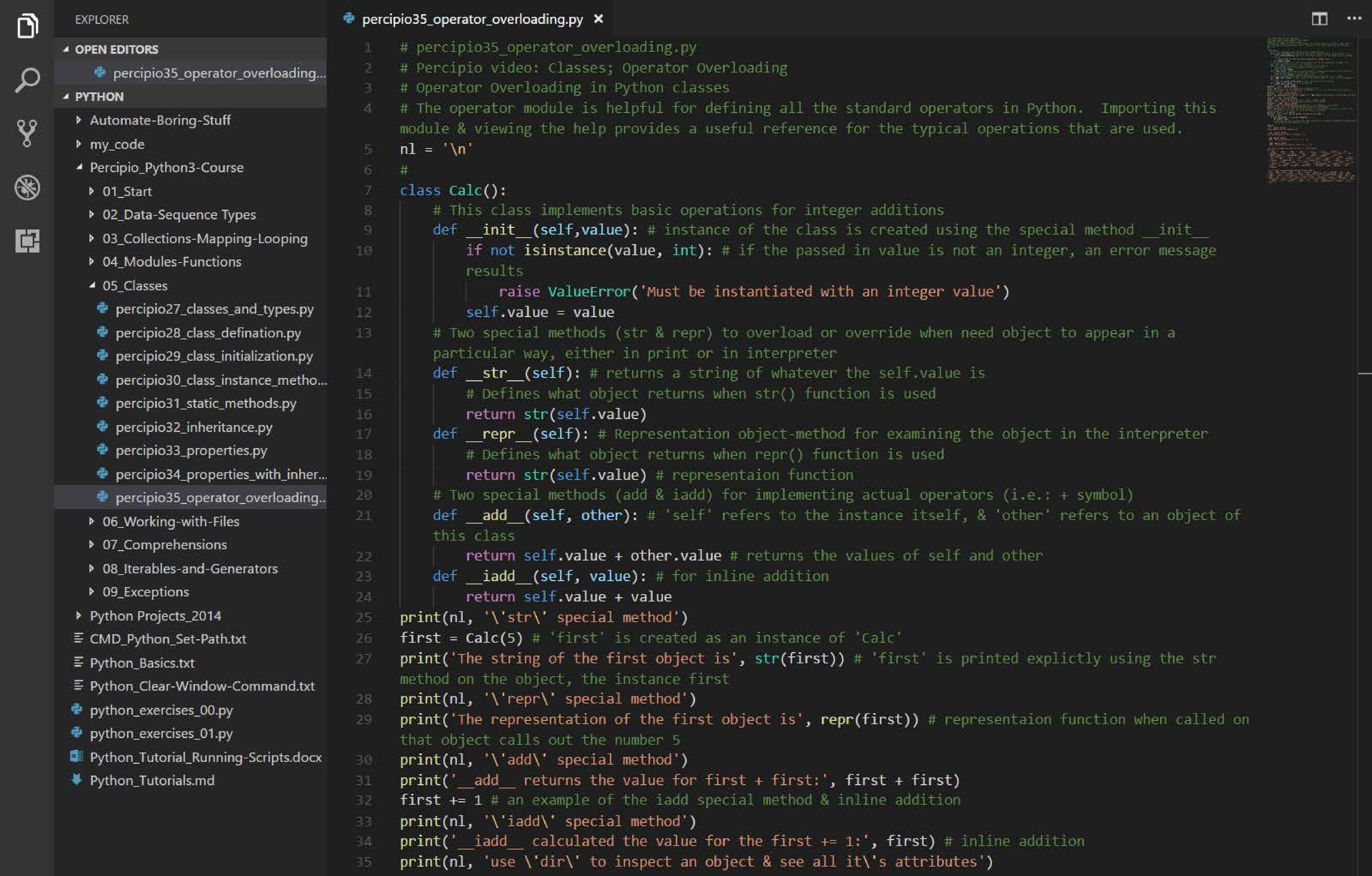
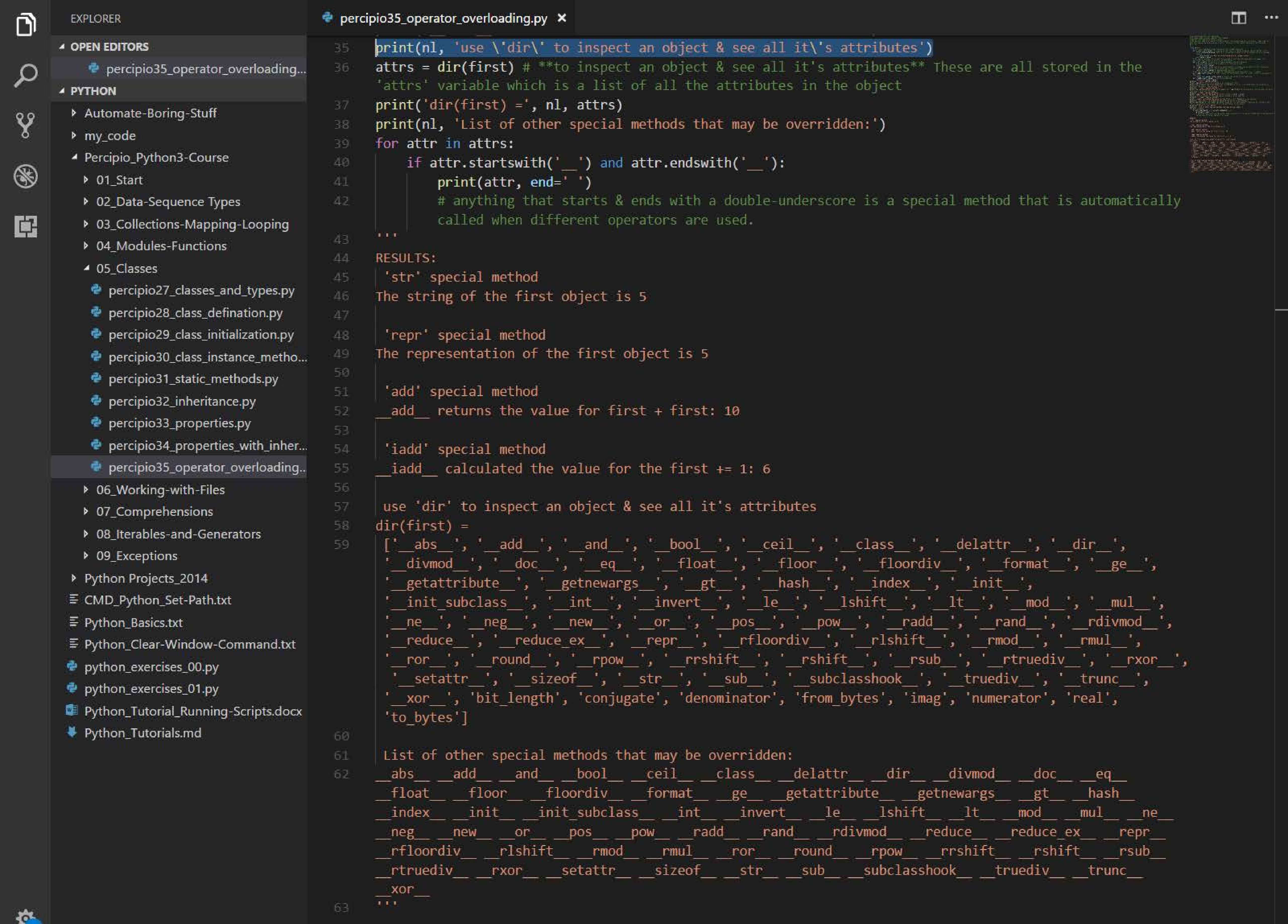

EXPLORER
OPEN EDITORS
percipio35_operator_overloading...
PYTHON
Automate-Boring-Stuff
my_code
Percipio_Python3-Course
01_Start
02_Data-Sequence Types
03_Collections-Mapping-Looping
04_Modules-Functions
05_Classes
percipio27_classes_and_types.py
percipio28_class_defination.py
percipio29_class_initialization.py
percipio30_class_instance_metho...
percipio31_static_methods.py
percipio32_inheritance.py
percipio33_properties.py
percipio34_properties_with_inher...
percipio35_operator_overloading...
