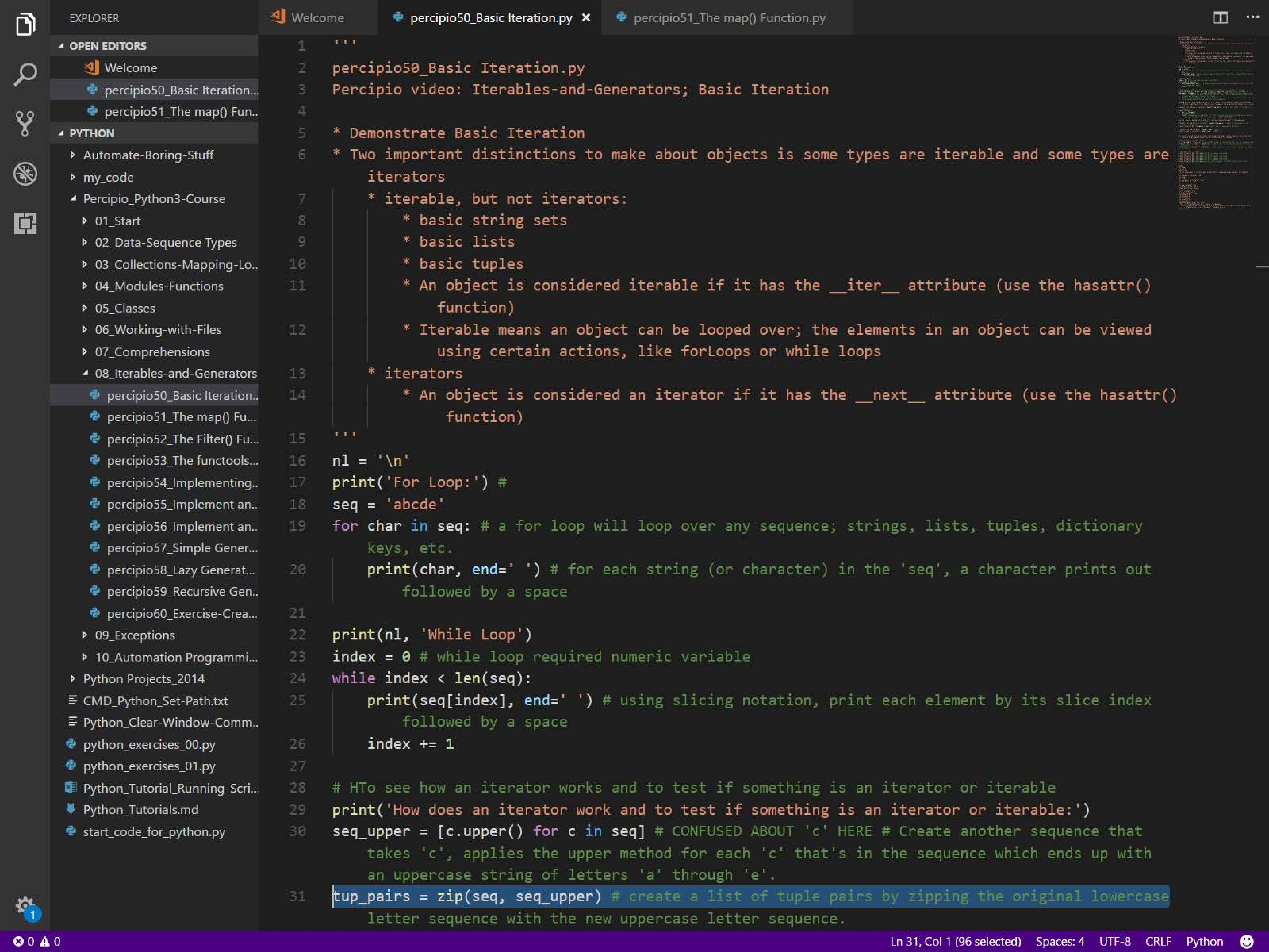
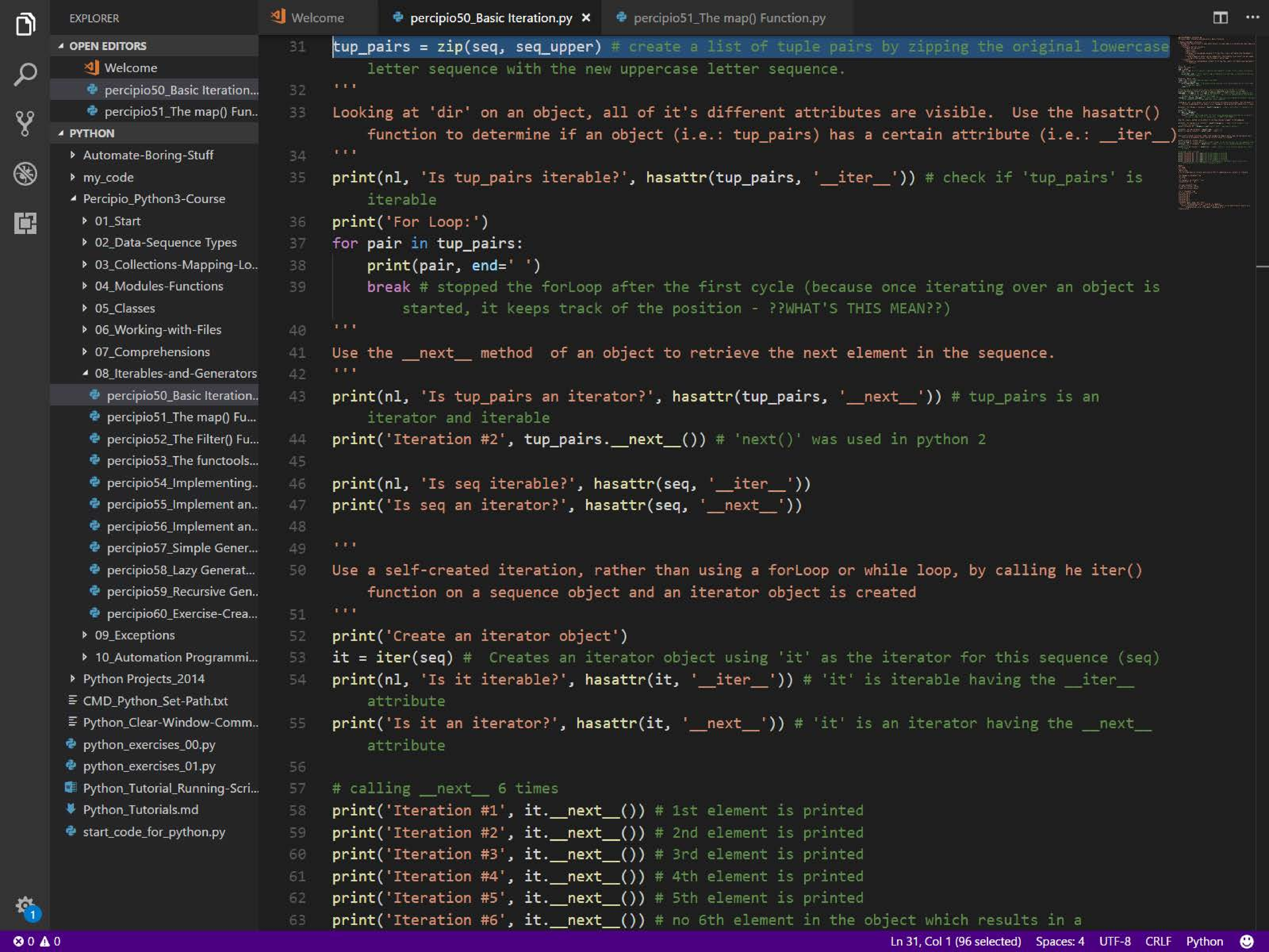

EXPLORER
Welcome
percipio50_Basic Iteration.py
percipio51_The map() Function.py
OPEN EDITORS
Welcome
percipio50_Basic Iteration...
percipio51_The map() Fun...
