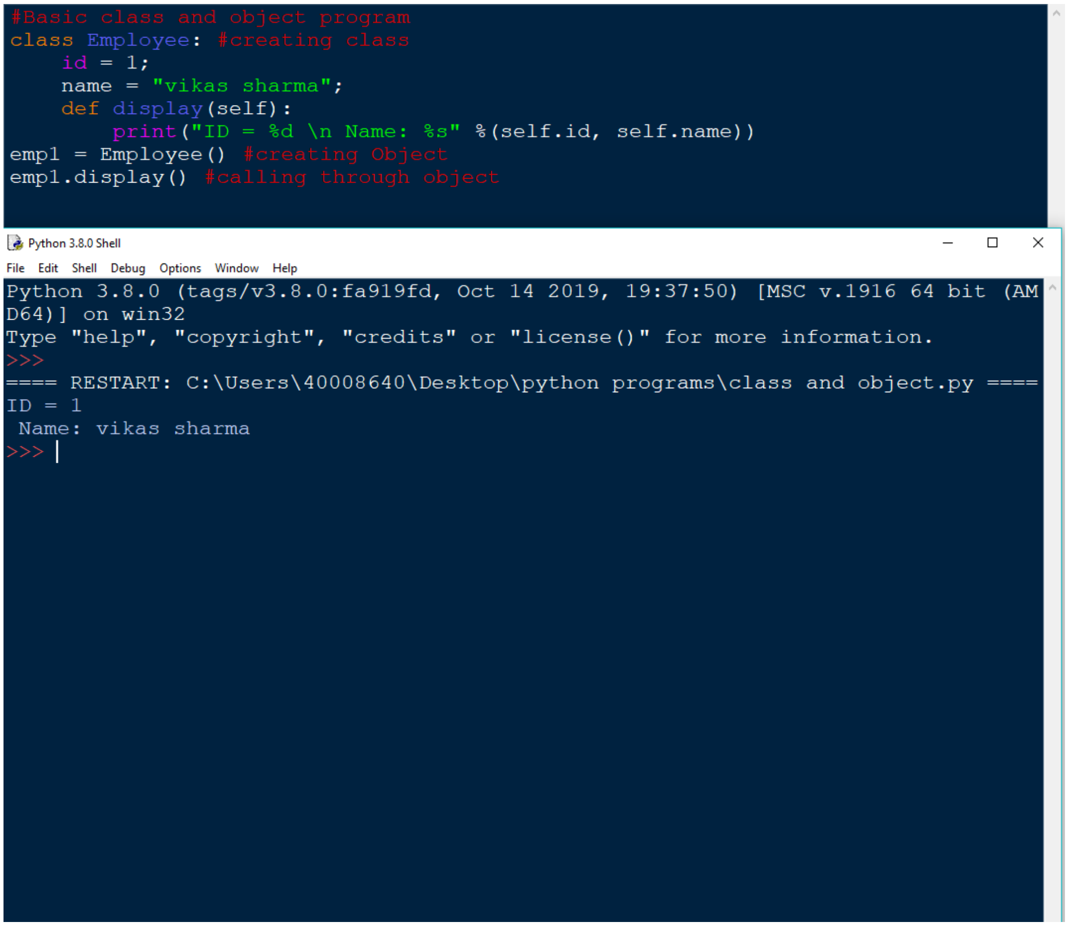
Python OOPS Concepts

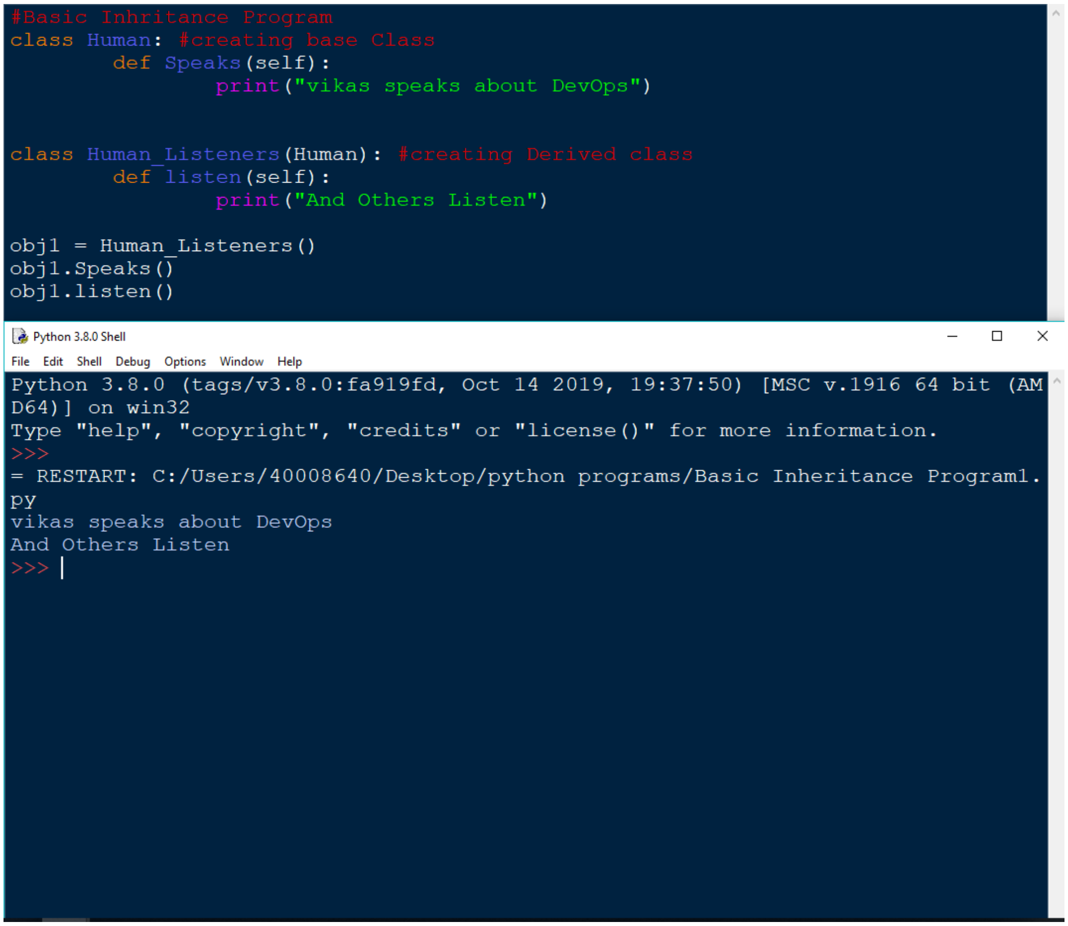
Basic class and object program



```
#Basic class and object program
class Employee: #creating class
    id = 1;
    name = "vikas sharma";
    def display(self):
        print("ID = %d \n Name: %s" %(self.id, self.name))
emp1 = Employee() #creating Object
emp1.display() #calling through object
```

