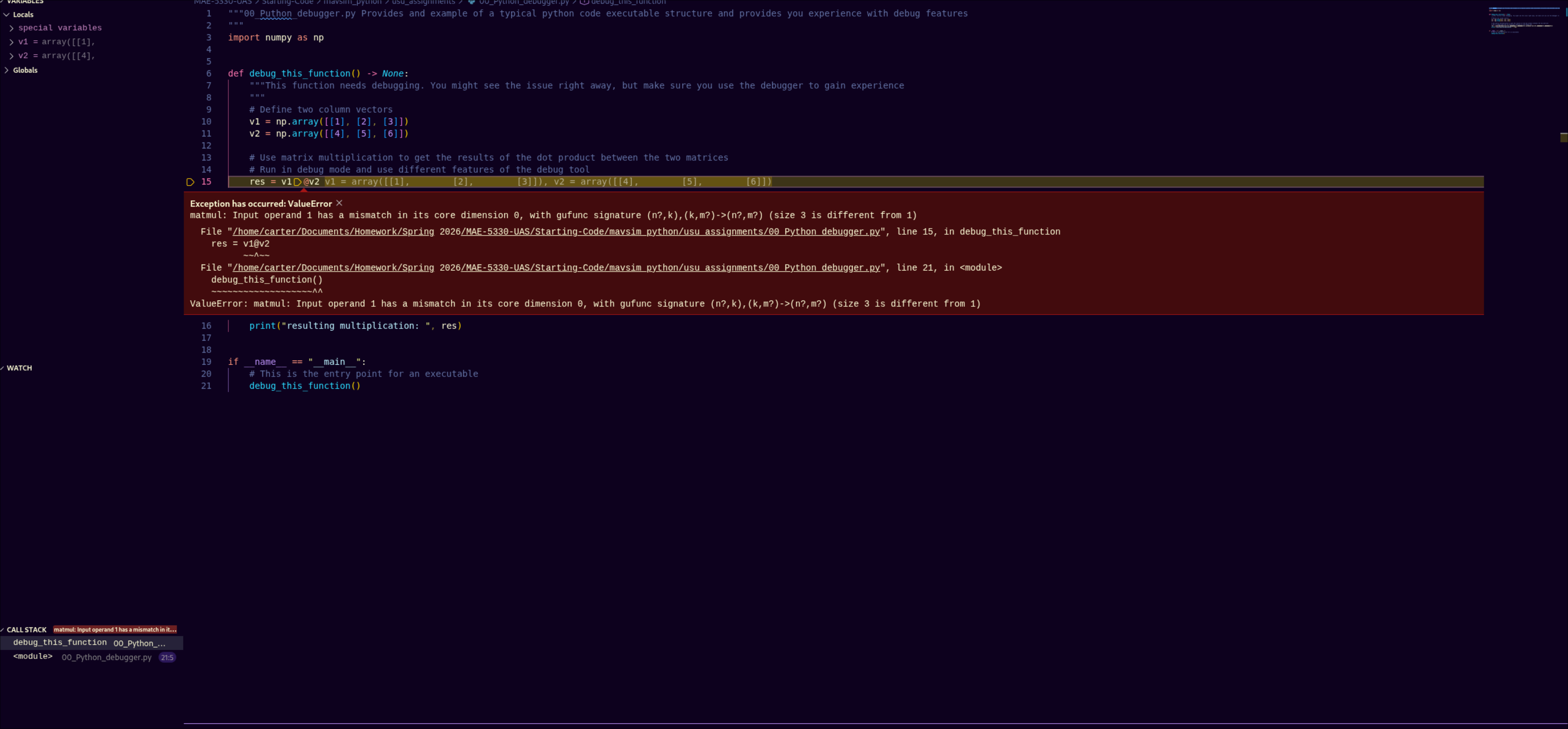


```
VARIABLES
∨ Locals
  > special variables
  > v1 = array([[1],
  > v2 = array([[4],
> Globals
```

```
MAE-5330-UAS > Starting-Code > mavsim_python > usu_assignments > 00_Python_debugger.py > debug_this_function
```

```
 1  """00_Python_debugger.py Provides and example of a typical python code executable structure and provides you experience with debug features
 2  """
 3  import numpy as np
 4
 5
 6  def debug_this_function() -> None:
 7      """This function needs debugging. You might see the issue right away, but make sure you use the debugger to gain experience
 8      """
 9      # Define two column vectors
10      v1 = np.array([[1], [2], [3]])
11      v2 = np.array([[4], [5], [6]])
12
13      # Use matrix multiplication to get the results of the dot product between the two matrices
14      # Run in debug mode and use different features of the debug tool
15      res = v1@v2 v1 = array([[1],        [2],        [3]]), v2 = array([[4],        [5],        [6]])
```

