

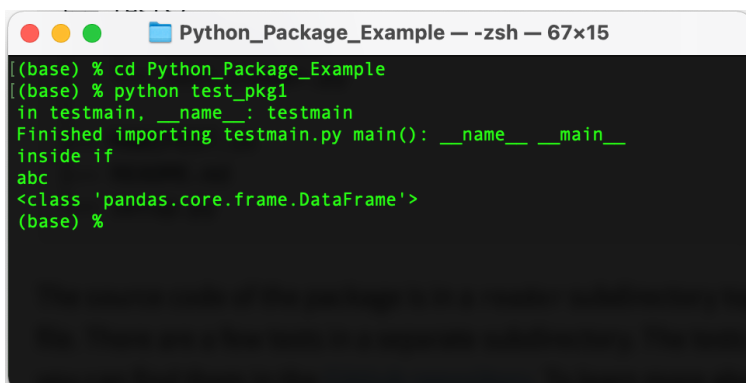
Python Project Setup

From the command line, running the package (>>python pkg_name) calls the `__main__.py` code at the package subfolder's root level. In the example, **test_pkg1** is self-contained with all needed code in its sub-folder.

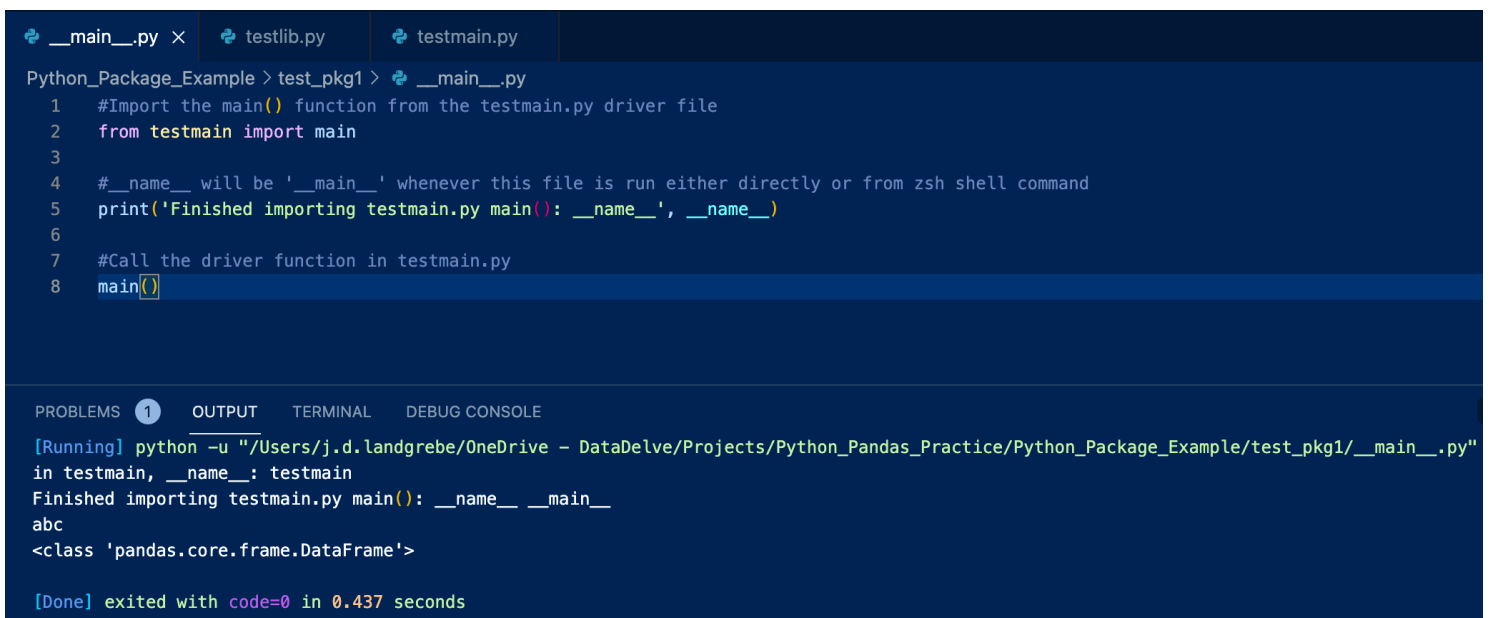
Example folder structure:

```
Python_Pandas_Practice\  
  |__ Python_Package_Example\  
      |  
      |__ test_pkg1\  
          |  
          |__ __main__.py  
          |__ testmain.py  
          |__ testlib.py  
      |  
      |__ test_pkg2\  
          |__ etc.  
      |  
      |__ tests\  
      |__ Readme.md  
      |__ etc.
```

Running from the `Python_Package_Example` folder is equivalent to running **`__main__.py`** from within VS Code:



```
Python_Package_Example - zsh - 67x15  
[(base) % cd Python_Package_Example  
[(base) % python test_pkg1  
in testmain, __name__: testmain  
Finished importing testmain.py main(): __name__ __main__  
inside if  
abc  
<class 'pandas.core.frame.DataFrame'>  
(base) %
```



```
__main__.py x testlib.py testmain.py  
Python_Package_Example > test_pkg1 > __main__.py  
1 #Import the main() function from the testmain.py driver file  
2 from testmain import main  
3  
4 #__name__ will be '__main__' whenever this file is run either directly or from zsh shell command  
5 print('Finished importing testmain.py main(): __name__ __main__')  
6  
7 #Call the driver function in testmain.py  
8 main()  
  
PROBLEMS 1 OUTPUT TERMINAL DEBUG CONSOLE  
[Running] python -u "/Users/j.d.landgrebe/OneDrive - DataDelve/Projects/Python_Pandas_Practice/Python_Package_Example/test_pkg1/__main__.py"  
in testmain, __name__: testmain  
Finished importing testmain.py main(): __name__ __main__  
abc  
<class 'pandas.core.frame.DataFrame'>  
[Done] exited with code=0 in 0.437 seconds
```

The driver file **testmain.py** and an imported Class in **testlib.py** showing how code can be subdivided into

The driver file, **testmain.py** and an imported class in **testlib.py** showing how code can be subdivided into multiple files within the package.

- Note that the package top-level driver code is in the **main()** function to control not running it when simply importing **testmain** as a library by other code
- The lines at bottom are standard and usually placed at the end of a *.py file. These lines run the **main()** function if testmain.py is run

```
Python_Package_Example > test_pkg1 > testmain.py > ...
1  import testlib
2
3  def main():
4      #Instantiate the class
5      cl = testlib.TestClass('abc')
6      print(cl.param)
7      print(type(cl.df))
8
9  print('in testmain, __name__:', __name__)
10 if __name__ == '__main__':
11     print('Running standalone from testmain.py')
12     main()
```

PROBLEMS 1 OUTPUT TERMINAL DEBUG CONSOLE

```
[Running] python -u "/Users/j.d.landgrebe/OneDrive - DataDelve/Projects/Python_Pandas_Practice/Python_Package_Example/test_pkg1/testmain.py"
in testmain, __name__: __main__
Running standalone
abc
<class 'pandas.core.frame.DataFrame'>

[Done] exited with code=0 in 0.564 seconds
```

```
Python_Package_Example > test_pkg1 > testlib.py > ...
1  import pandas as pd
2  import numpy as np
3
4  class TestClass:
5      def __init__(self, param):
6          self.param = param
7          self.df = pd.DataFrame()
8          return
```