```
Python 3.8.0 (tags/v3.8.0:fa919fd, Oct 14 2019, 19:37:50) [MSC v.1916 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: C:\Users\40008640\Desktop\python programs\class and object.py ====
ID = 1
 Name: vikas sharma
>>> |
```

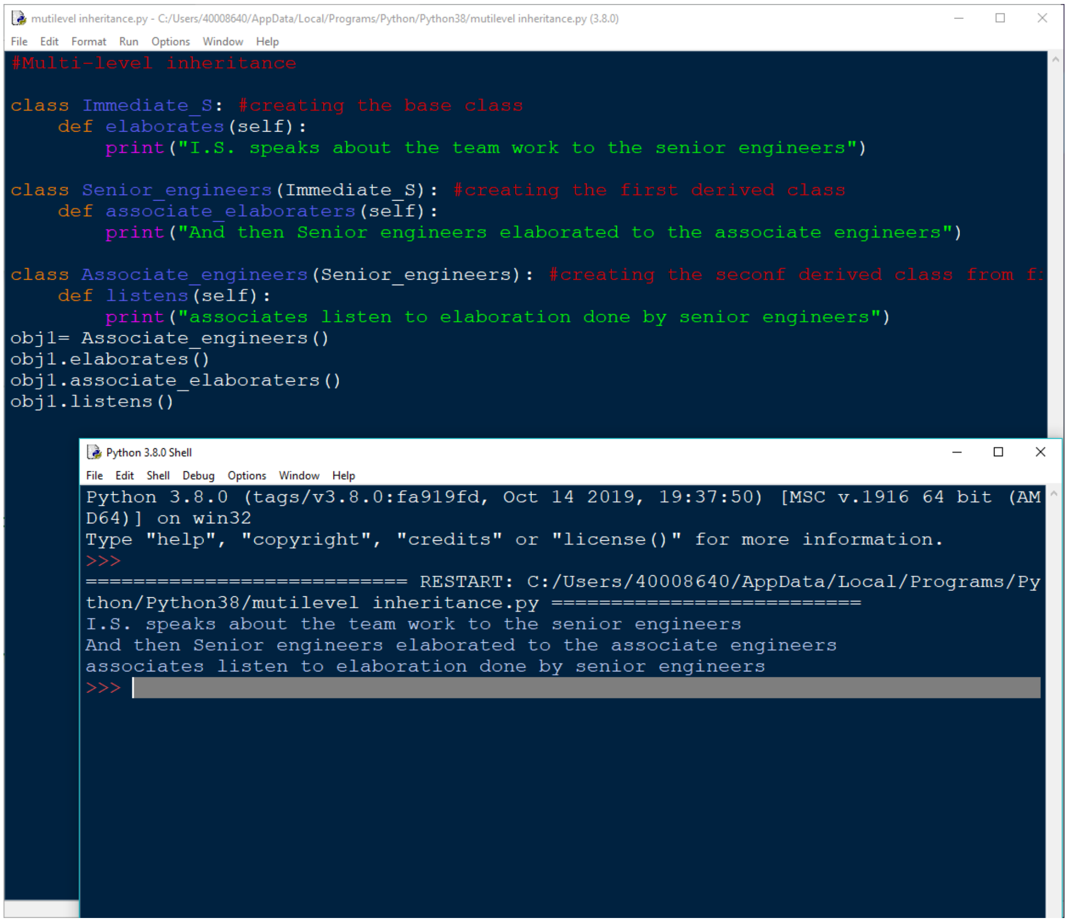
Basic Inheritance



```
#Basic Inhritance Program
class Human: #creating base Class
        def Speaks(self):
                print("vikas speaks about DevOps")


class Human_Listeners(Human): #creating Derived class
        def listen(self):
                print("And Others Listen")

obj1 = Human_Listeners()
obj1.Speaks()
obj1.listen()
```

```
Python 3.8.0 (tags/v3.8.0:fa919fd, Oct 14 2019, 19:37:50) [MSC v.1916 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/40008640/Desktop/python programs/Basic Inheritance Program1.
