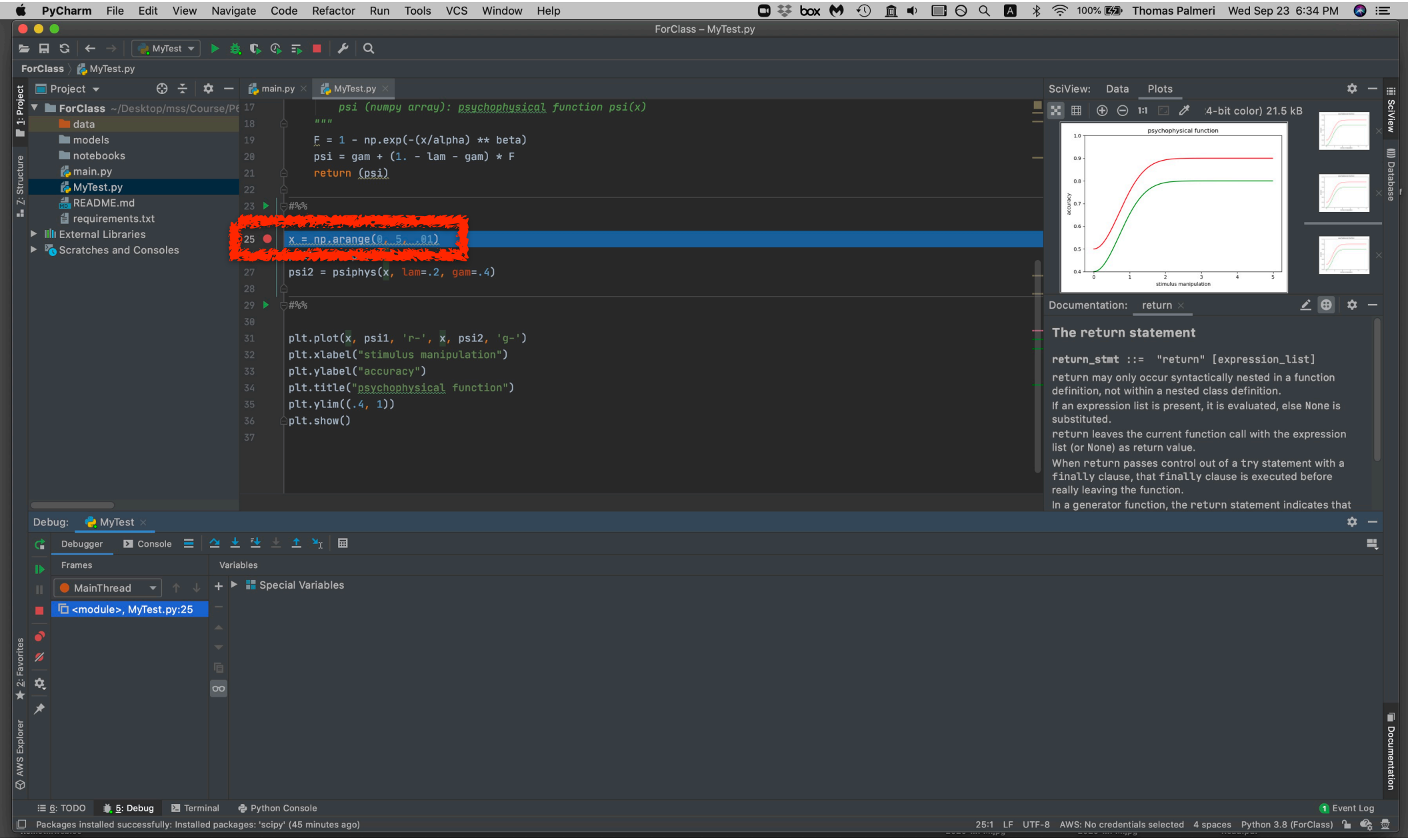
click in the "gutter" (in same column as green arrows) to set a **breakpoint** (can have many)

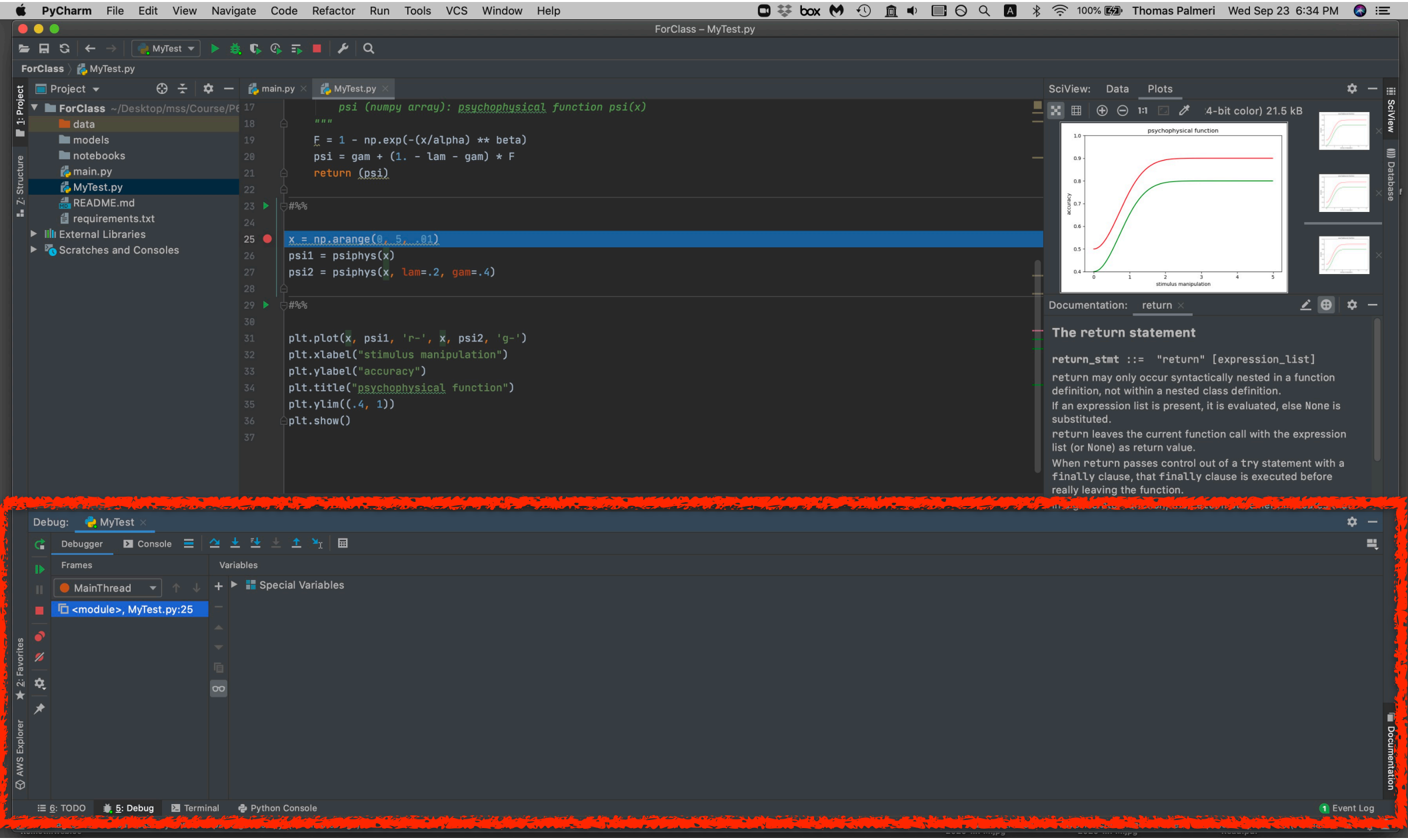


click on that to debug (it's supposed to look like a bug, but with my old eyes looks like a turtle)

