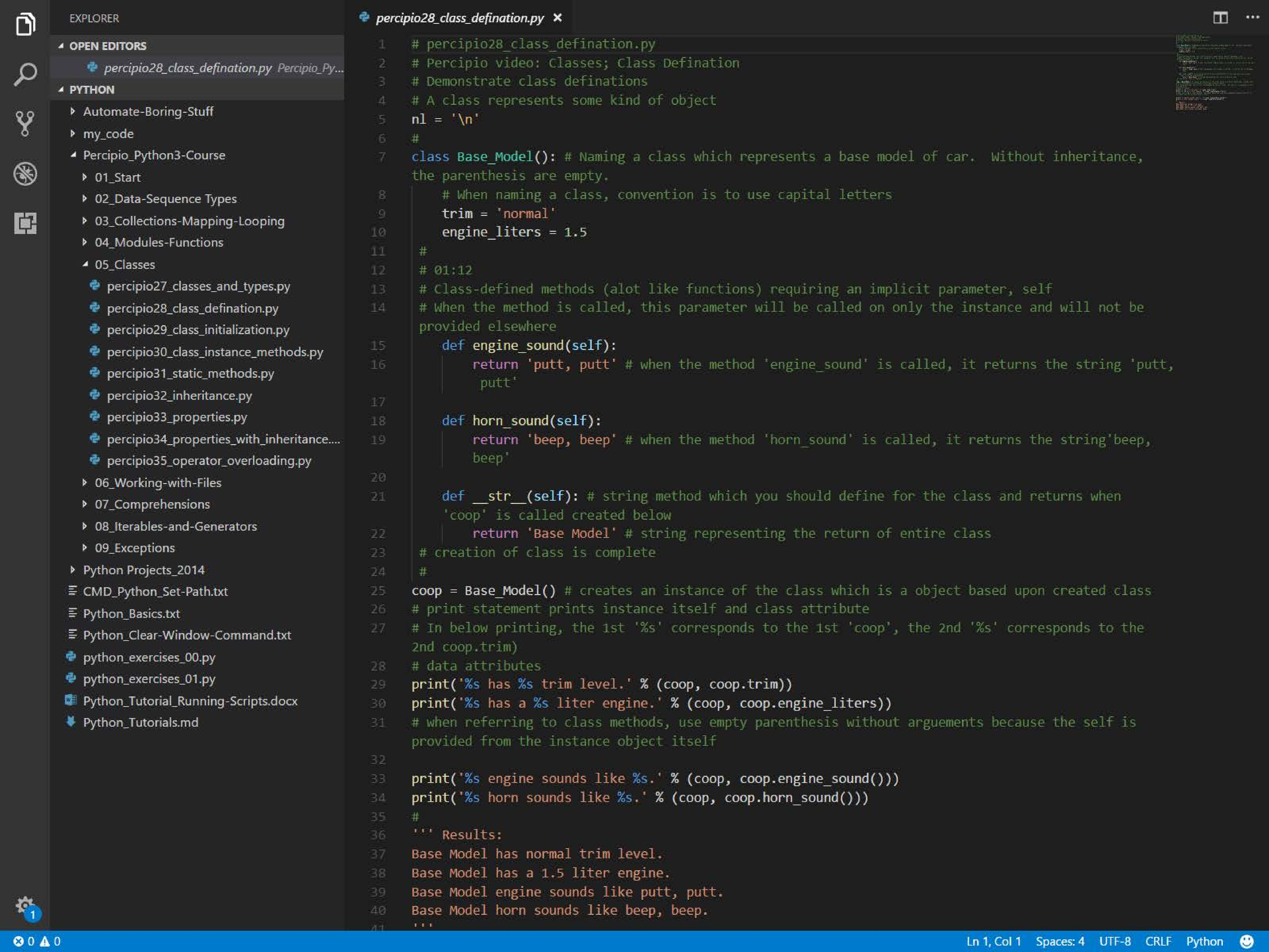
EXPLORER
OPEN EDITORS
percipio28_class_defination.py Percipio_Py...
PYTHON
Automate-Boring-Stuff
my_code
Percipio_Python3-Course
01_Start
02_Data-Sequence Types
03_Collections-Mapping-Looping
04_Modules-Functions
05_Classes
percipio27_classes_and_types.py
percipio28_class_defination.py
percipio29_class_initialization.py
percipio30_class_instance_methods.py
percipio31_static_methods.py
percipio32_inheritance.py
percipio33_properties.py
percipio34_properties_with_inheritance....
percipio35_operator_overloading.py
06_Working-with-Files
07_Comprehensions
08_Iterables-and-Generators
09_Exceptions
Python Projects_2014
CMD_Python_Set-Path.txt
Python_Basics.txt
Python_Clear-Window-Command.txt
python_exercises_00.py
python_exercises_01.py
Python_Tutorial_Running-Scripts.docx
Python_Tutorials.md
percipio28_class_defination.py
0 0
Ln 1, Col 1
Spaces: 4
UTF-8
CRLF
Python

```python
# percipio28_class_defination.py
# Percipio video: Classes; Class Defination
# Demonstrate class definations
# A class represents some kind of object
nl = '\n'
#
class Base_Model(): # Naming a class which represents a base model of car.  Without inheritance, the parenthesis are empty.
    # When naming a class, convention is to use capital letters
    trim = 'normal'
    engine_liters = 1.5
 #
 # 01:12
 # Class-defined methods (alot like functions) requiring an implicit parameter, self
 # When the method is called, this parameter will be called on only the instance and will not be provided elsewhere
    def engine_sound(self):
        return 'putt, putt' # when the method 'engine_sound' is called, it returns the string 'putt, putt'

    def horn_sound(self):
        return 'beep, beep' # when the method 'horn_sound' is called, it returns the string'beep, beep'

    def __str__(self): # string method which you should define for the class and returns when 'coop' is called created below
        return 'Base Model' # string representing the return of entire class
 # creation of class is complete
 #
coop = Base_Model() # creates an instance of the class which is a object based upon created class
# print statement prints instance itself and class attribute
# In below printing, the 1st '%s' corresponds to the 1st 'coop', the 2nd '%s' corresponds to the 2nd coop.trim)
# data attributes
print('%s has %s trim level.' % (coop, coop.trim))
print('%s has a %s liter engine.' % (coop, coop.engine_liters))
# when referring to class methods, use empty parenthesis without arguements because the self is provided from the instance object itself

print('%s engine sounds like %s.' % (coop, coop.engine_sound()))
print('%s horn sounds like %s.' % (coop, coop.horn_sound()))
#
''' Results:
Base Model has normal trim level.
Base Model has a 1.5 liter engine.
Base Model engine sounds like putt, putt.
Base Model horn sounds like beep, beep.
'''
```