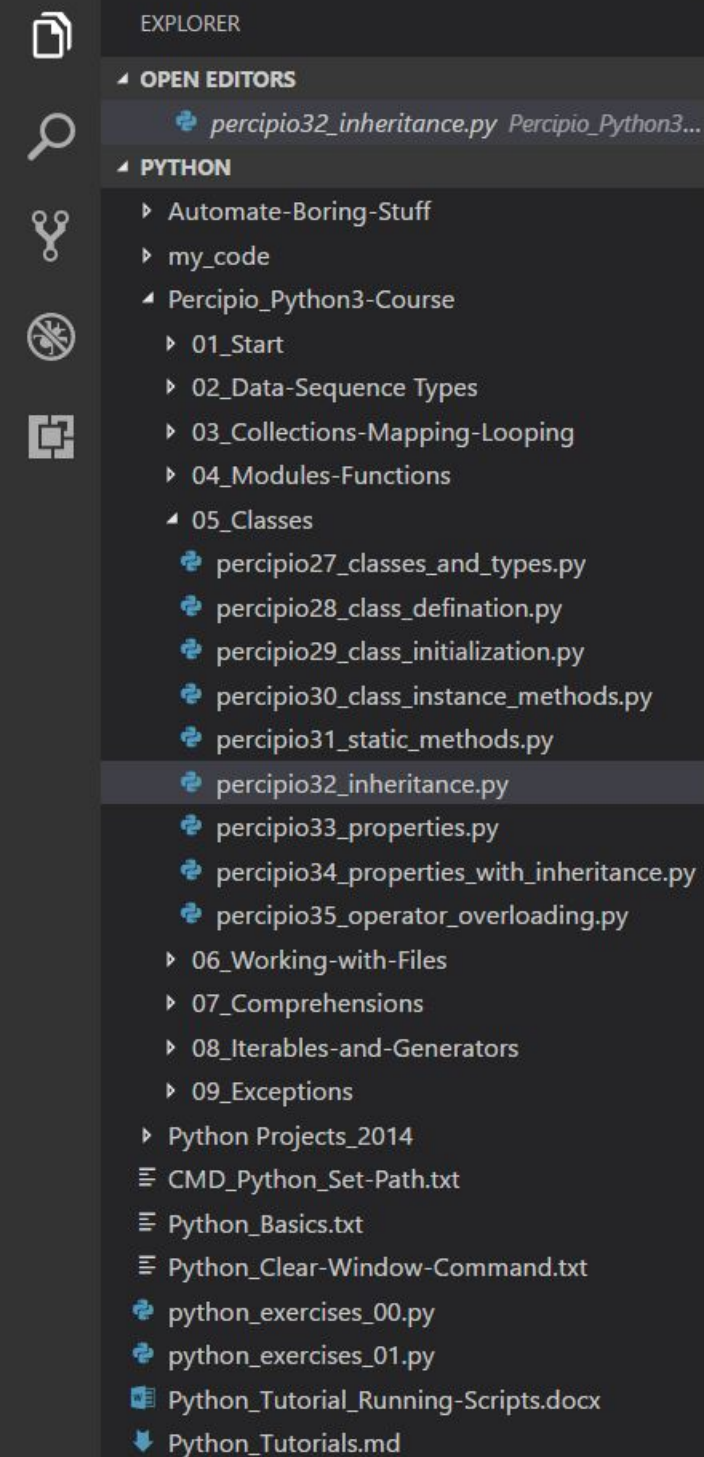
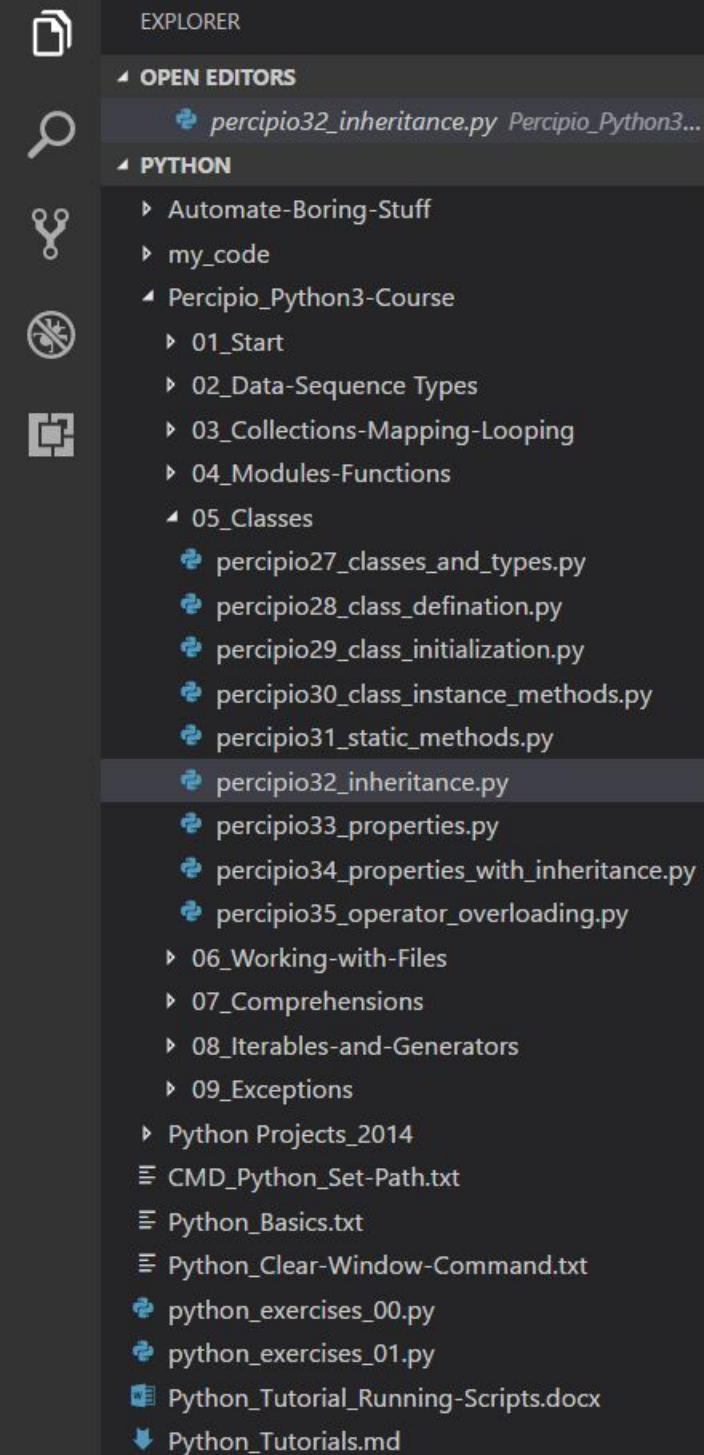


```
percipio32_inheritance.py x
1  # percipio32_inheritance.py
2  # Percipio video: Classes; Inheritance
3  # Demonstrate how inheritance works in Python classes
4  #
5  nl = '\n'
6  # 1)
7  print(nl, 'Base_Model')
8  class Base_Model(): # create a class called 'Base_Model'
9      # represent base model of a car
10     trim = 'normal' # Base_Model trim
11     engine_liters = 1.5 # Base_Model engine size
12
13     def engine_sound(self): #
14         return 'putt, putt' # Base_Model engine sound
15
16     def horn_sound(self): #
17         return 'beep, beep' # Base_Model Horn sound
18
19     def __str__(self): # special method used to print the instance of this class
20         return 'Base Model' # Base_Model
21
22     coop = Base_Model() # creates an instance of the Base_Model class
23     print('%s has %s trim level.' % (coop, coop.trim)) # the 1st '%s' is the string of the instance,
24     # the 1st 'coop' which calls automatically on the special method, __str__, returning Base_Model.
25     # The 2nd '%s' has the class attribute, trim, substituted in it's place.
26     print('%s has a %s liter engine.' % (coop, coop.engine_liters)) #
27     print('%s engine sounds like %s.' % (coop, coop.engine_sound())) #
28     print('%s horn sounds like %s.' % (coop, coop.horn_sound())) #
29     #
30     # Above is a basic class defination with basic class attributes and methods being used
31     #
32     print('Sport_Model')
33     # 2)
34     # Notice the next Sport_Model class has Base_Model in it's parentheses so it automatically
35     # inherits all of the attributes of the Base_Model
36     class Sport_Model(Base_Model): #
37         engine_liters = 2.0 # this overrides Base_Model engine_liters but trim is herited since it is
38         # not here
39
40         def engine_sound(self): #
41             return 'VROOM, VROOM' #
42
43         def horn_sound(self): #
```