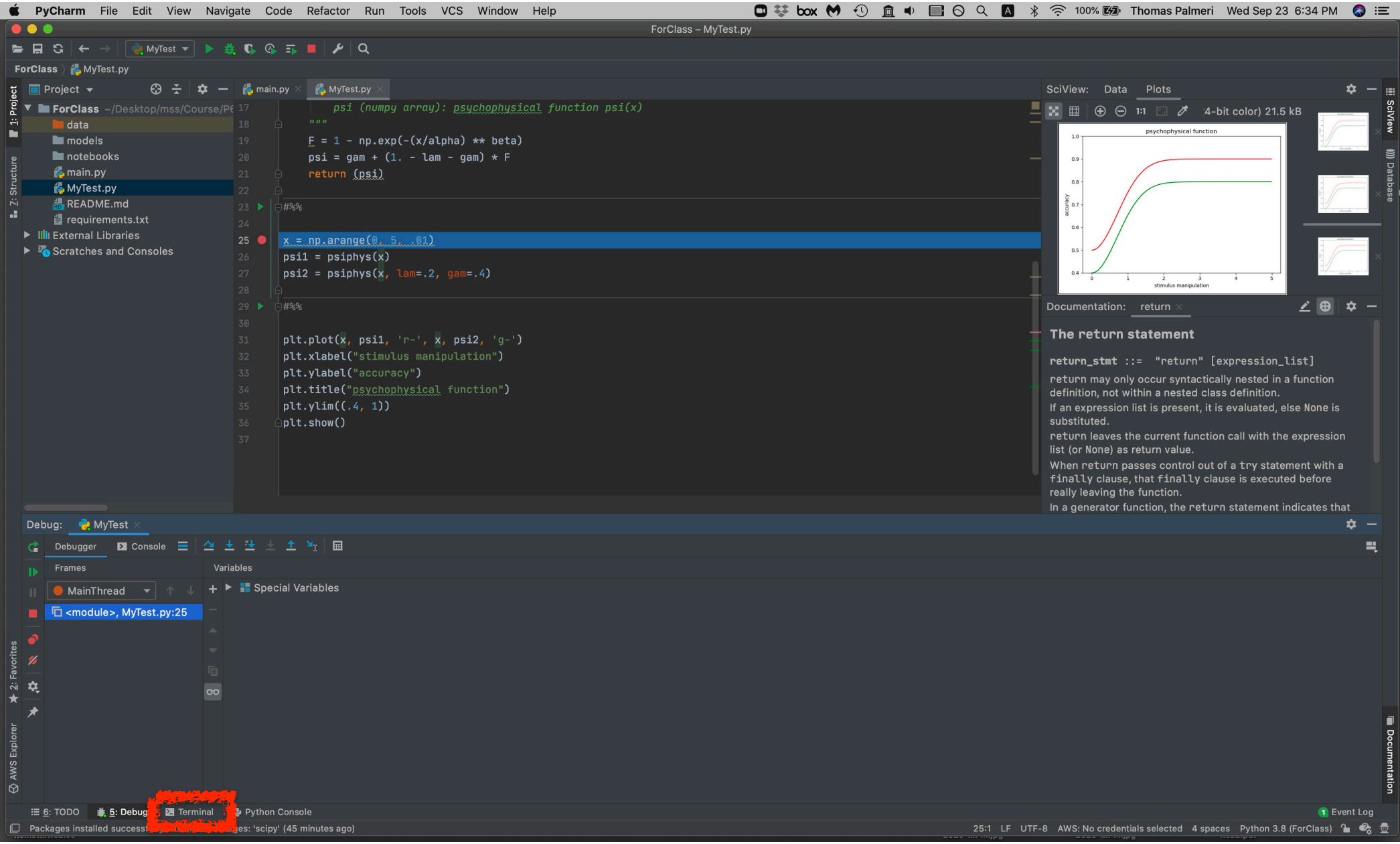


code suspended at that line, but has not executed that line yet

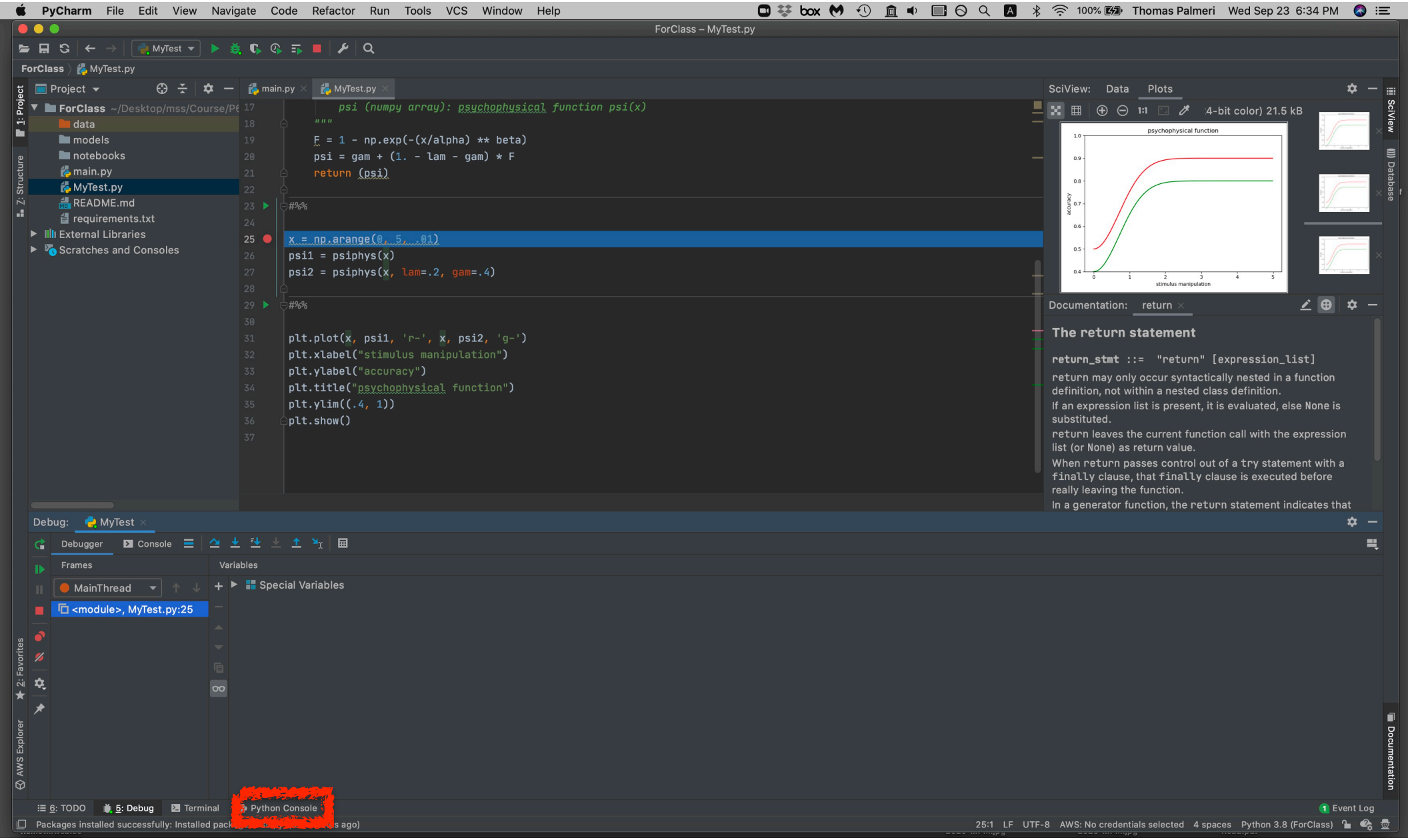




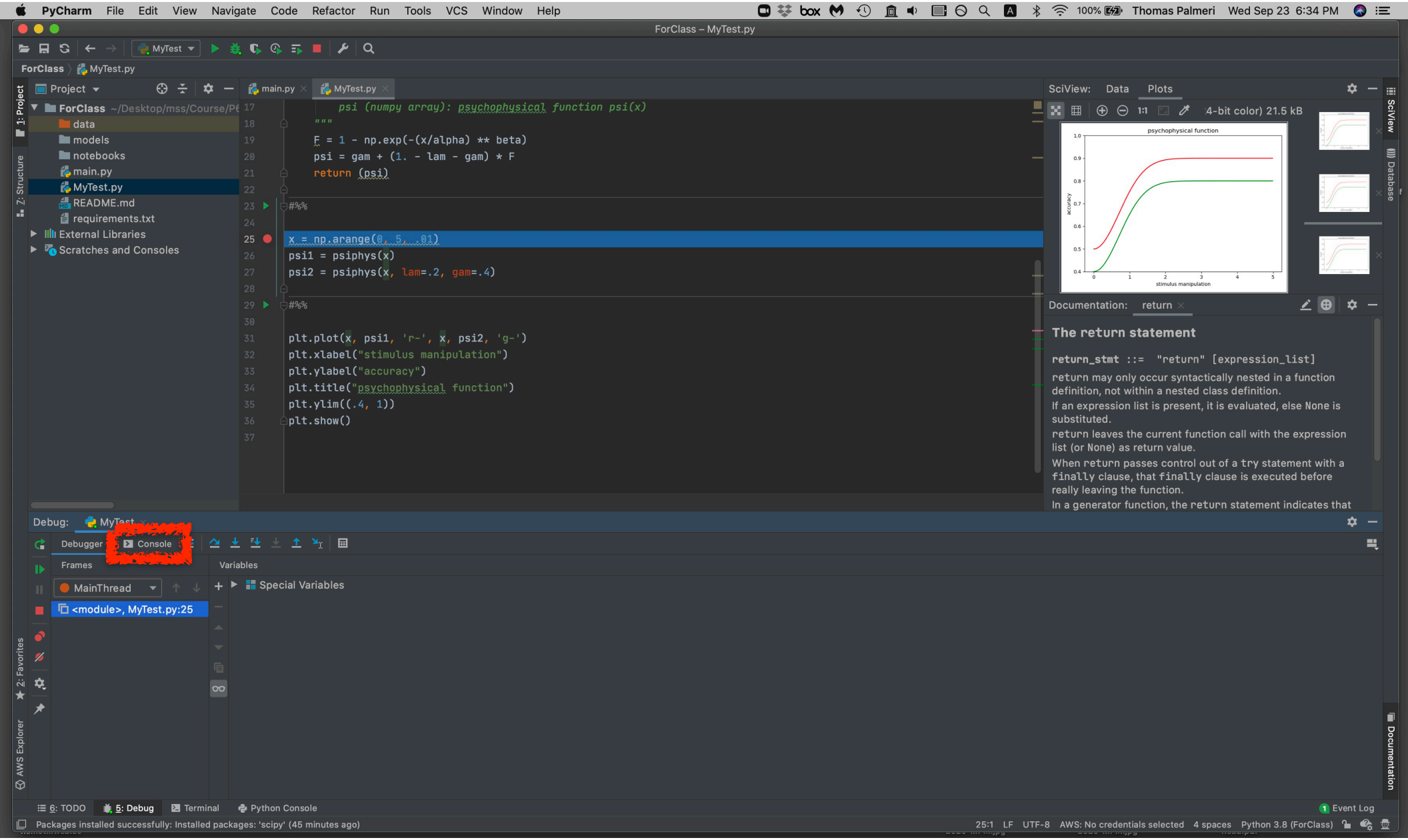
notice that the bottom has switched into Debug mode



can still access the Terminal



and the independent Python Console



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

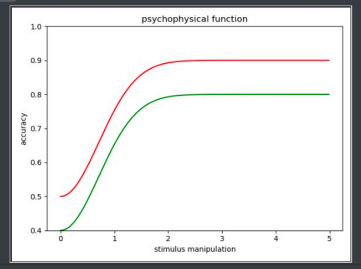
- ForClass ~/Desktop/mss/Course/P6
  - data
  - models
  - notebooks
  - main.py
  - MyTest.py
  - README.md
  - requirements.txt
- External Libraries
