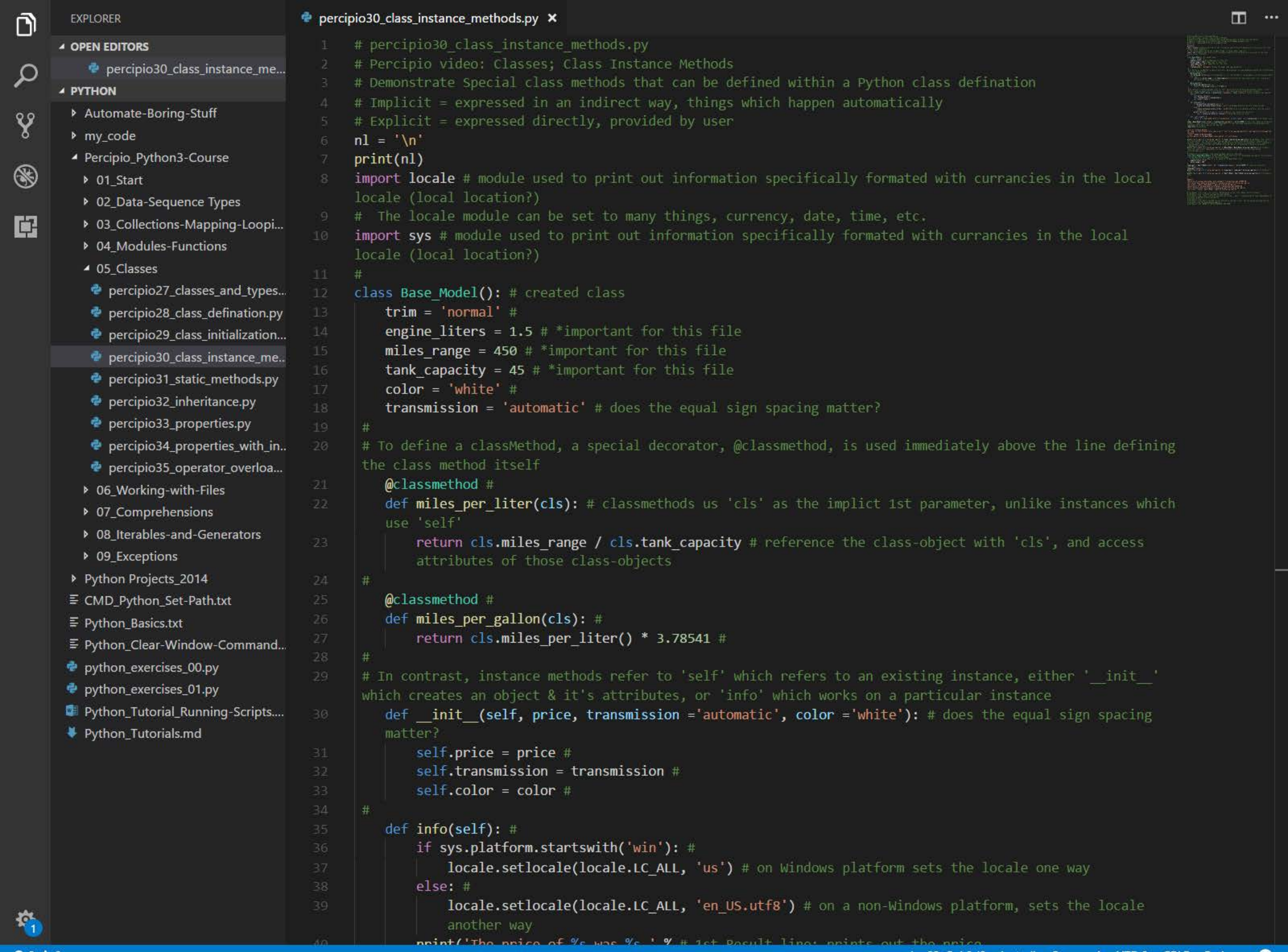


EXPLORER

OPEN EDITORS
- percipio30_class_instance_me...

PYTHON
- Automate-Boring-Stuff
- my_code
- Percipio_Python3-Course
  - 01_Start
  - 02_Data-Sequence Types
  - 03_Collections-Mapping-Loopi...
  - 04_Modules-Functions
  - 05_Classes
    - percipio27_classes_and_types...
    - percipio28_class_defination.py
    - percipio29_class_initialization...
    - percipio30_class_instance_me...
    - percipio31_static_methods.py
    - percipio32_inheritance.py
    - percipio33_properties.py
    - percipio34_properties_with_in...
    - percipio35_operator_overloa...
  - 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...
- python_exercises_00.py
- python_exercises_01.py
- Python_Tutorial_Running-Scripts....
- Python_Tutorials.md