py
vikas speaks about DevOps
And Others Listen
>>> 
```

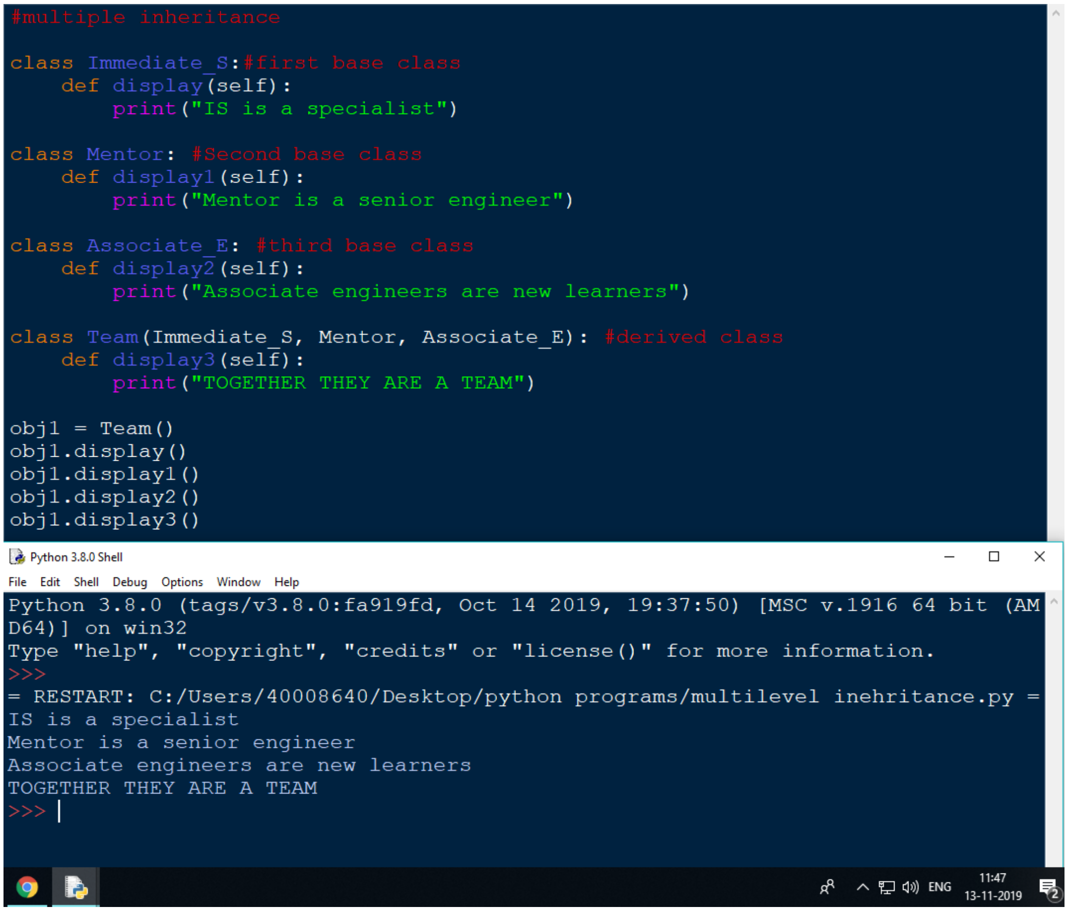
Multi-level Inheritance


mutilevel inheritance.py - C:/Users/40008640/AppData/Local/Programs/Python/Python38/mutilevel inheritance.py (3.8.0)
File Edit Format Run Options Window Help
Python 3.8.0 Shell
File Edit Shell Debug Options Window Help

```python
#Multi-level inheritance

class Immediate_S: #creating the base class
    def elaborates(self):
        print("I.S. speaks about the team work to the senior engineers")

class Senior_engineers(Immediate_S): #creating the first derived class
    def associate_elaboraters(self):
        print("And then Senior engineers elaborated to the associate engineers")

class Associate_engineers(Senior_engineers): #creating the seconf derived class from f
    def listens(self):
        print("associates listen to elaboration done by senior engineers")
obj1= Associate_engineers()
obj1.elaborates()
obj1.associate_elaboraters()
obj1.listens()
```

```