PYTHON
Automate-Boring-Stuff
my_code
Percipio_Python3-Course
01_Start
02_Data-Sequence Types
03_Collections-Mapping-Lo...
04_Modules-Functions
05_Classes
06_Working-with-Files
07_Comprehensions
08_Iterables-and-Generators
percipio50_Basic Iteration...
percipio51_The map() Fu...
percipio52_The Filter() Fu...
percipio53_The functools...
percipio54_Implementing...
percipio55_Implement an...
percipio56_Implement an...
percipio57_Simple Gener...
percipio58_Lazy Generat...
percipio59_Recursive Gen...
percipio60_Exercise-Crea...
09_Exceptions
10_Automation Programmi...
Python Projects_2014
CMD_Python_Set-Path.txt
Python_Clear-Window-Comm...
python_exercises_00.py
python_exercises_01.py
Python_Tutorial_Running-Scri...
Python_Tutorials.md
start_code_for_python.py
1
0 0
Ln 31, Col 1 (96 selected)
Spaces: 4
UTF-8
CRLF
Python

```python
'''
percipio50_Basic Iteration.py
Percipio video: Iterables-and-Generators; Basic Iteration

* Demonstrate Basic Iteration
* Two important distinctions to make about objects is some types are iterable and some types are iterators
    * iterable, but not iterators:
        * basic string sets
        * basic lists
        * basic tuples
        * An object is considered iterable if it has the __iter__ attribute (use the hasattr() function)
        * Iterable means an object can be looped over; the elements in an object can be viewed using certain actions, like forLoops or while loops
    * iterators
        * An object is considered an iterator if it has the __next__ attribute (use the hasattr() function)
'''
nl = '\n'
print('For Loop:') #
seq = 'abcde'
for char in seq: # a for loop will loop over any sequence; strings, lists, tuples, dictionary keys, etc.
    print(char, end=' ') # for each string (or character) in the 'seq', a character prints out followed by a space

print(nl, 'While Loop')
index = 0 # while loop required numeric variable