percipio30_class_instance_methods.py

```
# percipio30_class_instance_methods.py
# Percipio video: Classes; Class Instance Methods
# Demonstrate Special class methods that can be defined within a Python class defination
# Implicit = expressed in an indirect way, things which happen automatically
# Explicit = expressed directly, provided by user
nl = '\n'
print(nl)
import locale # module used to print out information specifically formated with currancies in the local locale (local location?)
#  The locale module can be set to many things, currency, date, time, etc.
import sys # module used to print out information specifically formated with currancies in the local locale (local location?)
#
class Base_Model(): # created class
    trim = 'normal' #
    engine_liters = 1.5 # *important for this file
    miles_range = 450 # *important for this file
    tank_capacity = 45 # *important for this file
    color = 'white' #
    transmission = 'automatic' # does the equal sign spacing matter?
#
# To define a classMethod, a special decorator, @classmethod, is used immediately above the line defining the class method itself
    @classmethod #
    def miles_per_liter(cls): # classmethods us 'cls' as the implict 1st parameter, unlike instances which use 'self'
        return cls.miles_range / cls.tank_capacity # reference the class-object with 'cls', and access attributes of those class-objects
#
    @classmethod #
    def miles_per_gallon(cls): #
        return cls.miles_per_liter() * 3.78541 #
#
# In contrast, instance methods refer to 'self' which refers to an existing instance, either '__init__' which creates an object & it's attributes, or 'info' which works on a particular instance
    def __init__(self, price, transmission ='automatic', color ='white'): # does the equal sign spacing matter?
        self.price = price #
        self.transmission = transmission #
        self.color = color #
#
    def info(self): #
        if sys.platform.startswith('win'): #
            locale.setlocale(locale.LC_ALL, 'us') # on Windows platform sets the locale one way
        else: #
            locale.setlocale(locale.LC_ALL, 'en_US.utf8') # on a non-Windows platform, sets the locale another way
        print('The price of %s was %s.' % # 1st Result line: prints out the price
```

EXPLORER

OPEN EDITORS

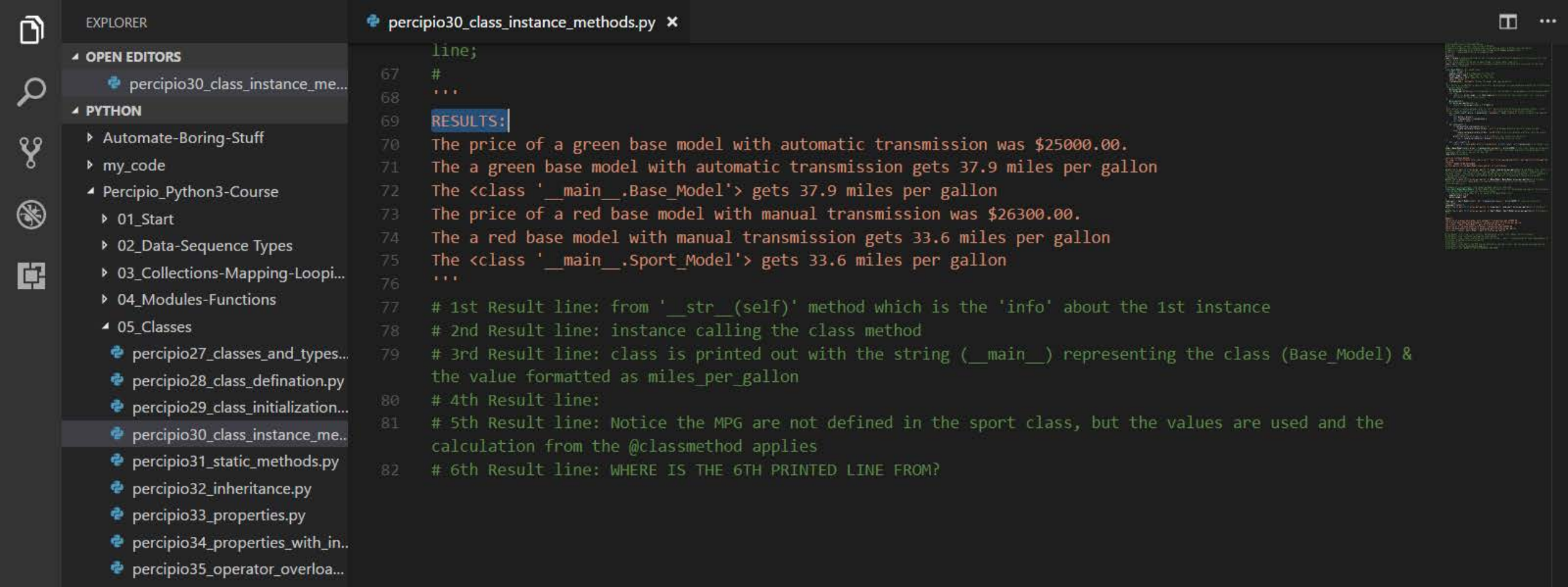
percipio30_class_instance_me...