- Scratches and Consoles

main.py | 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



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

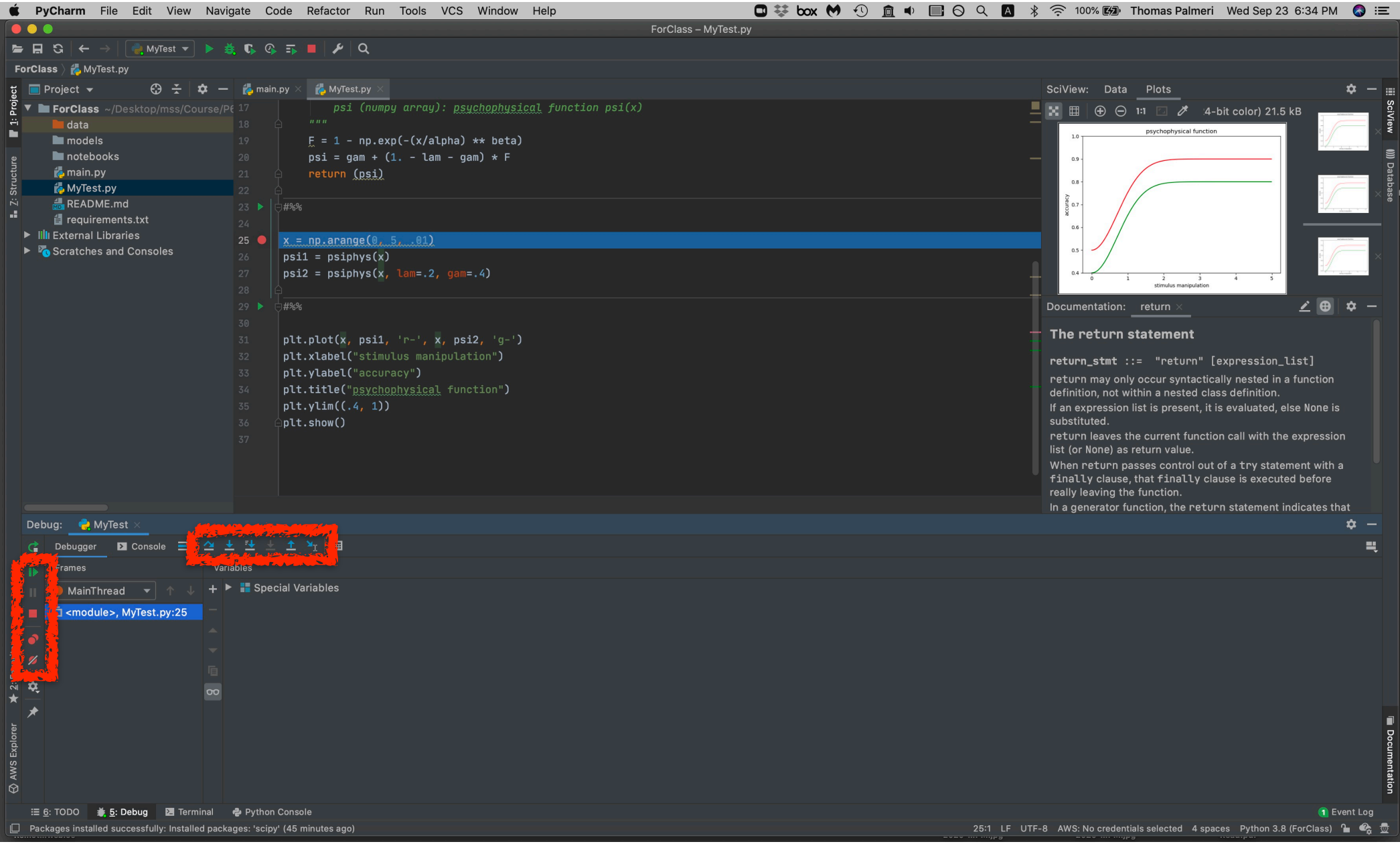
Debugger | Console

```
/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 "/Us
pydev debugger: process 85316 is connecting

Connected to pydev debugger (build 201.8743.11)

>>>
```

6: TODO | 5: Debug | Terminal | Python Console | Event Log

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

5:1 LF UTF-8 AWS: No credentials selected 4 spaces Python 3.8 (ForClass)



