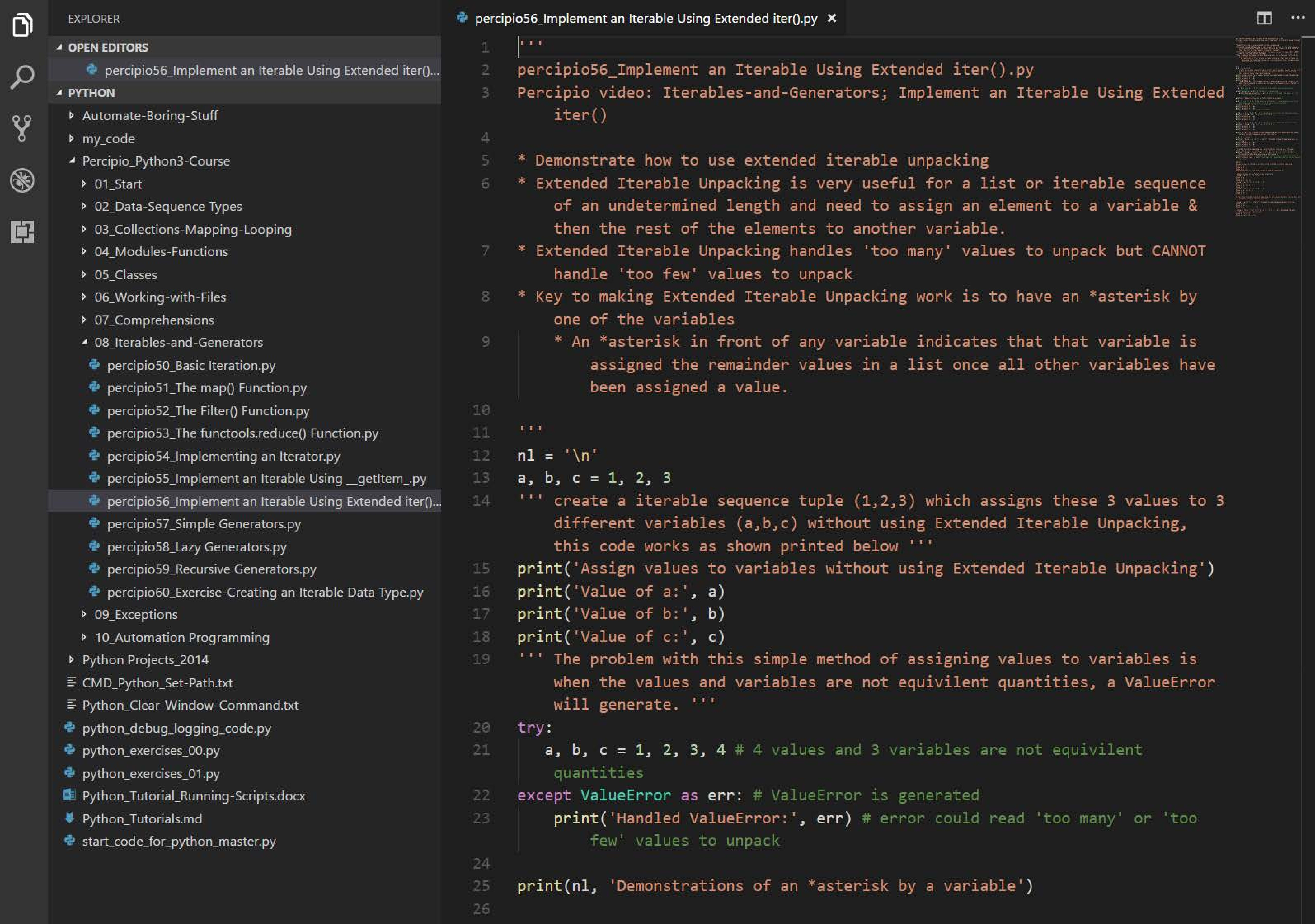
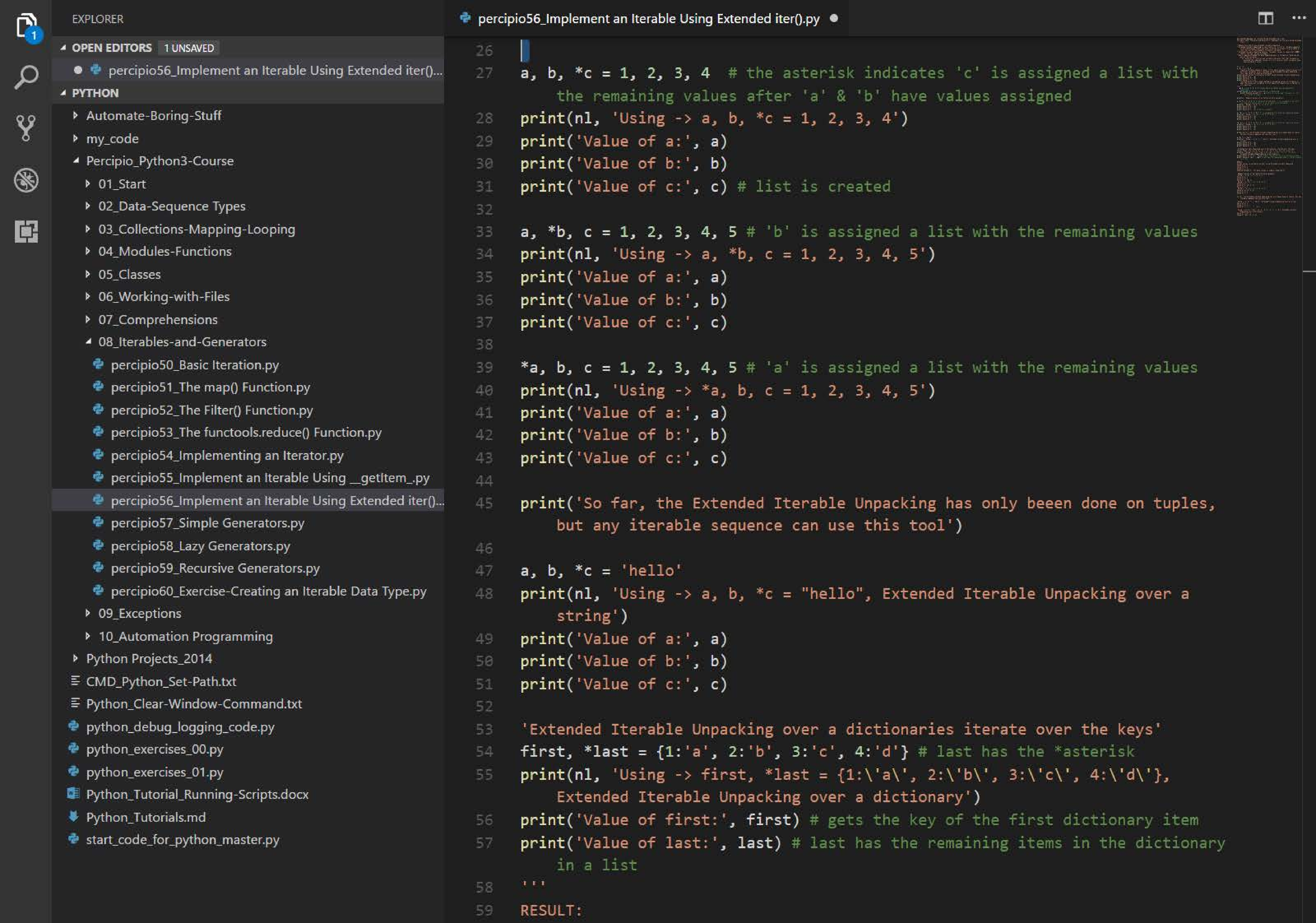