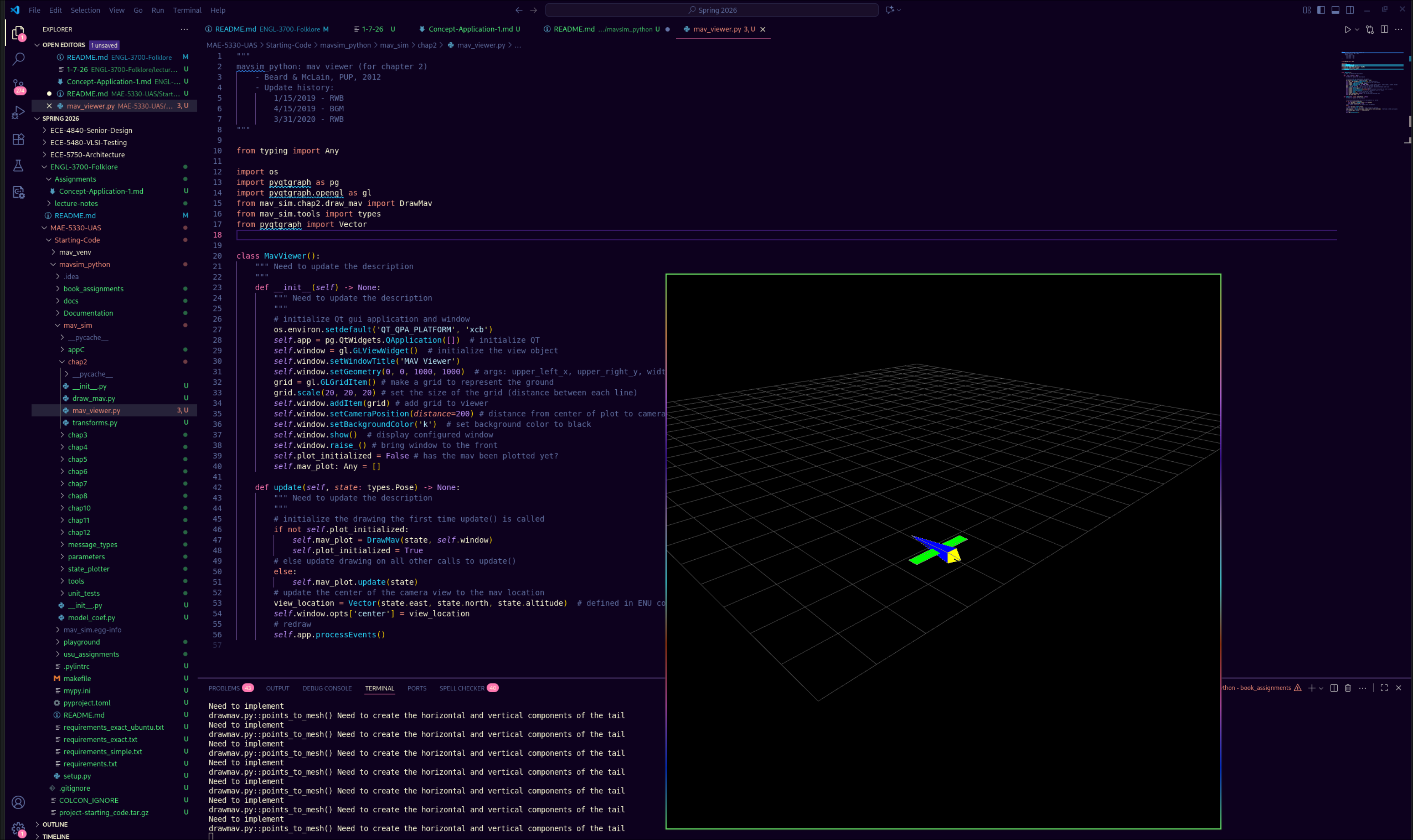
Exception has occurred: ValueError ×

```
matmul: Input operand 1 has a mismatch in its core dimension 0, with gufunc signature (n?,k),(k,m?)->(n?,m?) (size 3 is different from 1)

  File "/home/carter/Documents/Homework/Spring 2026/MAE-5330-UAS/Starting-Code/mavsim_python/usu_assignments/00_Python_debugger.py", line 15, in debug_this_function
    res = v1@v2
          ~~^~~
  File "/home/carter/Documents/Homework/Spring 2026/MAE-5330-UAS/Starting-Code/mavsim_python/usu_assignments/00_Python_debugger.py", line 21, in <module>
    debug_this_function()
    ~~~~~~~~~~~~~~~~~~~^^
ValueError: matmul: Input operand 1 has a mismatch in its core dimension 0, with gufunc signature (n?,k),(k,m?)->(n?,m?) (size 3 is different from 1)
```

```
16      print("resulting multiplication: ", res)
17
18
19  if __name__ == "__main__":
20      # This is the entry point for an executable
21      debug_this_function()
```