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

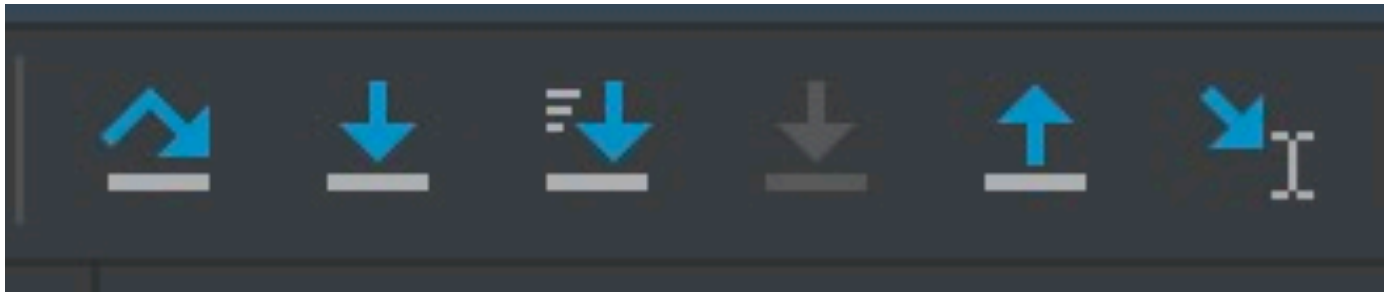
step over

step into

step into *my code*

step out

run to cursor



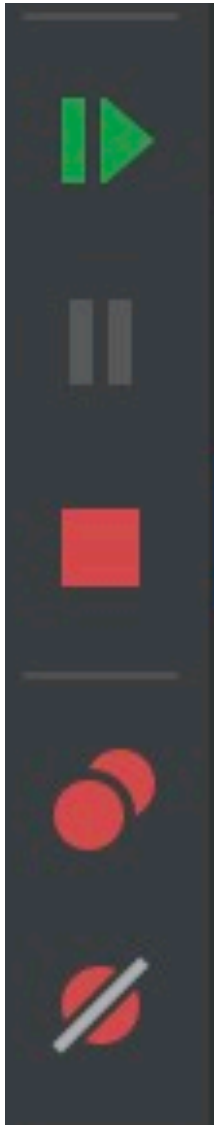
resume program

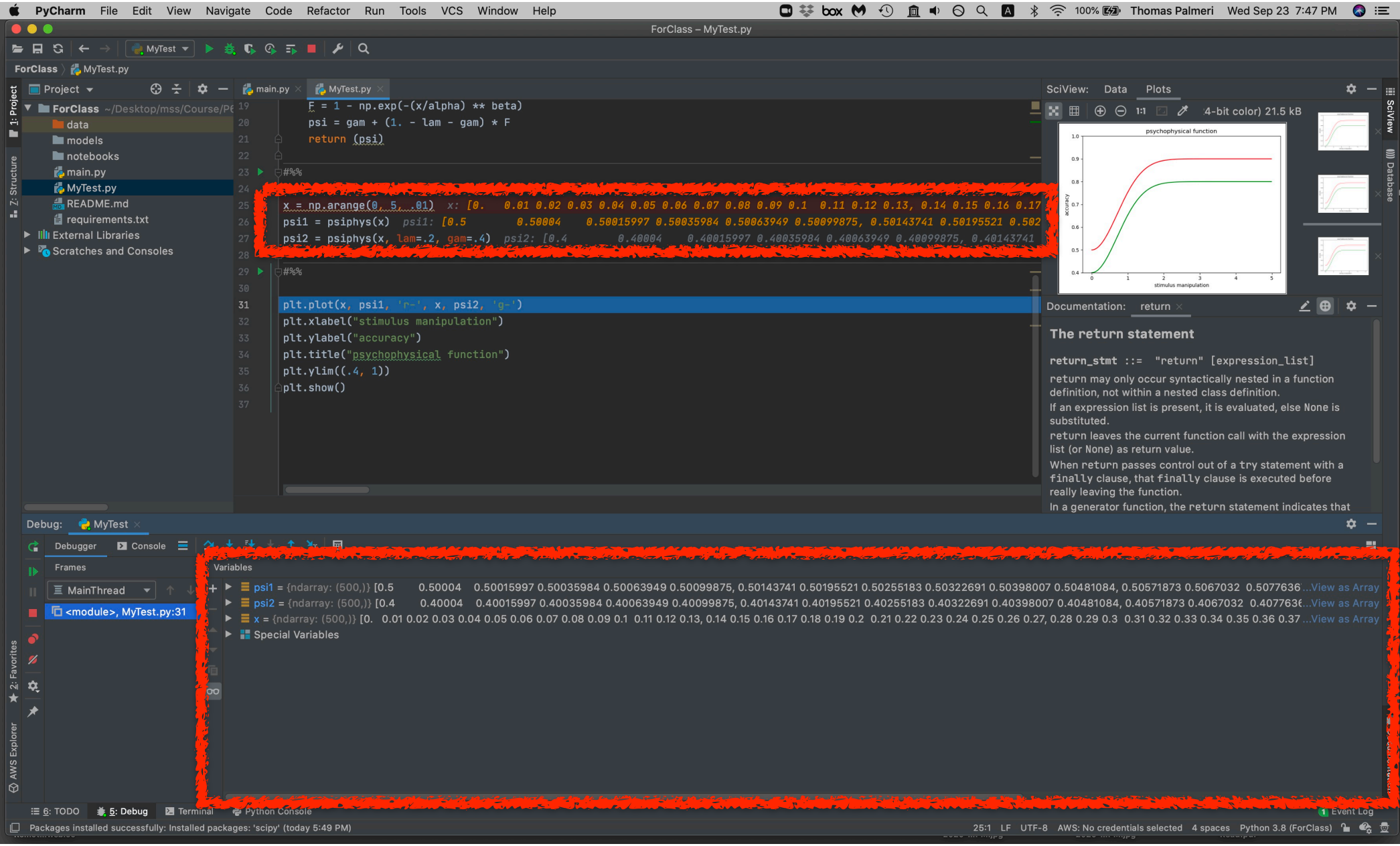
pause program

stop program

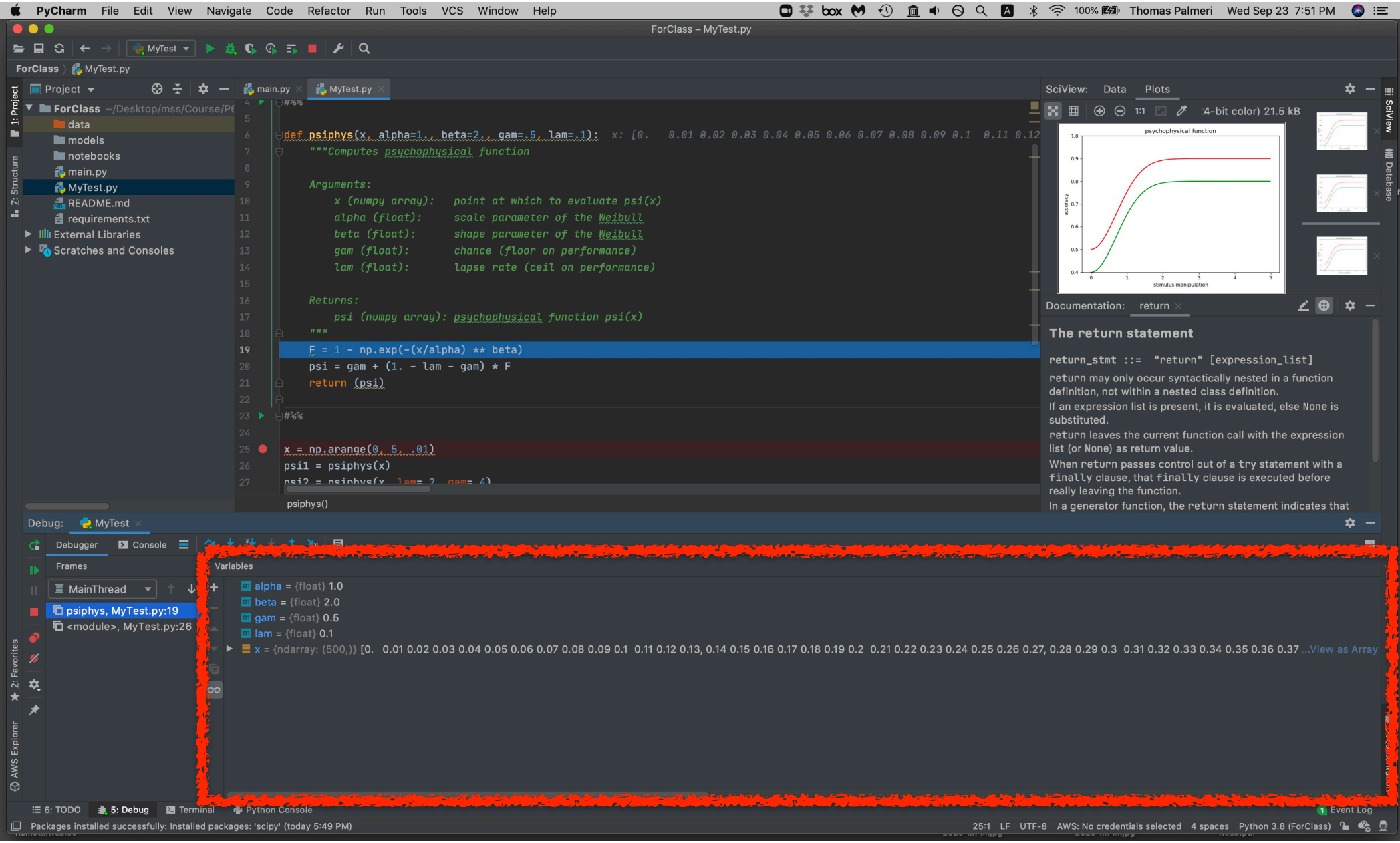
view breakpoints

mute breakpoints