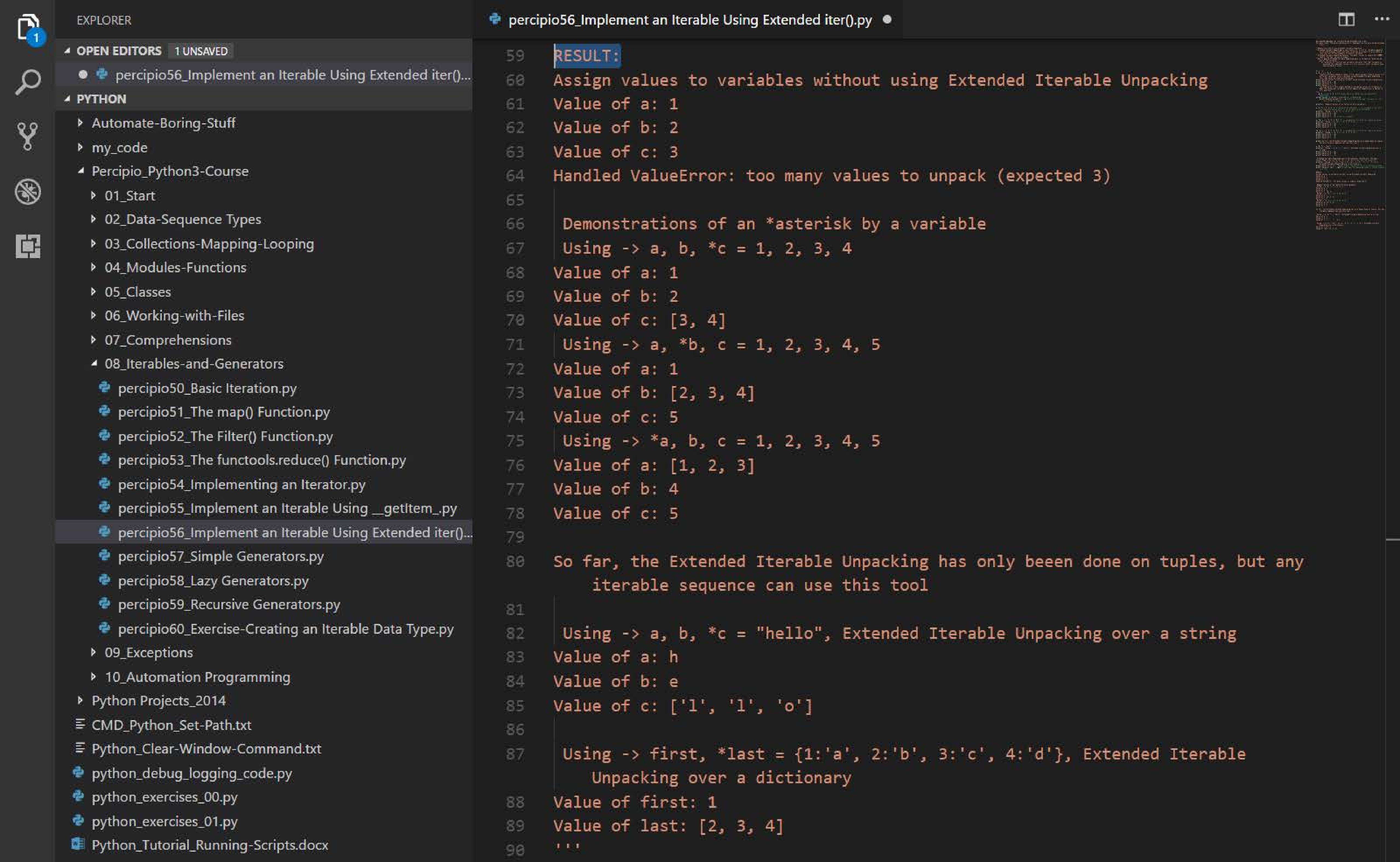

EXPLORER
OPEN EDITORS
percipio56_Implement an Iterable Using Extended iter()...