Python 3.8.0 (tags/v3.8.0:fa919fd, Oct 14 2019, 19:37:50) [MSC v.1916 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
=========================== RESTART: C:/Users/40008640/AppData/Local/Programs/Py
thon/Python38/mutilevel inheritance.py ==========================
I.S. speaks about the team work to the senior engineers
And then Senior engineers elaborated to the associate engineers
associates listen to elaboration done by senior engineers
>>>
```

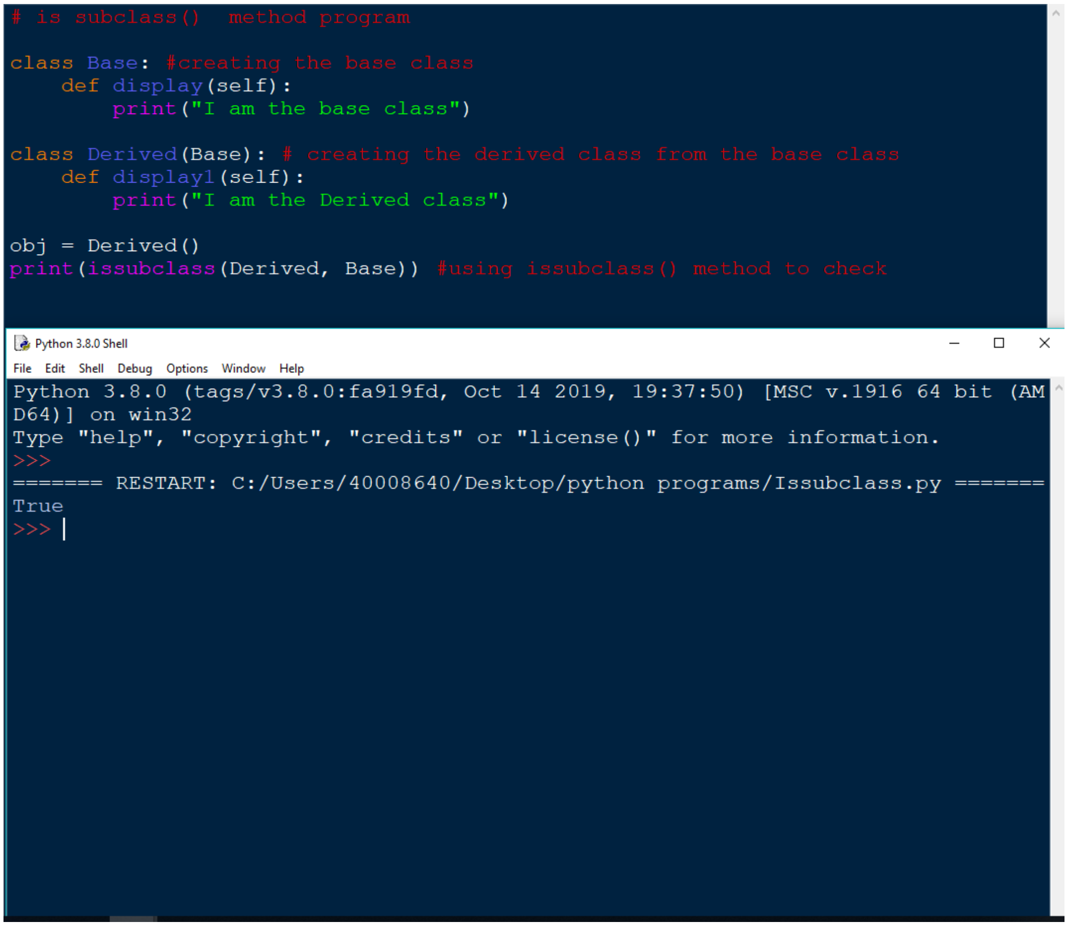
Multiple Inheritance



```
#multiple inheritance

class Immediate_S:#first base class
    def display(self):
        print("IS is a specialist")

class Mentor: #Second base class
    def display1(self):
        print("Mentor is a senior engineer")

class Associate_E: #third base class
    def display2(self):
        print("Associate engineers are new learners")

class Team(Immediate_S, Mentor, Associate_E): #derived class
    def display3(self):
        print("TOGETHER THEY ARE A TEAM")

obj1 = Team()
obj1.display()
obj1.display1()