PYTHON

- Automate-Boring-Stuff
- my_code
- Percipio_Python3-Course
  - 01_Start
  - 02_Data-Sequence Types
  - 03_Collections-Mapping-Loopi...
  - 04_Modules-Functions
  - 05_Classes
    - percipio27_classes_and_types...
    - percipio28_class_defination.py
    - percipio29_class_initialization...
    - percipio30_class_instance_me...
    - percipio31_static_methods.py
    - percipio32_inheritance.py
    - percipio33_properties.py
    - percipio34_properties_with_in...
    - percipio35_operator_overloa...
  - 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...
- python_exercises_00.py
- python_exercises_01.py
- Python_Tutorial_Running-Scripts....
- Python_Tutorials.md

percipio30_class_instance_methods.py

```
38          else: #
39              locale.setlocale(locale.LC_ALL, 'en_US.utf8') # on a non-Windows platform, sets the locale
                another way
40          print('The price of %s was %s.' % # 1st Result line; prints out the price,...
41              (self, locale.currency(self.price))) # using the locale-currency
42  #
43      def __str__(self): #
44          return 'a %s base model with %s transmission' % (self.color, self.transmission) # 1st Result line
45  #
46  coop = Base_Model(color='green', transmission='automatic', price=25000) # This line creates an instance of
    objects.  'ccop' is instance variable, 'Base_Model' is the class name, and within the paranthesis are
    arguements which may or may not be required.
47  coop.info() # see below
48  '''
49  calls an instance method
50  So 'coop' is the instance referring to self. 'self is not passed explicitly, but implicitily through the
    instance.
51  'info()' NEEDS TO BE EXPLAINED
52  So this points to the Base_Model class and all it's attributes.
53  '''
54  print('The %s gets %4.1f miles per gallon' % (coop, coop.miles_per_gallon())) # 2nd Result line; refers to
    the class method, not the instance. 'miles_per_gallon()' is the class method which is used with the
    instance (coop).  the instance (coop) gets formatted with the special string method. '%4.1f' indicates a
    floating-point-value from the class method (miles_per_gallon) is formatted 4-characters wide with
    1-decimal of precision
55  print('The %s gets %4.1f miles per gallon' % (Base_Model, Base_Model.miles_per_gallon())) # 3rd Result
    line; The class itself (Base_Model) is used as an objects on which to call the class method
    (miles_per_gallon())
56  #
57  # Example of class methods, like normal methods, which is inherited
58  class Sport_Model(Base_Model): # Sport_Model which inherits all of the methods and some of the attributes
    from the Base_Model with some attributes overridden
59      # Represent a sport model of a car based on the Base_Model class
60      engine_liters = 2.0
61      miles_range = 400
62  #
63  coop_sport = Sport_Model(color='red', transmission='manual', price=26300) # create an instance of
    Sport_Model
64  coop_sport.info() #
65  print('The %s gets %4.1f miles per gallon' % (coop_sport, coop_sport.miles_per_gallon())) # 4th Result
    line;
66  print('The %s gets %4.1f miles per gallon' % (Sport_Model, Sport_Model.miles_per_gallon())) # 5th Result
    line;
67  #
68  '''
69  RESULTS:
70  The price of a green base model with automatic transmission was $25000.00.
```


EXPLORER
percipio30_class_instance_methods.py
OPEN EDITORS
percipio30_class_instance_me...
PYTHON
Automate-Boring-Stuff
my_code
Percipio_Python3-Course
01_Start
02_Data-Sequence Types
03_Collections-Mapping-Loopi...
04_Modules-Functions
05_Classes
percipio27_classes_and_types...
percipio28_class_defination.py
percipio29_class_initialization...
percipio30_class_instance_me...
percipio31_static_methods.py
percipio32_inheritance.py
percipio33_properties.py
percipio34_properties_with_in...
percipio35_operator_overloa...
line;
#
'''
RESULTS:
The price of a green base model with automatic transmission was $25000.00.
The a green base model with automatic transmission gets 37.9 miles per gallon
The <class '__main__.Base_Model'> gets 37.9 miles per gallon
The price of a red base model with manual transmission was $26300.00.
The a red base model with manual transmission gets 33.6 miles per gallon
The <class '__main__.Sport_Model'> gets 33.6 miles per gallon
'''
1st Result line: from '__str__(self)' method which is the 'info' about the 1st instance
2nd Result line: instance calling the class method
3rd Result line: class is printed out with the string (__main__) representing the class (Base_Model) &
the value formatted as miles_per_gallon
4th Result line:
5th Result line: Notice the MPG are not defined in the sport class, but the values are used and the
calculation from the @classmethod applies
6th Result line: WHERE IS THE 6TH PRINTED LINE FROM?