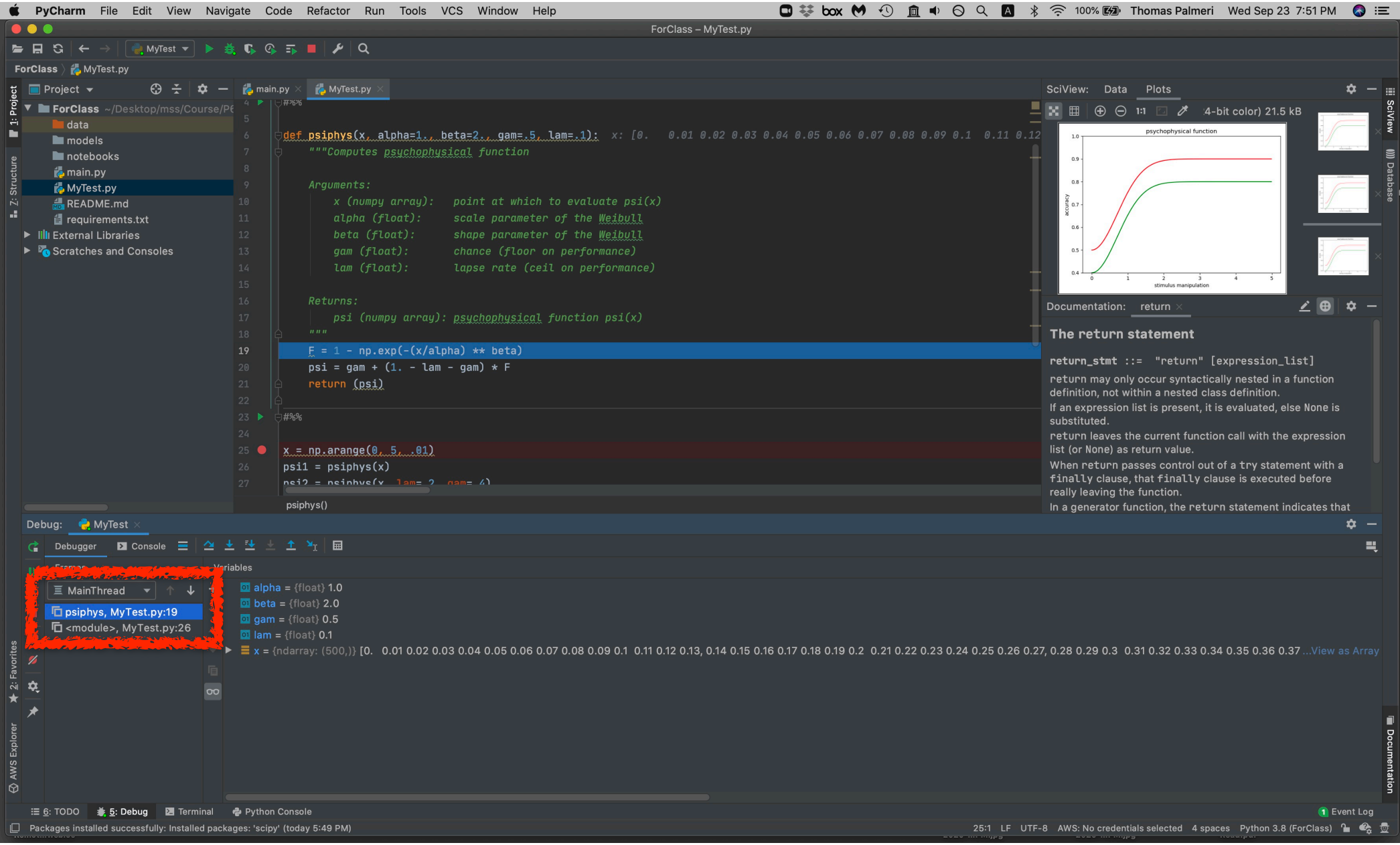




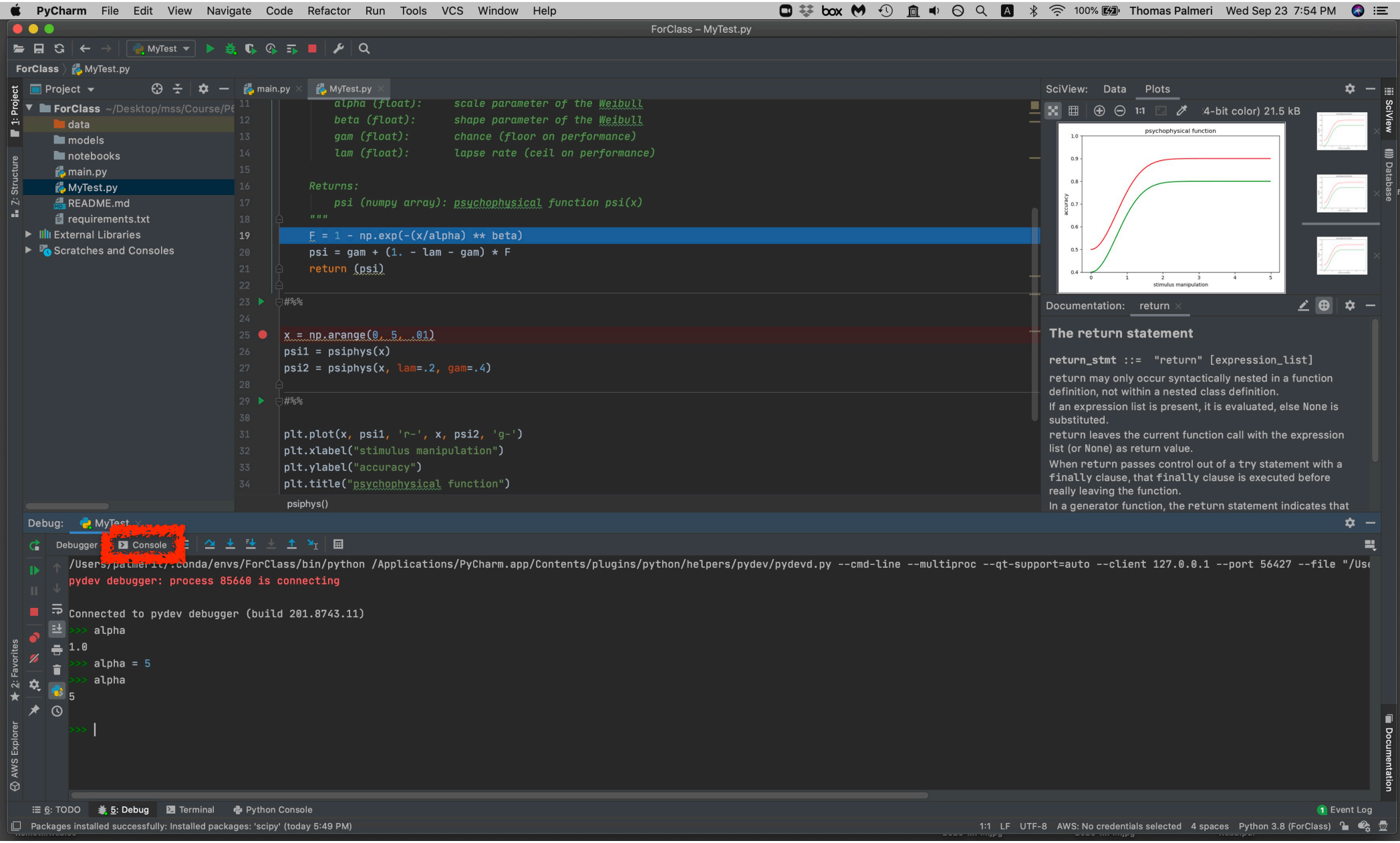
variable values are shown in the code and in the Variables window



stepping into a function, you see the local variables



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

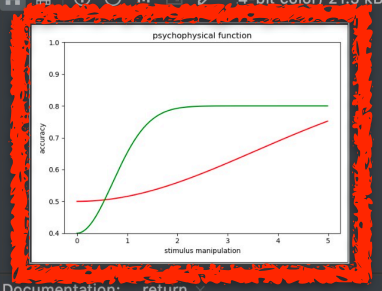


can view and change values of variables in the Console (tied to MyTest.py)