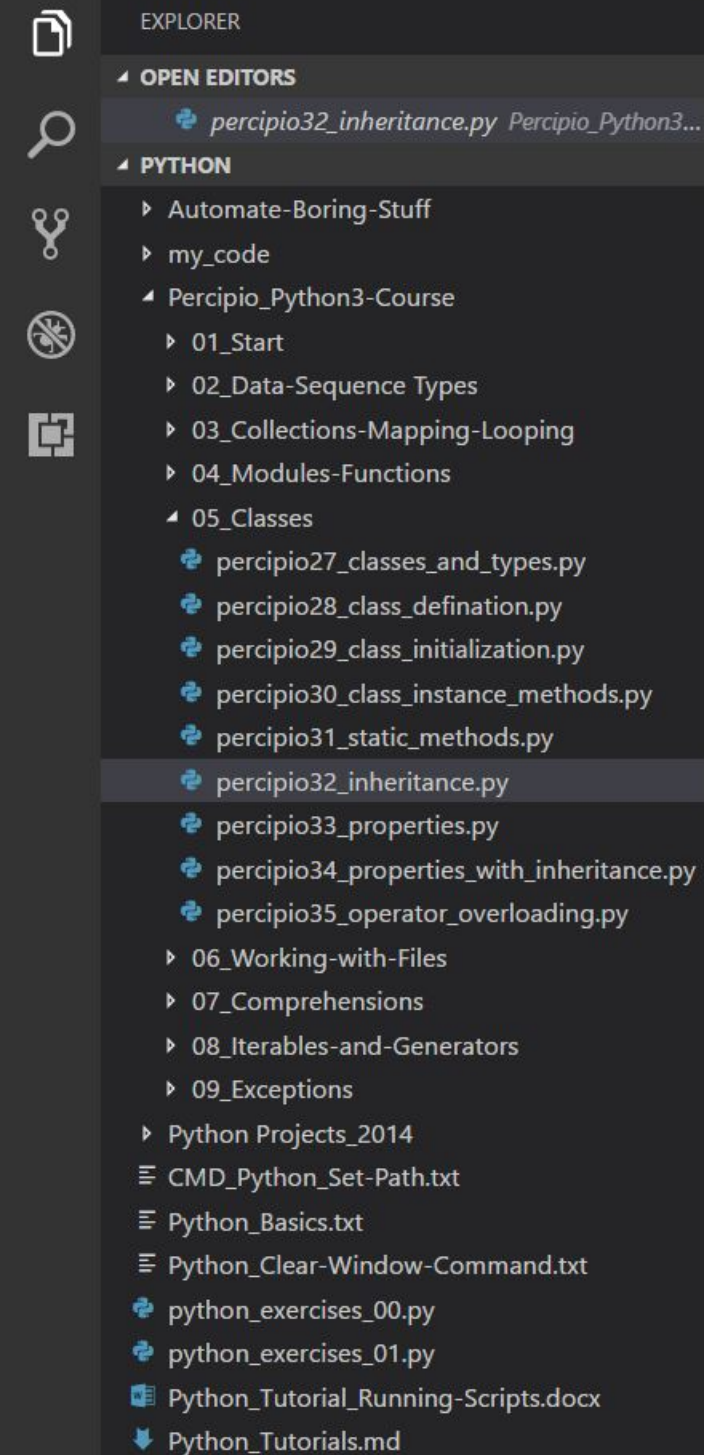



percipio32_inheritance.py x

```
39 def horn_sound(self): #
40     return 'BEEP, BEEP' #
41
42 def __str__(self): #
43     return 'Sport Model' # returns a different string of the instance
44
45 coop_sport = Sport_Model() #
46 print('%s has %s trim level.' % (coop_sport, coop_sport.trim)) #
47 print('%s has a %s liter engine.' % (coop_sport, coop_sport.engine_liters)) #
48 print('%s engine sounds like %s.' % (coop_sport, coop_sport.engine_sound())) #
49 print('%s horn sounds like %s.' % (coop_sport, coop_sport.horn_sound())) #
50 #
51 print(nl, 'Luxury_Sport_Model')
52 # 3)
53 class Luxury_Model(Base_Model): # class also inherits all attributes of the Base_Model
54     trim = 'luxury' # trim is overridden
55
56     def engine_sound(self): #
57         return 'vroom, vroom' #
58
59     def horn_sound(self): #
60         return 'honk, honk' #
61
62     def __str__(self): #
63         return 'Luxury Model' #
64 # This class (Luxury_Model) is not instantiated directly but it's used in the 'Luxury_Sport_Model'
65 # showing inheritance from more than one object
66 # The order within the parenthesis is important with the 1st attribute having primary importance.
67 # (i.e.(Luxury_Model, Sport_Model) with Luxury_Model having the 1st attribute chosen by Python)
68 class Luxury_Sport_Model(Luxury_Model, Sport_Model): #
69     def __str__(self): #
70         return 'Luxury Sport Model' # Only thing overridden by this 'Luxury_Sport_Model' class is
71         the rendering of the str function, __str__, also changing the string and how an instance
72         of this class prints.
73
74 coop_luxury_sport = Luxury_Sport_Model() # creating an instance
75 print('%s has %s trim level.' % (coop_luxury_sport, coop_luxury_sport.trim)) #
76 print('%s has a %s liter engine.' % (coop_luxury_sport, coop_luxury_sport.engine_liters)) #
77 print('%s engine sounds like %s.' % (coop_luxury_sport, coop_luxury_sport.engine_sound())) #
78 print('%s horn sounds like %s.' % (coop_luxury_sport, coop_luxury_sport.horn_sound())) #
79 #
80 # 4)
81 print(nl, 'Sport_Luxury_Model')
```

```
percipio32_inheritance.py x
77 print(nl, 'Sport_Luxury_Model')
78 class Sport_Luxury_Model(Sport_Model, Luxury_Model): # To further demonstrate inheritacne and the
# importance of the order within the parenthesis, this class reverses (Luxury_Model, Sport_Model) to
# (Sport_Model, Luxury_Model). Notice the Sport_Model attribues having priority in the results.
79     def __str__(self): #
80         return 'Sport Luxury Model' #
81
82 coop_sport_luxury = Sport_Luxury_Model() #
83 print('%s has %s trim level.' % (coop_sport_luxury, coop_sport_luxury.trim)) #
84 print('%s has a %s liter engine.' % (coop_sport_luxury, coop_sport_luxury.engine_liters)) #
85 print('%s engine sounds like %s.' % (coop_sport_luxury, coop_sport_luxury.engine_sound())) #
86 print('%s horn sounds like %s.' % (coop_sport_luxury, coop_sport_luxury.horn_sound())) #
87 #