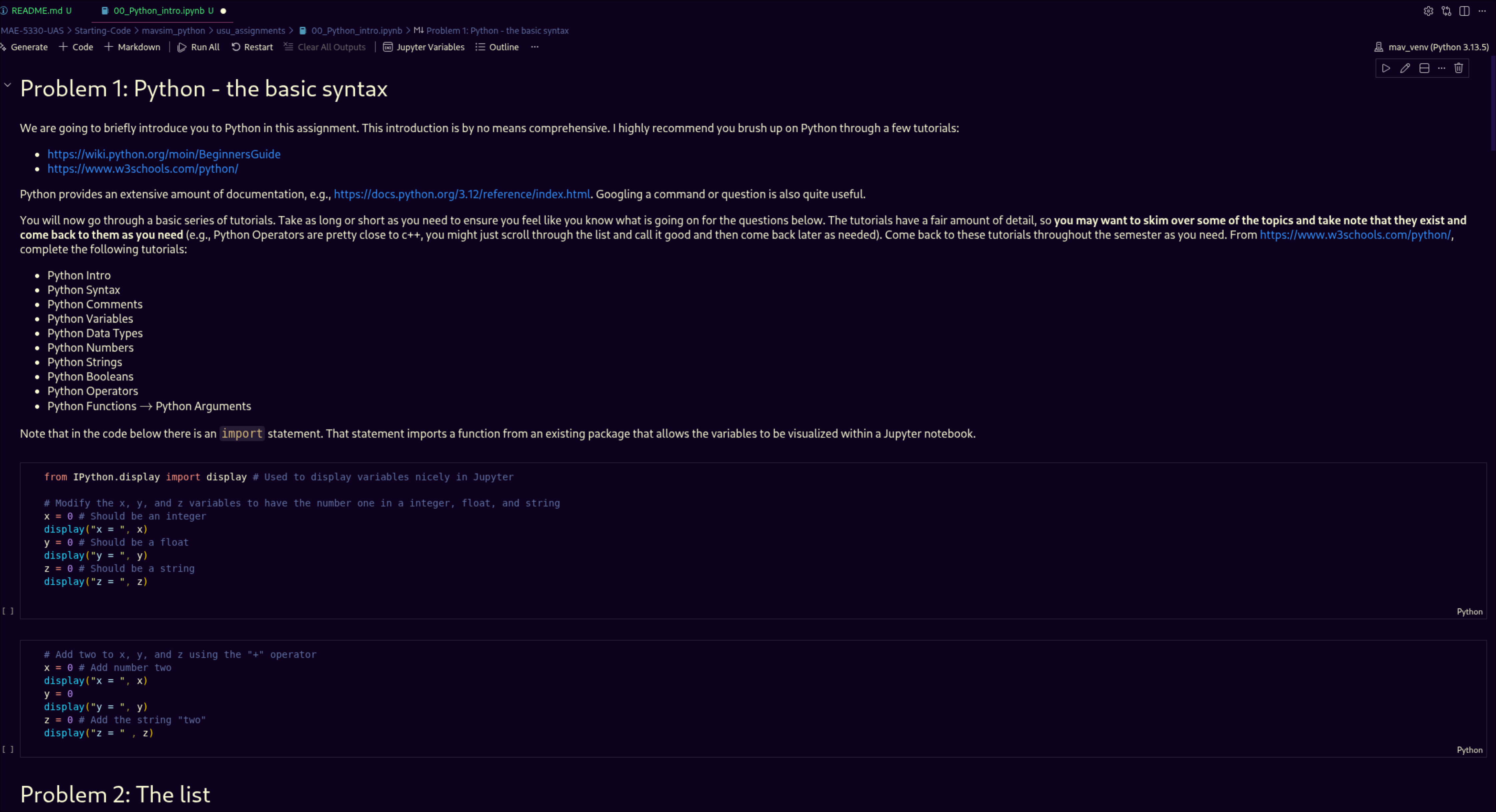
```
∨ WATCH
```

```
∨ CALL STACK   matmul: Input operand 1 has a mismatch in it...
  debug_this_function   00_Python_...
  <module>   00_Python_debugger.py   21:5
```



```python
"""
mavsim_python: mav viewer (for chapter 2)
    - Beard & McLain, PUP, 2012
    - Update history:
        1/15/2019 - RWB
        4/15/2019 - BGM
        3/31/2020 - RWB
"""

from typing import Any

import os
import pyqtgraph as pg
import pyqtgraph.opengl as gl
from mav_sim.chap2.draw_mav import DrawMav
from mav_sim.tools import types
from pyqtgraph import Vector


class MavViewer():
    """ Need to update the description
    """
    def __init__(self) -> None:
        """ Need to update the description
        """
        # initialize Qt gui application and window
        os.environ.setdefault('QT_QPA_PLATFORM', 'xcb')
        self.app = pg.QtWidgets.QApplication([])  # initialize QT
        self.window = gl.GLViewWidget()  # initialize the view object
        self.window.setWindowTitle('MAV Viewer')
        self.window.setGeometry(0, 0, 1000, 1000)  # args: upper_left_x, upper_right_y, widt
        grid = gl.GLGridItem() # make a grid to represent the ground
        grid.scale(20, 20, 20) # set the size of the grid (distance between each line)
        self.window.addItem(grid) # add grid to viewer
        self.window.setCameraPosition(distance=200) # distance from center of plot to camera
        self.window.setBackgroundColor('k')  # set background color to black
        self.window.show()  # display configured window
        self.window.raise_() # bring window to the front
        self.plot_initialized = False # has the mav been plotted yet?
        self.mav_plot: Any = []

    def update(self, state: types.Pose) -> None:
        """ Need to update the description
        """
        # initialize the drawing the first time update() is called
        if not self.plot_initialized:
            self.mav_plot = DrawMav(state, self.window)
            self.plot_initialized = True
        # else update drawing on all other calls to update()
        else:
            self.mav_plot.update(state)
        # update the center of the camera view to the mav location
        view_location = Vector(state.east, state.north, state.altitude)  # defined in ENU co
        self.window.opts['center'] = view_location
        # redraw
        self.app.processEvents()
```

```
Need to implement
drawmav.py::points_to_mesh() Need to create the horizontal and vertical components of the tail
Need to implement
drawmav.py::points_to_mesh() Need to create the horizontal and vertical components of the tail
Need to implement
drawmav.py::points_to_mesh() Need to create the horizontal and vertical components of the tail
Need to implement
drawmav.py::points_to_mesh() Need to create the horizontal and vertical components of the tail
Need to implement
drawmav.py::points_to_mesh() Need to create the horizontal and vertical components of the tail
Need to implement
drawmav.py::points_to_mesh() Need to create the horizontal and vertical components of the tail
Need to implement
drawmav.py::points_to_mesh() Need to create the horizontal and vertical components of the tail
```