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
percipio35_operator_overloading.py ×

```python
# percipio35_operator_overloading.py
# Percipio video: Classes; Operator Overloading
# Operator Overloading in Python classes
# The operator module is helpful for defining all the standard operators in Python.  Importing this module & viewing the help provides a useful reference for the typical operations that are used.
nl = '\n'
#
class Calc():
    # This class implements basic operations for integer additions
    def __init__(self,value): # instance of the class is created using the special method __init__
        if not isinstance(value, int): # if the passed in value is not an integer, an error message results
            raise ValueError('Must be instantiated with an integer value')
        self.value = value
    # Two special methods (str & repr) to overload or override when need object to appear in a particular way, either in print or in interpreter
    def __str__(self): # returns a string of whatever the self.value is
        # Defines what object returns when str() function is used
        return str(self.value)
    def __repr__(self): # Representation object-method for examining the object in the interpreter
        # Defines what object returns when repr() function is used
        return str(self.value) # representaion function
    # Two special methods (add & iadd) for implementing actual operators (i.e.: + symbol)
    def __add__(self, other): # 'self' refers to the instance itself, & 'other' refers to an object of this class
        return self.value + other.value # returns the values of self and other
    def __iadd__(self, value): # for inline addition
        return self.value + value
print(nl, '\'str\' special method')
first = Calc(5) # 'first' is created as an instance of 'Calc'
print('The string of the first object is', str(first)) # 'first' is printed explictly using the str method on the object, the instance first
print(nl, '\'repr\' special method')
print('The representation of the first object is', repr(first)) # representaion function when called on that object calls out the number 5
print(nl, '\'add\' special method')
print('__add__ returns the value for first + first:', first + first)
first += 1 # an example of the iadd special method & inline addition
print(nl, '\'iadd\' special method')
print('__iadd__ calculated the value for the first += 1:', first) # inline addition
print(nl, 'use \'dir\' to inspect an object & see all it\'s attributes')
```


EXPLORER
percipio35_operator_overloading.py
OPEN EDITORS
percipio35_operator_overloading...
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_metho...
percipio31_static_methods.py
percipio32_inheritance.py
percipio33_properties.py
percipio34_properties_with_inher...
percipio35_operator_overloading...
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

```
print(nl, 'use \'dir\' to inspect an object & see all it\'s attributes')
attrs = dir(first) # **to inspect an object & see all it's attributes** These are all stored in the 'attrs' variable which is a list of all the attributes in the object
print('dir(first) =', nl, attrs)
print(nl, 'List of other special methods that may be overridden:')
for attr in attrs:
    if attr.startswith('__') and attr.endswith('__'):
        print(attr, end=' ')
        # anything that starts & ends with a double-underscore is a special method that is automatically called when different operators are used.
'''
RESULTS:
 'str' special method
The string of the first object is 5

 'repr' special method
The representation of the first object is 5

 'add' special method
__add__ returns the value for first + first: 10

 'iadd' special method
__iadd__ calculated the value for the first += 1: 6

 use 'dir' to inspect an object & see all it's attributes
dir(first) =
 ['__abs__', '__add__', '__and__', '__bool__', '__ceil__', '__class__', '__delattr__', '__dir__', '__divmod__', '__doc__', '__eq__', '__float__', '__floor__', '__floordiv__', '__format__', '__ge__', '__getattribute__', '__getnewargs__', '__gt__', '__hash__', '__index__', '__init__', '__init_subclass__', '__int__', '__invert__', '__le__', '__lshift__', '__lt__', '__mod__', '__mul__', '__ne__', '__neg__', '__new__', '__or__', '__pos__', '__pow__', '__radd__', '__rand__', '__rdivmod__', '__reduce__', '__reduce_ex__', '__repr__', '__rfloordiv__', '__rlshift__', '__rmod__', '__rmul__', '__ror__', '__round__', '__rpow__', '__rrshift__', '__rshift__', '__rsub__', '__rtruediv__', '__rxor__', '__setattr__', '__sizeof__', '__str__', '__sub__', '__subclasshook__', '__truediv__', '__trunc__', '__xor__', 'bit_length', 'conjugate', 'denominator', 'from_bytes', 'imag', 'numerator', 'real', 'to_bytes']

 List of other special methods that may be overridden:
__abs__ __add__ __and__ __bool__ __ceil__ __class__ __delattr__ __dir__ __divmod__ __doc__ __eq__ __float__ __floor__ __floordiv__ __format__ __ge__ __getattribute__ __getnewargs__ __gt__ __hash__ __index__ __init__ __init_subclass__ __int__ __invert__ __le__ __lshift__ __lt__ __mod__ __mul__ __ne__ __neg__ __new__ __or__ __pos__ __pow__ __radd__ __rand__ __rdivmod__ __reduce__ __reduce_ex__ __repr__ __rfloordiv__ __rlshift__ __rmod__ __rmul__ __ror__ __round__ __rpow__ __rrshift__ __rshift__ __rsub__ __rtruediv__ __rxor__ __setattr__ __sizeof__ __str__ __sub__ __subclasshook__ __truediv__ __trunc__ __xor__
'''
```