88 # All of the above attributes are defined as part of the class itself
89 # 5) Working with custom attributes
90 # Take an instance and modify the attributes that it has, which will then be different only for
# that instance. The class will still maintain its original attributes.
91 print(nl, 'Custom_Models')
92 coop_custom = Sport_Luxury_Model() #
93 print('%s has %s trim level.' % (coop_custom, coop_custom.trim)) #
94 coop_custom.trim = 'custom' #
95 print('%s now has %s trim level.' % (coop_custom, coop_custom.trim)) # prints what the instance
# coop_custom trim is
96 print('The class %s still has %s trim level.' % (Sport_Luxury_Model, Sport_Luxury_Model.trim)) #
# The trim is coming still from the class itself, not from the modified custom instance
97 coop_custom.brakes = 'racing' # add extra attributes to an instance
98 Base_Model.brakes = 'standard' # the Base_Model class adds an attribute
99 print('%s has %s brakes.' % (coop_custom, coop_custom.brakes)) # prints what the instance
# Sport_Luxury_Model brakes are
100 print('%s has %s brakes.' % (Sport_Luxury_Model, Sport_Luxury_Model.brakes)) # prints what the
# class Sport_Luxury_Model brakes are. Notice above the brakes were set for the Base_Model only,
# but through inheritance, the Sport_Luxury_Model has it's brakes set to standard
101 #
102 ...
103 RESULT:
104 ===== RESTART: C:/Python36x64/test.py =====
105 Base_Model
106 Base Model has normal trim level.
107 Base Model has a 1.5 liter engine.
108 Base Model engine sounds like putt, putt.
109 Base Model horn sounds like beep, beep.
110
111 Sport Model
```

EXPLORER

OPEN EDITORS

- percipio32_inheritance.py Percipio_Python3...

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.py
 - 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

percipio32_inheritance.py x

```
101 #
102 ...
103 RESULT:
104 ===== RESTART: C:/Python36x64/test.py =====
105 Base_Model
106 Base Model has normal trim level.
107 Base Model has a 1.5 liter engine.
108 Base Model engine sounds like putt, putt.
109 Base Model horn sounds like beep, beep.
110
111 Sport_Model
112 Sport Model has normal trim level.
113 Sport Model has a 2.0 liter engine.
114 Sport Model engine sounds like VROOM, VROOM.
115 Sport Model horn sounds like BEEP, BEEP.
116
117 Luxury_Sport_Model
118 Luxury Sport Model has luxury trim level.
119 Luxury Sport Model has a 2.0 liter engine.
120 Luxury Sport Model engine sounds like vroom, vroom.
121 Luxury Sport Model horn sounds like honk, honk.
122
123 Sport_Luxury_Model
124 Sport Luxury Model has luxury trim level.
125 Sport Luxury Model has a 2.0 liter engine.
126 Sport Luxury Model engine sounds like VROOM, VROOM.
127 Sport Luxury Model horn sounds like BEEP, BEEP.
128
129 Custom_Models
130 Sport Luxury Model has luxury trim level.
131 Sport Luxury Model now has custom trim level.
132 The class <class '__main__.Sport_Luxury_Model'> still has luxury trim level.
133 Sport Luxury Model has racing brakes.
134 <class '__main__.Sport_Luxury_Model'> has standard brakes.
135 ...
```