obj1.display2()
obj1.display3()
```

```
Python 3.8.0 (tags/v3.8.0:fa919fd, Oct 14 2019, 19:37:50) [MSC v.1916 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/40008640/Desktop/python programs/multilevel inehritance.py =
IS is a specialist
Mentor is a senior engineer
Associate engineers are new learners
TOGETHER THEY ARE A TEAM
>>> |
```

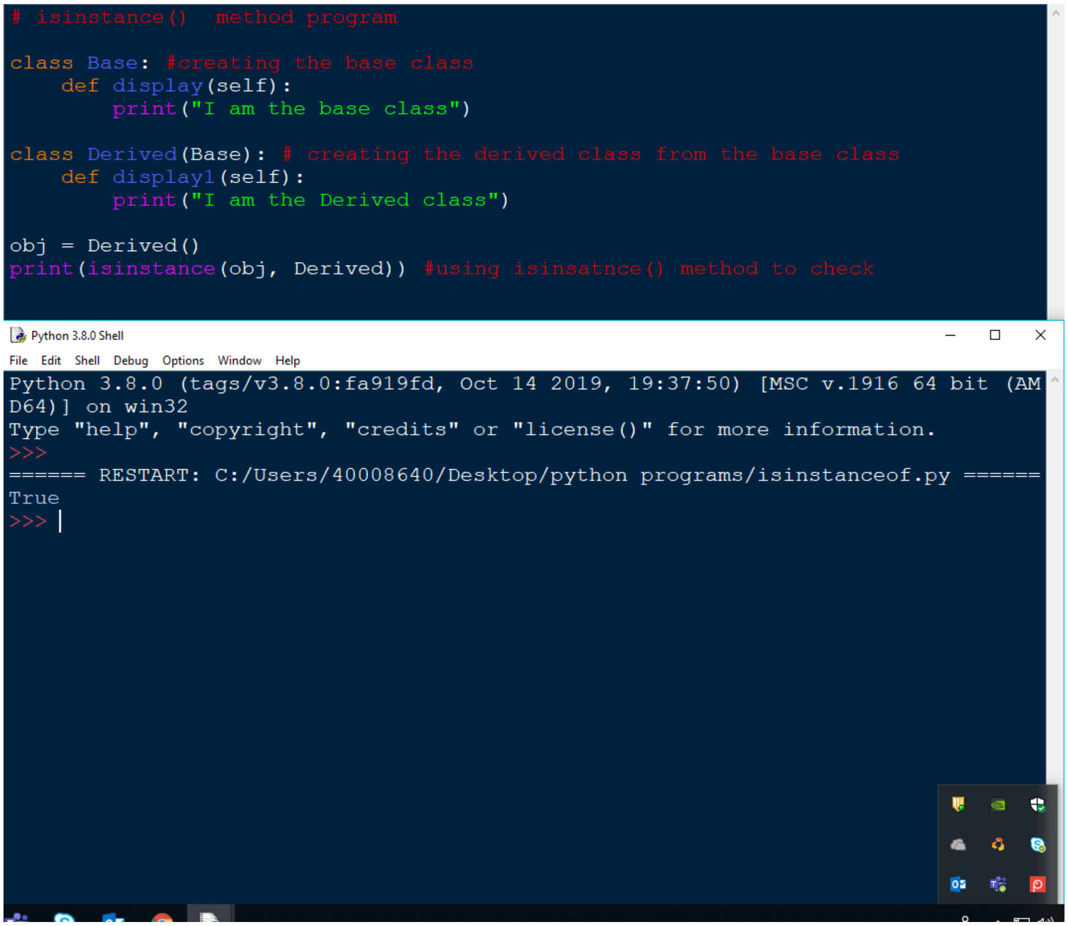
issubclass() method



```
# is subclass()  method program

class Base: #creating the base class
    def display(self):
        print("I am the base class")

class Derived(Base): # creating the derived class from the base class
    def display1(self):
        print("I am the Derived class")

obj = Derived()
print(issubclass(Derived, Base)) #using issubclass() method to check
```

```
Python 3.8.0 (tags/v3.8.0:fa919fd, Oct 14 2019, 19:37:50) [MSC v.1916 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
======= RESTART: C:/Users/40008640/Desktop/python programs/Issubclass.py =======
True
>>> 
```