while index < len(seq):
    print(seq[index], end=' ') # using slicing notation, print each element by its slice index followed by a space
    index += 1

# HTo see how an iterator works and to test if something is an iterator or iterable
print('How does an iterator work and to test if something is an iterator or iterable:')
seq_upper = [c.upper() for c in seq] # CONFUSED ABOUT 'c' HERE # Create another sequence that takes 'c', applies the upper method for each 'c' that's in the sequence which ends up with an uppercase string of letters 'a' through 'e'.
tup_pairs = zip(seq, seq_upper) # create a list of tuple pairs by zipping the original lowercase letter sequence with the new uppercase letter sequence.
```

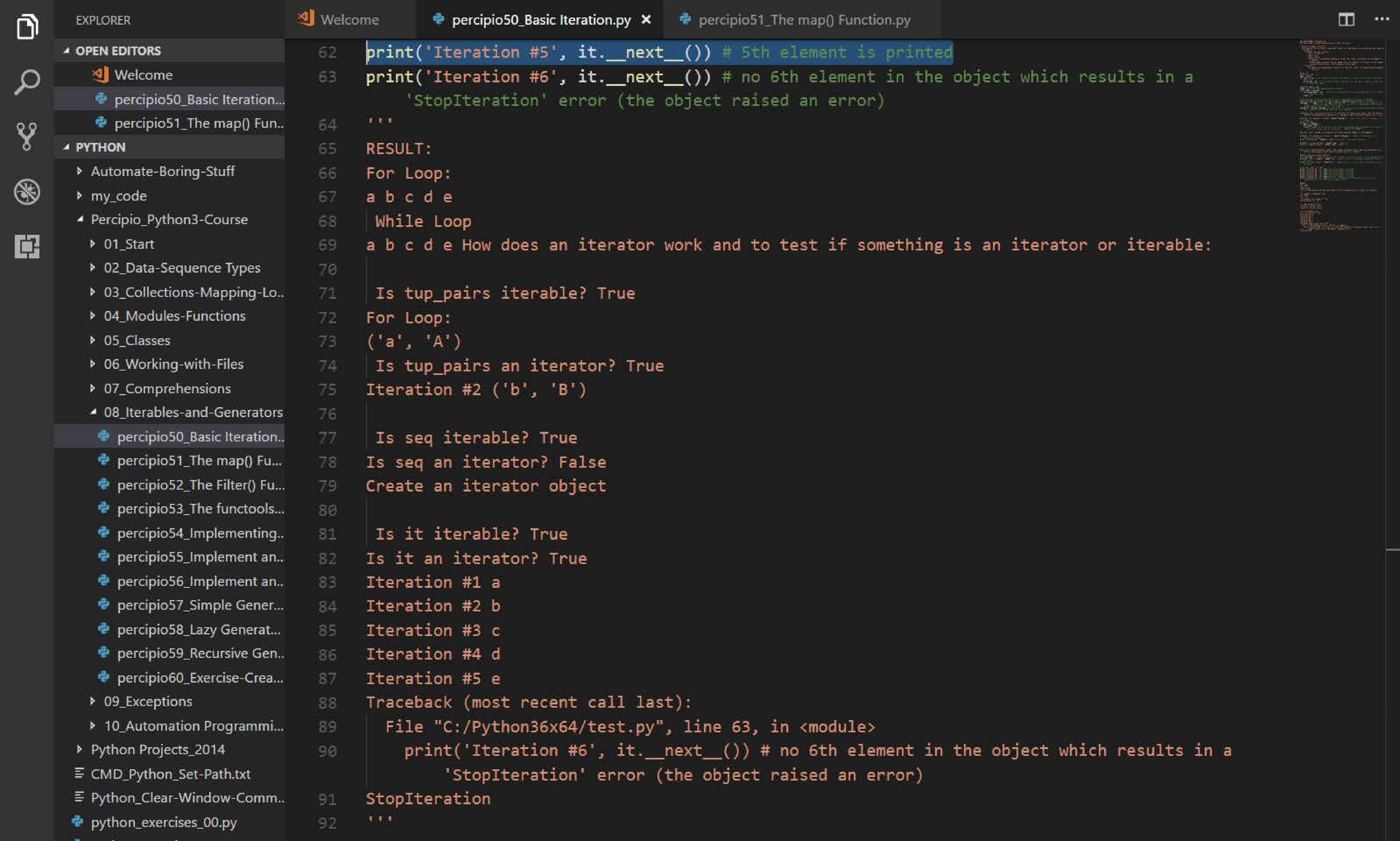

EXPLORER
Welcome
percipio50_Basic Iteration.py
percipio51_The map() Function.py
OPEN EDITORS
Welcome
percipio50_Basic Iteration...
percipio51_The map() Fun...
PYTHON
Automate-Boring-Stuff
my_code
Percipio_Python3-Course
01_Start
02_Data-Sequence Types
03_Collections-Mapping-Lo...
04_Modules-Functions
05_Classes
06_Working-with-Files
07_Comprehensions
08_Iterables-and-Generators
percipio50_Basic Iteration...
percipio51_The map() Fu...
percipio52_The Filter() Fu...
percipio53_The functools...
percipio54_Implementing...
percipio55_Implement an...
percipio56_Implement an...
percipio57_Simple Gener...
percipio58_Lazy Generat...
percipio59_Recursive Gen...
percipio60_Exercise-Crea...
09_Exceptions
10_Automation Programmi...
Python Projects_2014
CMD_Python_Set-Path.txt
Python_Clear-Window-Comm...
python_exercises_00.py
python_exercises_01.py
Python_Tutorial_Running-Scri...
Python_Tutorials.md
start_code_for_python.py
1
ⓧ 0 ⚠ 0
Ln 31, Col 1 (96 selected)
Spaces: 4
UTF-8
CRLF
Python

```
tup_pairs = zip(seq, seq_upper) # create a list of tuple pairs by zipping the original lowercase letter sequence with the new uppercase letter sequence.