PyCharm File Edit View Navigate Code Refactor Run Tools VCS Window Help — Thomas Palmeri Wed Sep 23 7:55 PM

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")
```



The return statement

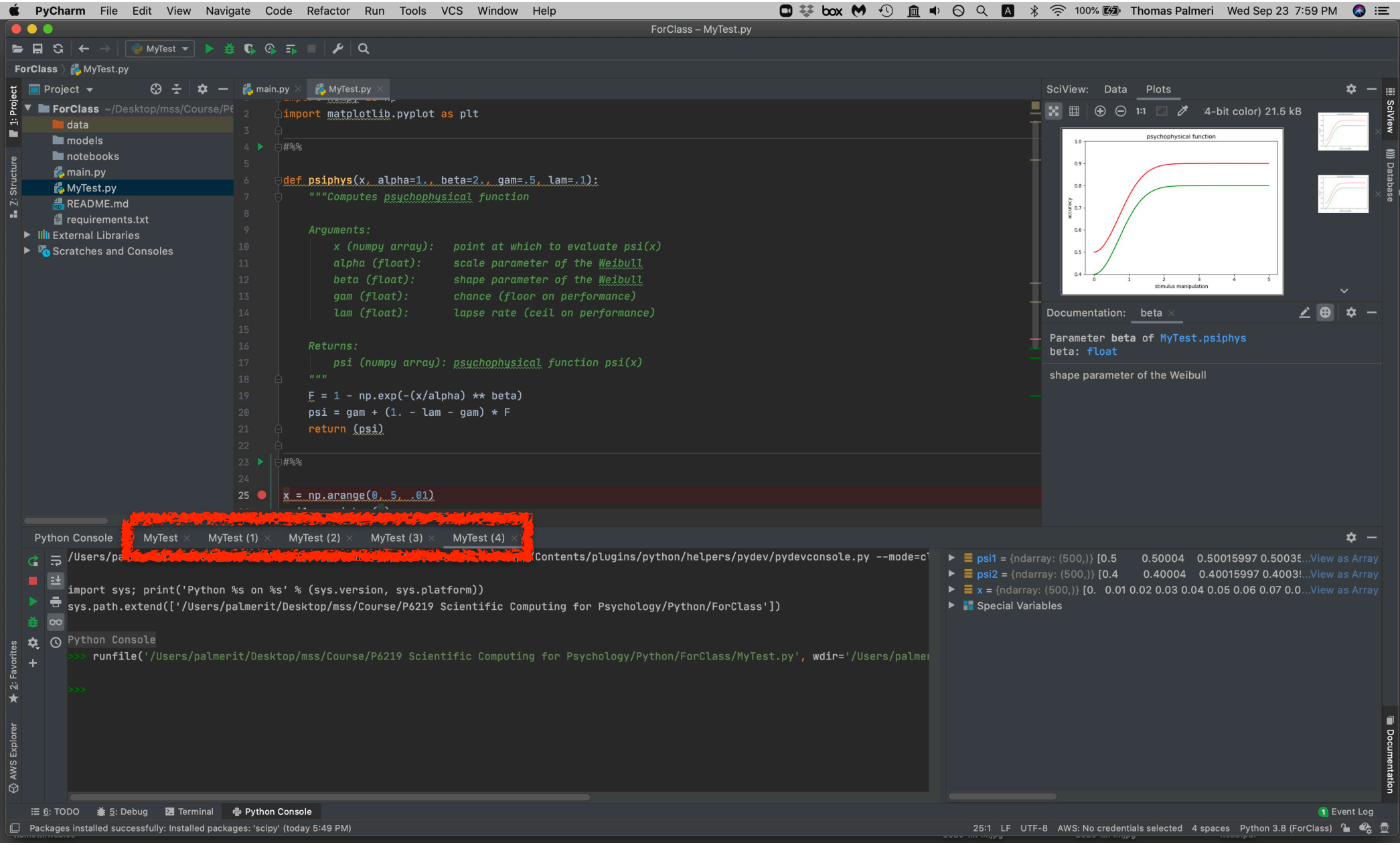
`return_stmt ::= "return" [expression_list]`

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

Frames are not available

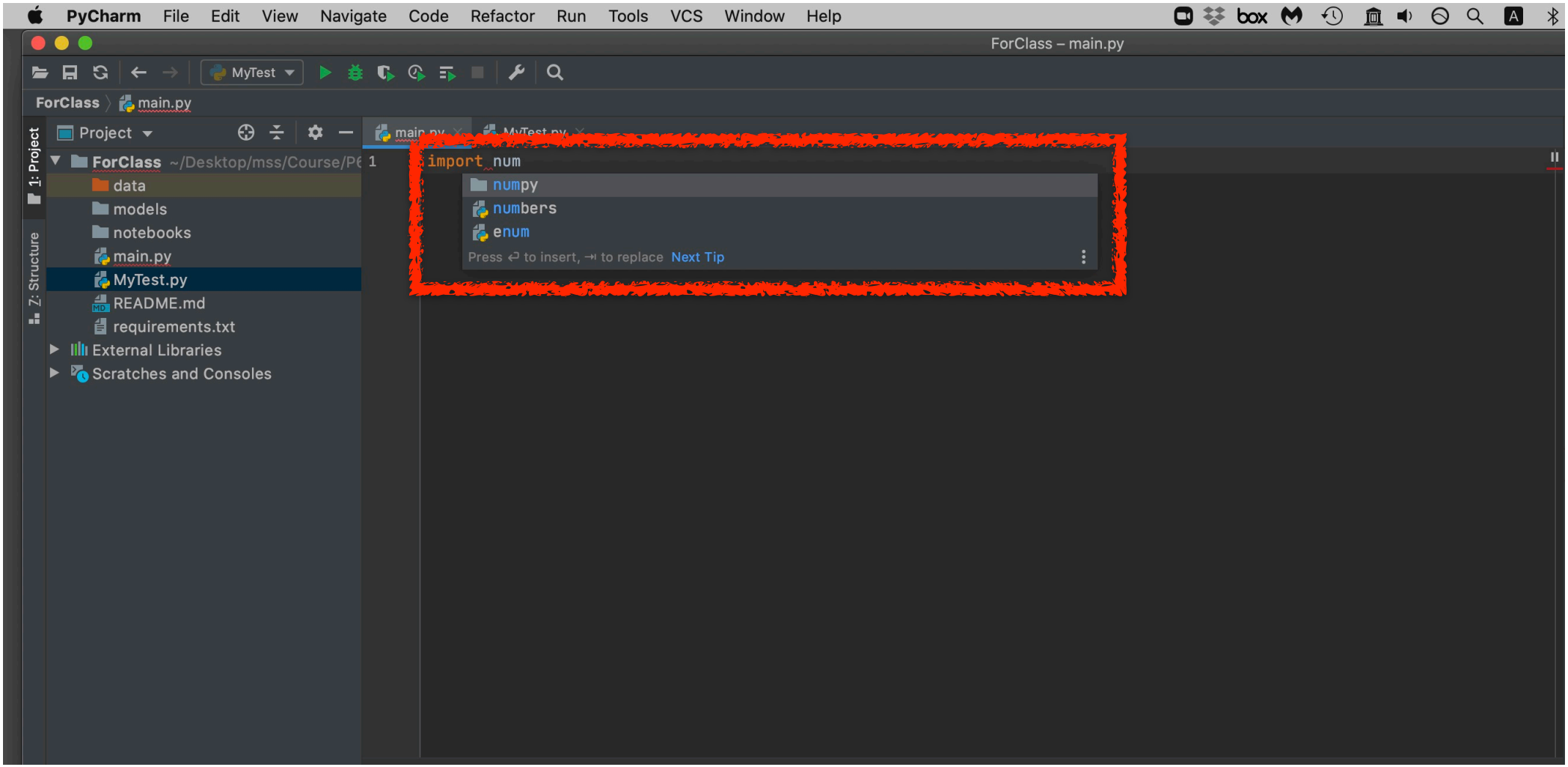
Connected

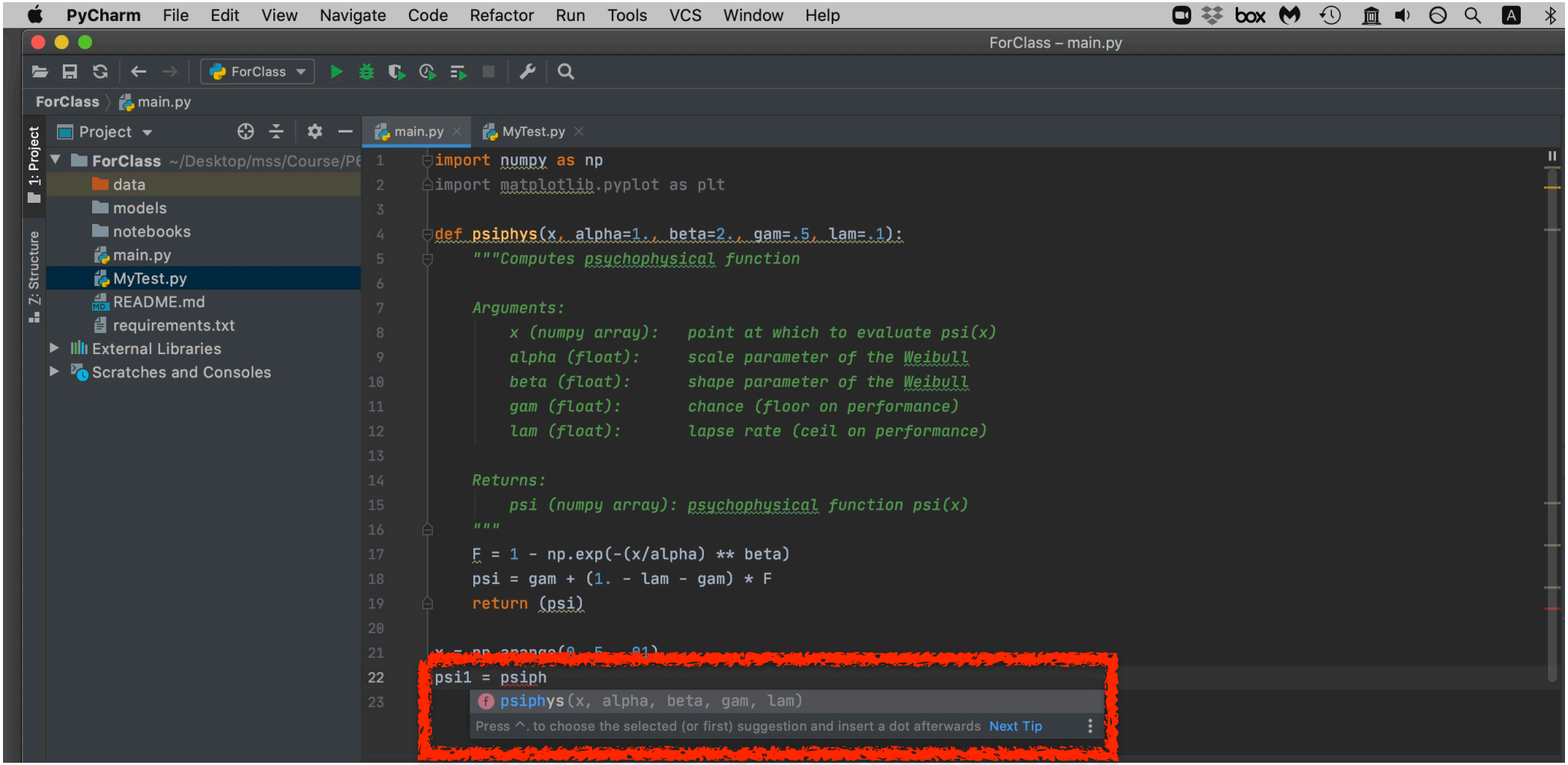