PYTHON
Automate-Boring-Stuff
my_code
Percipio_Python3-Course
01_Start
02_Data-Sequence Types
03_Collections-Mapping-Looping
04_Modules-Functions
05_Classes
06_Working-with-Files
07_Comprehensions
08_Iterables-and-Generators
percipio50_Basic Iteration.py
percipio51_The map() Function.py
percipio52_The Filter() Function.py
percipio53_The functools.reduce() Function.py
percipio54_Implementing an Iterator.py
percipio55_Implement an Iterable Using __getItem_.py
percipio56_Implement an Iterable Using Extended iter()...
percipio57_Simple Generators.py
percipio58_Lazy Generators.py
percipio59_Recursive Generators.py
percipio60_Exercise-Creating an Iterable Data Type.py
09_Exceptions
10_Automation Programming
Python Projects_2014
CMD_Python_Set-Path.txt
Python_Clear-Window-Command.txt
python_debug_logging_code.py
python_exercises_00.py
python_exercises_01.py
Python_Tutorial_Running-Scripts.docx
Python_Tutorials.md
start_code_for_python_master.py
percipio56_Implement an Iterable Using Extended iter().py

```python
'''
percipio56_Implement an Iterable Using Extended iter().py
Percipio video: Iterables-and-Generators; Implement an Iterable Using Extended
    iter()

* Demonstrate how to use extended iterable unpacking
* Extended Iterable Unpacking is very useful for a list or iterable sequence
    of an undetermined length and need to assign an element to a variable &
    then the rest of the elements to another variable.
* Extended Iterable Unpacking handles 'too many' values to unpack but CANNOT
    handle 'too few' values to unpack
* Key to making Extended Iterable Unpacking work is to have an *asterisk by
    one of the variables
    * An *asterisk in front of any variable indicates that that variable is
        assigned the remainder values in a list once all other variables have
        been assigned a value.

'''
nl = '\n'
a, b, c = 1, 2, 3
''' create a iterable sequence tuple (1,2,3) which assigns these 3 values to 3
    different variables (a,b,c) without using Extended Iterable Unpacking,
    this code works as shown printed below '''
print('Assign values to variables without using Extended Iterable Unpacking')
print('Value of a:', a)
print('Value of b:', b)
print('Value of c:', c)
''' The problem with this simple method of assigning values to variables is
    when the values and variables are not equivilent quantities, a ValueError
    will generate. '''
try:
    a, b, c = 1, 2, 3, 4 # 4 values and 3 variables are not equivilent
    quantities
except ValueError as err: # ValueError is generated
    print('Handled ValueError:', err) # error could read 'too many' or 'too
        few' values to unpack

print(nl, 'Demonstrations of an *asterisk by a variable')

```


EXPLORER
OPEN EDITORS 1 UNSAVED
percipio56_Implement an Iterable Using Extended iter()...
PYTHON
Automate-Boring-Stuff
my_code
Percipio_Python3-Course
01_Start
02_Data-Sequence Types
03_Collections-Mapping-Looping
04_Modules-Functions
05_Classes
06_Working-with-Files
07_Comprehensions
08_Iterables-and-Generators
percipio50_Basic Iteration.py
percipio51_The map() Function.py
percipio52_The Filter() Function.py
percipio53_The functools.reduce() Function.py
percipio54_Implementing an Iterator.py
percipio55_Implement an Iterable Using __getItem_.py
percipio56_Implement an Iterable Using Extended iter()...