Problem 1: Python - the basic syntax

We are going to briefly introduce you to Python in this assignment. This introduction is by no means comprehensive. I highly recommend you brush up on Python through a few tutorials:

- https://wiki.python.org/moin/BeginnersGuide
- https://www.w3schools.com/python/

Python provides an extensive amount of documentation, e.g., https://docs.python.org/3.12/reference/index.html. Googling a command or question is also quite useful.

You will now go through a basic series of tutorials. Take as long or short as you need to ensure you feel like you know what is going on for the questions below. The tutorials have a fair amount of detail, so **you may want to skim over some of the topics and take note that they exist and come back to them as you need** (e.g., Python Operators are pretty close to c++, you might just scroll through the list and call it good and then come back later as needed). Come back to these tutorials throughout the semester as you need. From https://www.w3schools.com/python/, complete the following tutorials:

- Python Intro
- Python Syntax
- Python Comments
- Python Variables
- Python Data Types
- Python Numbers
- Python Strings
- Python Booleans
- Python Operators
- Python Functions $\longrightarrow$ Python Arguments

Note that in the code below there is an `import` statement. That statement imports a function from an existing package that allows the variables to be visualized within a Jupyter notebook.

```python
from IPython.display import display # Used to display variables nicely in Jupyter

# Modify the x, y, and z variables to have the number one in a integer, float, and string
x = 0 # Should be an integer
display("x = ", x)
y = 0 # Should be a float
display("y = ", y)
z = 0 # Should be a string
display("z = ", z)
```

[ ] Python

```python
# Add two to x, y, and z using the "+" operator
x = 0 # Add number two
display("x = ", x)
y = 0
display("y = ", y)
z = 0 # Add the string "two"
display("z = " , z)
```

[ ] Python

Problem 2: The list