Packages installed successfully: Installed packages: 'scipy' (today 5:49 PM)



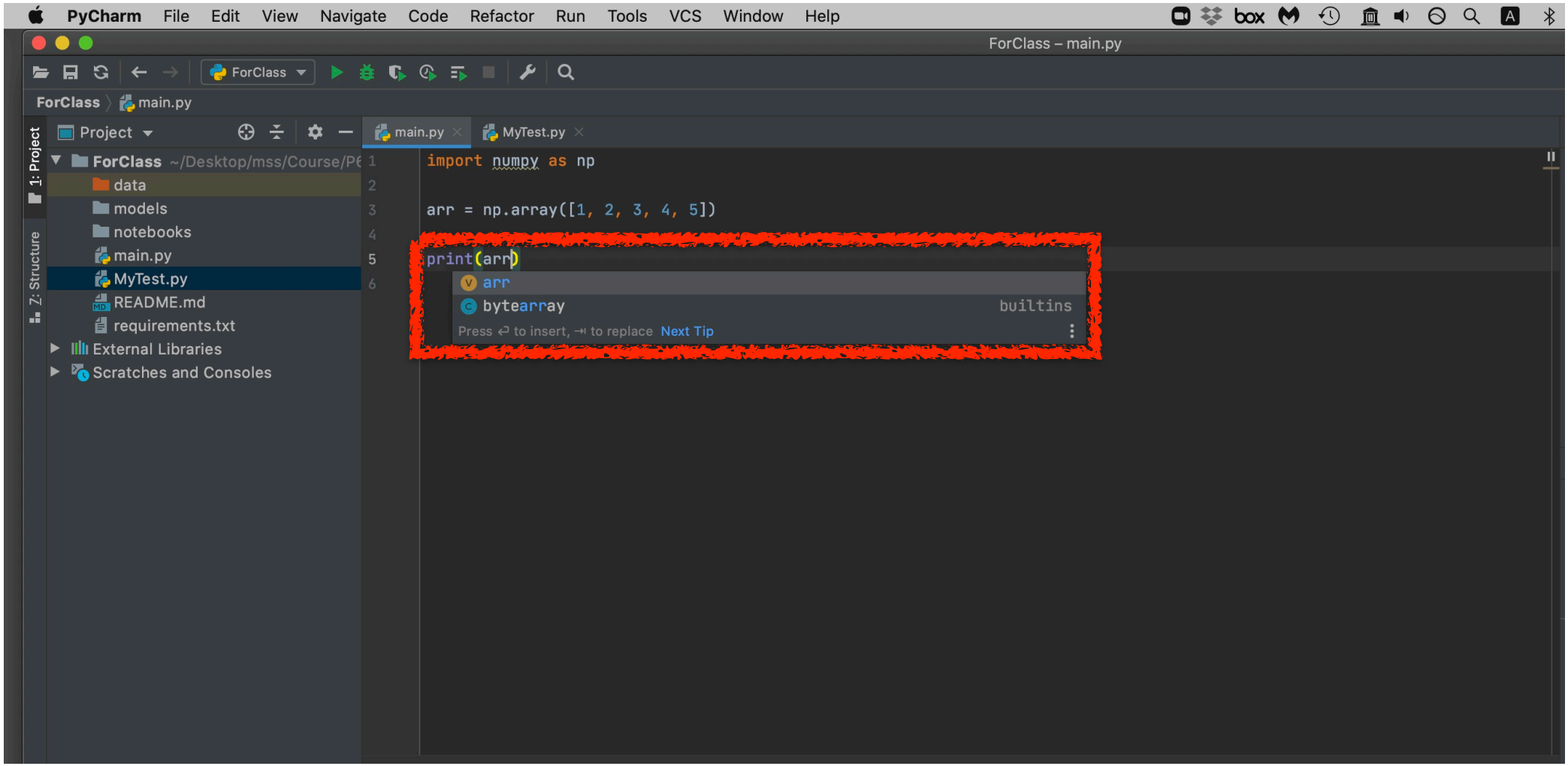
each time you run the .py file, you will create a new Console (so, you'll want to close them)

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

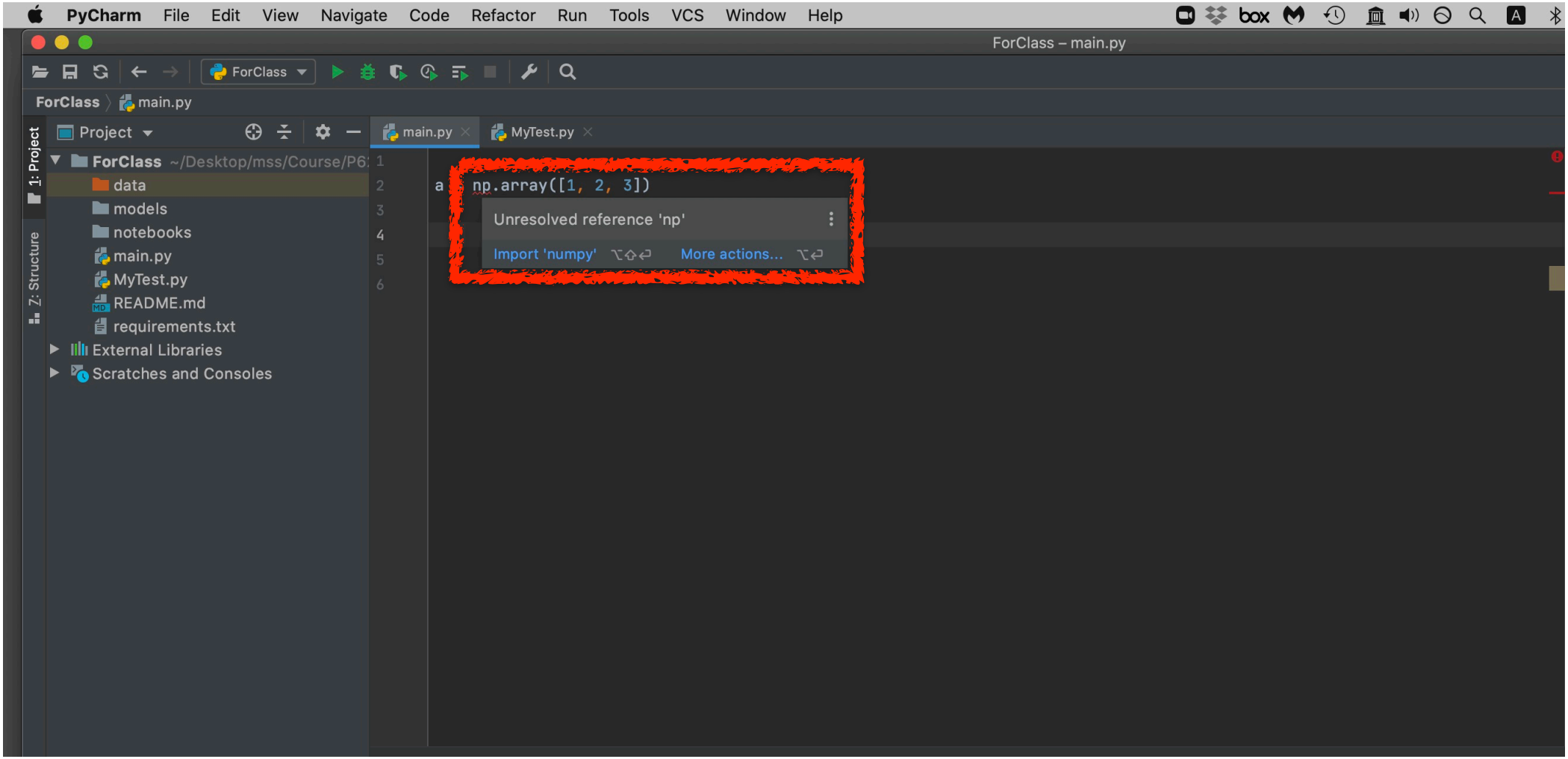




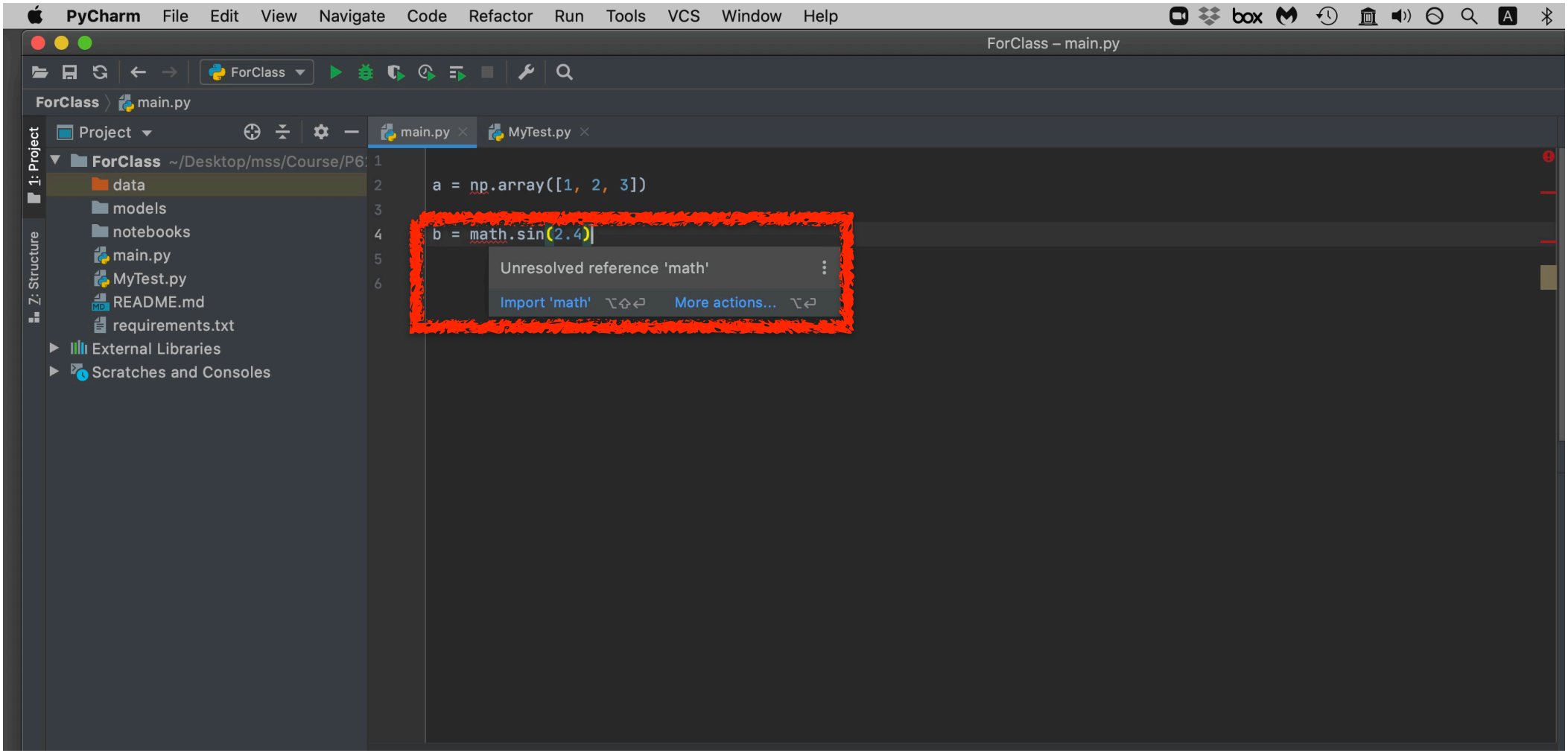
autocompletion as you're typing in code (Tab to complete, or use arrows and Enter to select)

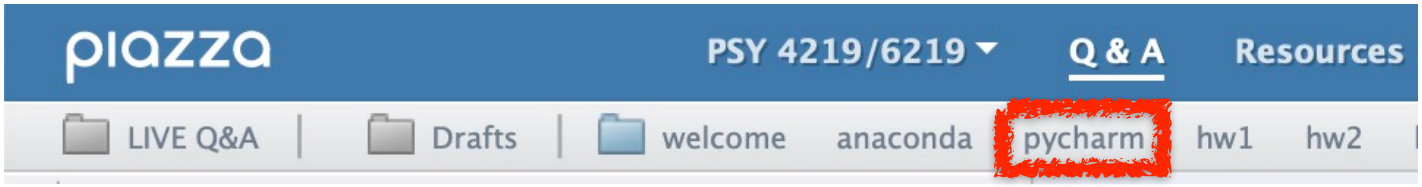

PyCharm File Edit View Navigate Code Refactor Run Tools VCS Window Help
ForClass – main.py
ForClass
ForClass 〉 main.py
Project
main.py
MyTest.py
ForClass ~/Desktop/mss/Course/P6
data
models
notebooks
main.py
MyTest.py
README.md
requirements.txt
External Libraries
Scratches and Consoles
1: Project
Z: Structure
1 import numpy as np
2
3 arr = np.array([1, 2, 3, 4, 5])
4
5 print(arr)
6
arr
bytearray builtins
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)





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