'''
Looking at 'dir' on an object, all of it's different attributes are visible.  Use the hasattr() function to determine if an object (i.e.: tup_pairs) has a certain attribute (i.e.: __iter__)
'''
print(nl, 'Is tup_pairs iterable?', hasattr(tup_pairs, '__iter__')) # check if 'tup_pairs' is iterable
print('For Loop:')
for pair in tup_pairs:
    print(pair, end=' ')
    break # stopped the forLoop after the first cycle (because once iterating over an object is started, it keeps track of the position - ??WHAT'S THIS MEAN??)
'''
Use the __next__ method  of an object to retrieve the next element in the sequence.
'''
print(nl, 'Is tup_pairs an iterator?', hasattr(tup_pairs, '__next__')) # tup_pairs is an iterator and iterable
print('Iteration #2', tup_pairs.__next__()) # 'next()' was used in python 2

print(nl, 'Is seq iterable?', hasattr(seq, '__iter__'))
print('Is seq an iterator?', hasattr(seq, '__next__'))

'''
Use a self-created iteration, rather than using a forLoop or while loop, by calling he iter() function on a sequence object and an iterator object is created
'''
print('Create an iterator object')
it = iter(seq) #  Creates an iterator object using 'it' as the iterator for this sequence (seq)
print(nl, 'Is it iterable?', hasattr(it, '__iter__')) # 'it' is iterable having the __iter__ attribute
print('Is it an iterator?', hasattr(it, '__next__')) # 'it' is an iterator having the __next__ attribute

# calling __next__ 6 times
print('Iteration #1', it.__next__()) # 1st element is printed
print('Iteration #2', it.__next__()) # 2nd element is printed
print('Iteration #3', it.__next__()) # 3rd element is printed
print('Iteration #4', it.__next__()) # 4th element is printed
print('Iteration #5', it.__next__()) # 5th element is printed
print('Iteration #6', it.__next__()) # no 6th element in the object which results in a
```


EXPLORER
OPEN EDITORS
Welcome
percipio50_Basic Iteration...
percipio51_The map() Fun...
PYTHON
Automate-Boring-Stuff
my_code
Percipio_Python3-Course
01_Start
02_Data-Sequence Types
03_Collections-Mapping-Lo...
04_Modules-Functions
05_Classes
06_Working-with-Files
07_Comprehensions
08_Iterables-and-Generators
percipio50_Basic Iteration...
percipio51_The map() Fu...
percipio52_The Filter() Fu...
percipio53_The functools...
percipio54_Implementing...
percipio55_Implement an...
percipio56_Implement an...
percipio57_Simple Gener...
percipio58_Lazy Generat...
percipio59_Recursive Gen...
percipio60_Exercise-Crea...
09_Exceptions
10_Automation Programmi...
Python Projects_2014
CMD_Python_Set-Path.txt
Python_Clear-Window-Comm...
python_exercises_00.py
Welcome
percipio50_Basic Iteration.py
percipio51_The map() Function.py

```
print('Iteration #5', it.__next__()) # 5th element is printed
print('Iteration #6', it.__next__()) # no 6th element in the object which results in a 'StopIteration' error (the object raised an error)
'''
RESULT:
For Loop:
a b c d e
 While Loop
a b c d e How does an iterator work and to test if something is an iterator or iterable:

 Is tup_pairs iterable? True
For Loop:
('a', 'A')
 Is tup_pairs an iterator? True
Iteration #2 ('b', 'B')

 Is seq iterable? True
Is seq an iterator? False
Create an iterator object

 Is it iterable? True
Is it an iterator? True
Iteration #1 a
Iteration #2 b
Iteration #3 c
Iteration #4 d
Iteration #5 e
Traceback (most recent call last):
  File "C:/Python36x64/test.py", line 63, in <module>
    print('Iteration #6', it.__next__()) # no 6th element in the object which results in a 'StopIteration' error (the object raised an error)
StopIteration
'''
```