percipio57_Simple Generators.py
percipio58_Lazy Generators.py
percipio59_Recursive Generators.py
percipio60_Exercise-Creating an Iterable Data Type.py
09_Exceptions
10_Automation Programming
Python Projects_2014
CMD_Python_Set-Path.txt
Python_Clear-Window-Command.txt
python_debug_logging_code.py
python_exercises_00.py
python_exercises_01.py
Python_Tutorial_Running-Scripts.docx
Python_Tutorials.md
start_code_for_python_master.py
percipio56_Implement an Iterable Using Extended iter().py

```python

a, b, *c = 1, 2, 3, 4  # the asterisk indicates 'c' is assigned a list with the remaining values after 'a' & 'b' have values assigned
print(nl, 'Using -> a, b, *c = 1, 2, 3, 4')
print('Value of a:', a)
print('Value of b:', b)
print('Value of c:', c) # list is created

a, *b, c = 1, 2, 3, 4, 5 # 'b' is assigned a list with the remaining values
print(nl, 'Using -> a, *b, c = 1, 2, 3, 4, 5')
print('Value of a:', a)
print('Value of b:', b)
print('Value of c:', c)

*a, b, c = 1, 2, 3, 4, 5 # 'a' is assigned a list with the remaining values
print(nl, 'Using -> *a, b, c = 1, 2, 3, 4, 5')
print('Value of a:', a)
print('Value of b:', b)
print('Value of c:', c)

print('So far, the Extended Iterable Unpacking has only beeen done on tuples, but any iterable sequence can use this tool')

a, b, *c = 'hello'
print(nl, 'Using -> a, b, *c = "hello", Extended Iterable Unpacking over a string')
print('Value of a:', a)
print('Value of b:', b)
print('Value of c:', c)

'Extended Iterable Unpacking over a dictionaries iterate over the keys'
first, *last = {1:'a', 2:'b', 3:'c', 4:'d'} # last has the *asterisk
print(nl, 'Using -> first, *last = {1:\'a\', 2:\'b\', 3:\'c\', 4:\'d\'}, Extended Iterable Unpacking over a dictionary')
print('Value of first:', first) # gets the key of the first dictionary item
print('Value of last:', last) # last has the remaining items in the dictionary in a list
'''
RESULT:
```


EXPLORER
OPEN EDITORS 1 UNSAVED
percipio56_Implement an Iterable Using Extended iter()...
PYTHON
Automate-Boring-Stuff
my_code
Percipio_Python3-Course
01_Start
02_Data-Sequence Types
03_Collections-Mapping-Looping
04_Modules-Functions
05_Classes
06_Working-with-Files
07_Comprehensions
08_Iterables-and-Generators
percipio50_Basic Iteration.py
percipio51_The map() Function.py
percipio52_The Filter() Function.py
percipio53_The functools.reduce() Function.py
percipio54_Implementing an Iterator.py
percipio55_Implement an Iterable Using __getItem_.py
percipio56_Implement an Iterable Using Extended iter()...
percipio57_Simple Generators.py
percipio58_Lazy Generators.py
percipio59_Recursive Generators.py
percipio60_Exercise-Creating an Iterable Data Type.py
09_Exceptions
10_Automation Programming
Python Projects_2014
CMD_Python_Set-Path.txt
Python_Clear-Window-Command.txt
python_debug_logging_code.py
python_exercises_00.py
python_exercises_01.py
Python_Tutorial_Running-Scripts.docx
percipio56_Implement an Iterable Using Extended iter().py

```
RESULT:
Assign values to variables without using Extended Iterable Unpacking
Value of a: 1
Value of b: 2
Value of c: 3
Handled ValueError: too many values to unpack (expected 3)

 Demonstrations of an *asterisk by a variable
 Using -> a, b, *c = 1, 2, 3, 4
Value of a: 1
Value of b: 2
Value of c: [3, 4]
 Using -> a, *b, c = 1, 2, 3, 4, 5
Value of a: 1
Value of b: [2, 3, 4]
Value of c: 5
 Using -> *a, b, c = 1, 2, 3, 4, 5
Value of a: [1, 2, 3]
Value of b: 4
Value of c: 5

So far, the Extended Iterable Unpacking has only beeen done on tuples, but any iterable sequence can use this tool

 Using -> a, b, *c = "hello", Extended Iterable Unpacking over a string
Value of a: h
Value of b: e
Value of c: ['l', 'l', 'o']

 Using -> first, *last = {1:'a', 2:'b', 3:'c', 4:'d'}, Extended Iterable Unpacking over a dictionary
Value of first: 1
Value of last: [2, 3, 4]
'''
```