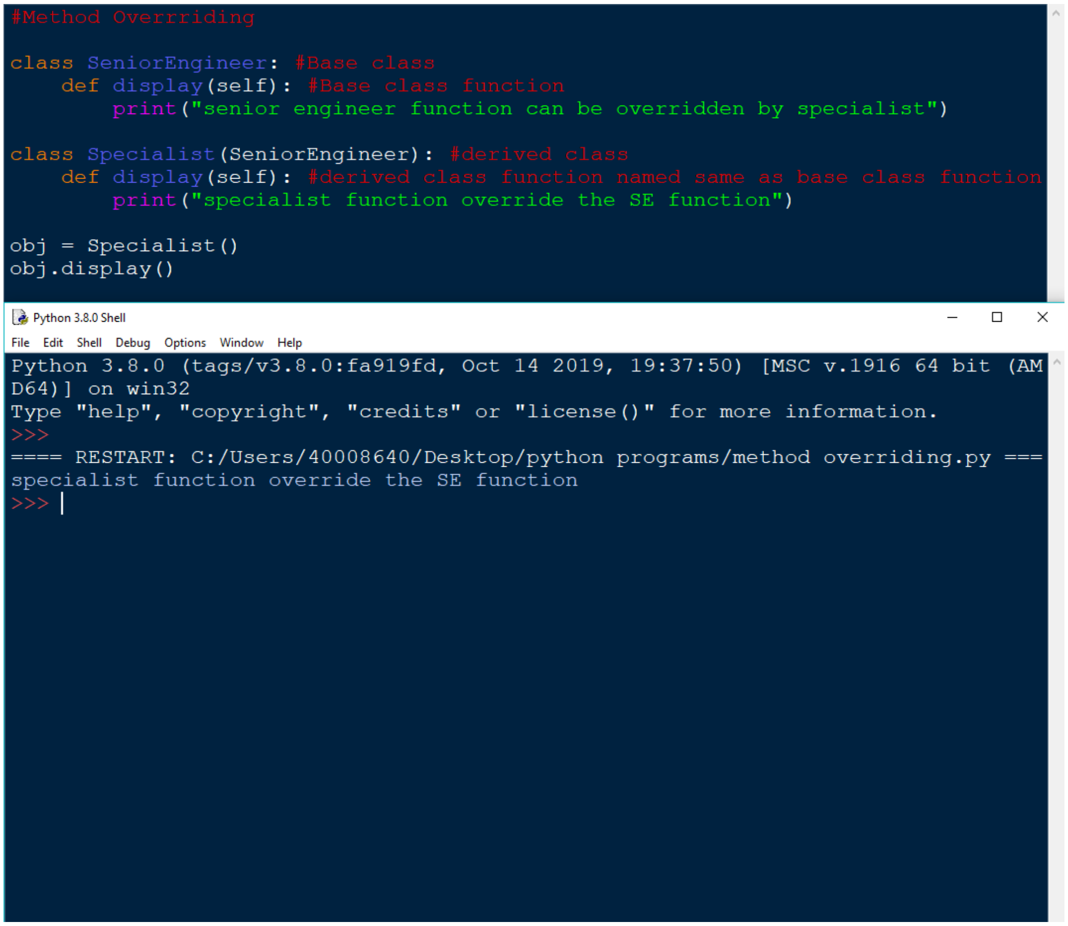
isinstance() method



```
# isinstance()  method program

class Base: #creating the base class
    def display(self):
        print("I am the base class")

class Derived(Base): # creating the derived class from the base class
    def display1(self):
        print("I am the Derived class")

obj = Derived()
print(isinstance(obj, Derived)) #using isinsatnce() method to check
```

```
Python 3.8.0 (tags/v3.8.0:fa919fd, Oct 14 2019, 19:37:50) [MSC v.1916 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
====== RESTART: C:/Users/40008640/Desktop/python programs/isinstanceof.py ======
True
>>> |
```

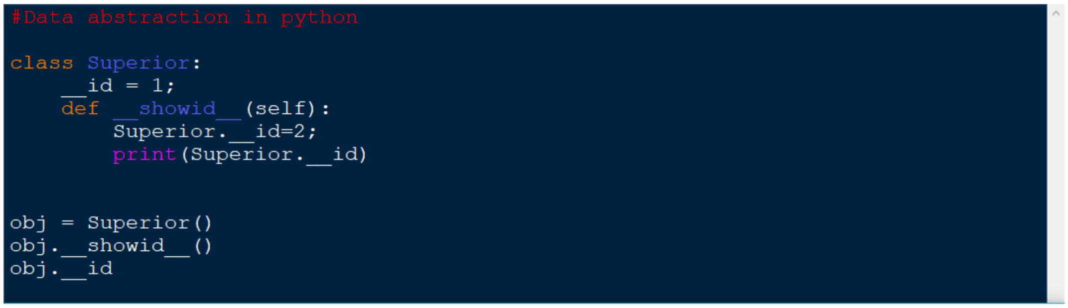
Method Overriding



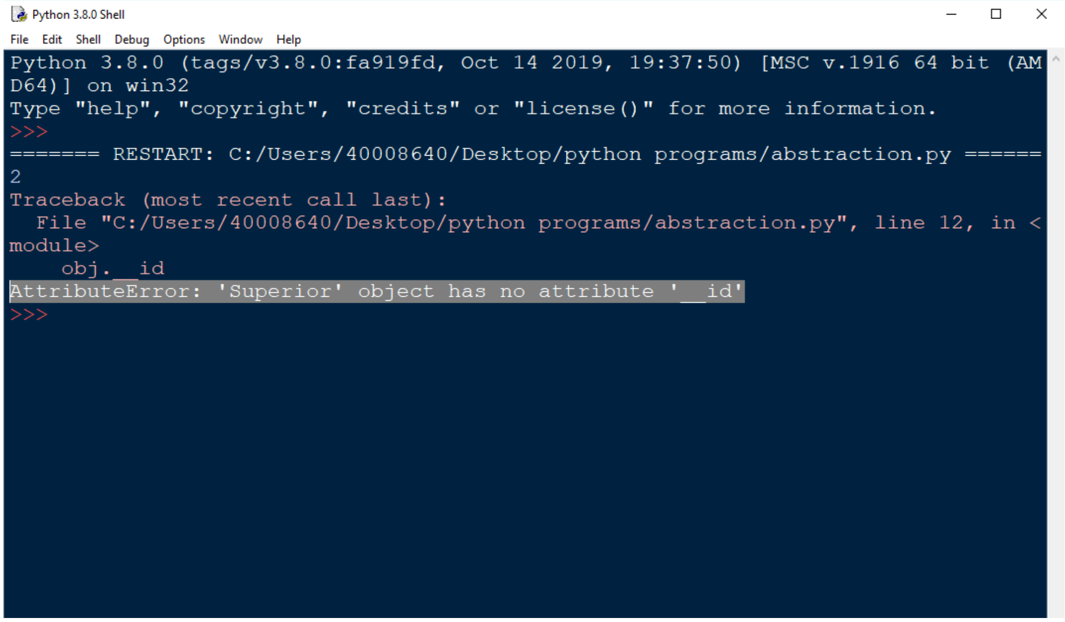
```
#Method Overrriding

class SeniorEngineer: #Base class
    def display(self): #Base class function
        print("senior engineer function can be overridden by specialist")

class Specialist(SeniorEngineer): #derived class
    def display(self): #derived class function named same as base class function
        print("specialist function override the SE function")

obj = Specialist()
obj.display()
```

```
Python 3.8.0 (tags/v3.8.0:fa919fd, Oct 14 2019, 19:37:50) [MSC v.1916 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: C:/Users/40008640/Desktop/python programs/method overriding.py ===
specialist function override the SE function
>>> 
```

Data abstraction in python



```
#Data abstraction in python

class Superior:
    __id = 1;
    def __showid__(self):
        Superior.__id=2;
        print(Superior.__id)


obj = Superior()
obj.__showid__()
obj.__id
```



```
Python 3.8.0 (tags/v3.8.0:fa919fd, Oct 14 2019, 19:37:50) [MSC v.1916 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
======= RESTART: C:/Users/40008640/Desktop/python programs/abstraction.py ======
2
Traceback (most recent call last):
  File "C:/Users/40008640/Desktop/python programs/abstraction.py", line 12, in <
module>
    obj.__id
AttributeError: 'Superior' object has no attribute '__id'
>>>
```