Object Oriented Programming using Python (I)

Lecture(8)

Class Inheritance
Method Resolution Order (MRO)

Prepared by: Ahmed Eskander Mezher
University of Information Technology and Communications
College of Business Informatics

Multiple Inheritance and the Lookup Tree

As its name indicates, multiple inheritance is Python is when a class inherits from multiple classes.
For example, a child inherits personality traits from both parents (Mother and Father).

Python Multiple Inheritance Syntax

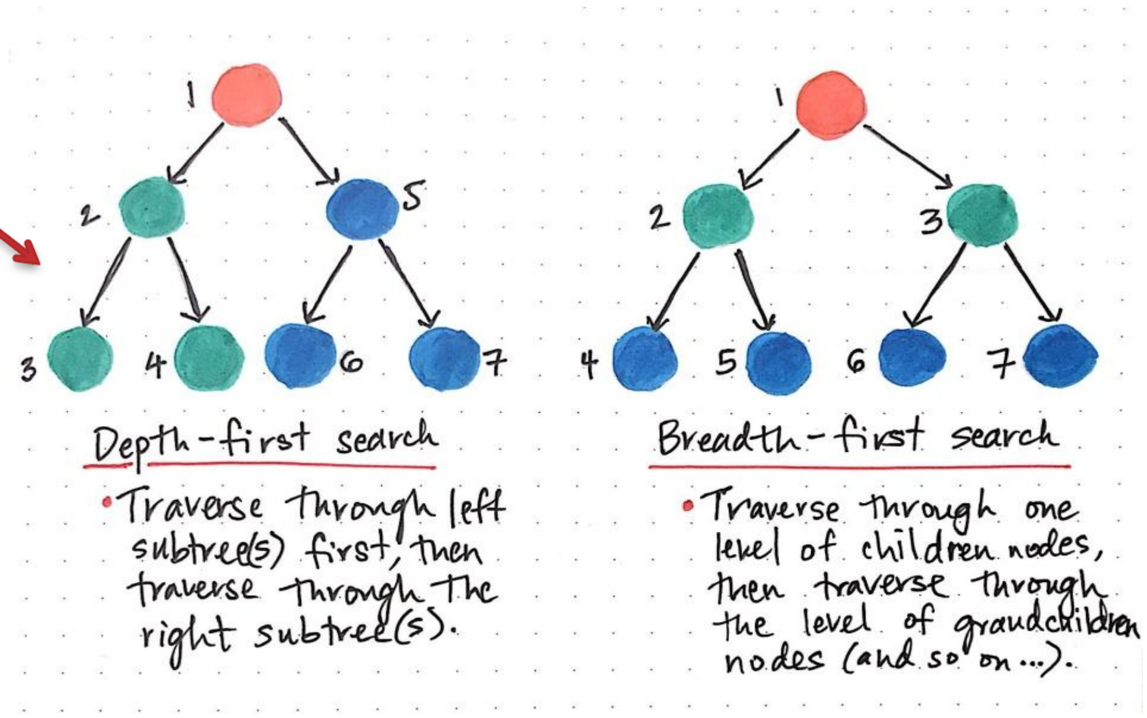
To make a class inherits from multiple parents classes, we write the the names of these classes inside the parentheses to the derived class while defining it. We separate these names with comma.
Below is an example of that:

```
>>> class Mother:
      pass

>>> class Father:
     pass

>>> class Child(Mother, Father):
     pass

>>> issubclass(Child, Mother) and issubclass(Child, Father)
True
```

Multiple inheritance refers to the ability of inheriting from two or more than two class. The complexity arises as child inherits from parent and parents inherits from the grandparent class. Python climbs an inheriting tree looking for attributes that is being requested to be read from an object. It will check the in the instance, within class then

parent class and lastly from the grandparent class. Now the question arises in what order the classes will be searched - breath-first or depth-first. By default, Python goes with the depth-first.

Python choose depth first


1
2
5
3
4
6
7
Depth-first search
• Traverse through left subtree(s) first, then traverse through the right subtree(s).
1
2
3
4
5
6
7
Breadth-first search
• Traverse through one level of children nodes, then traverse through the level of grandchildren nodes (and so on...).

Method Resolution Order (**MRO**)

is the order in which **Python** looks for a method in a hierarchy of classes. it plays vital role in the context of multiple inheritance as single method may be found in multiple super classes.

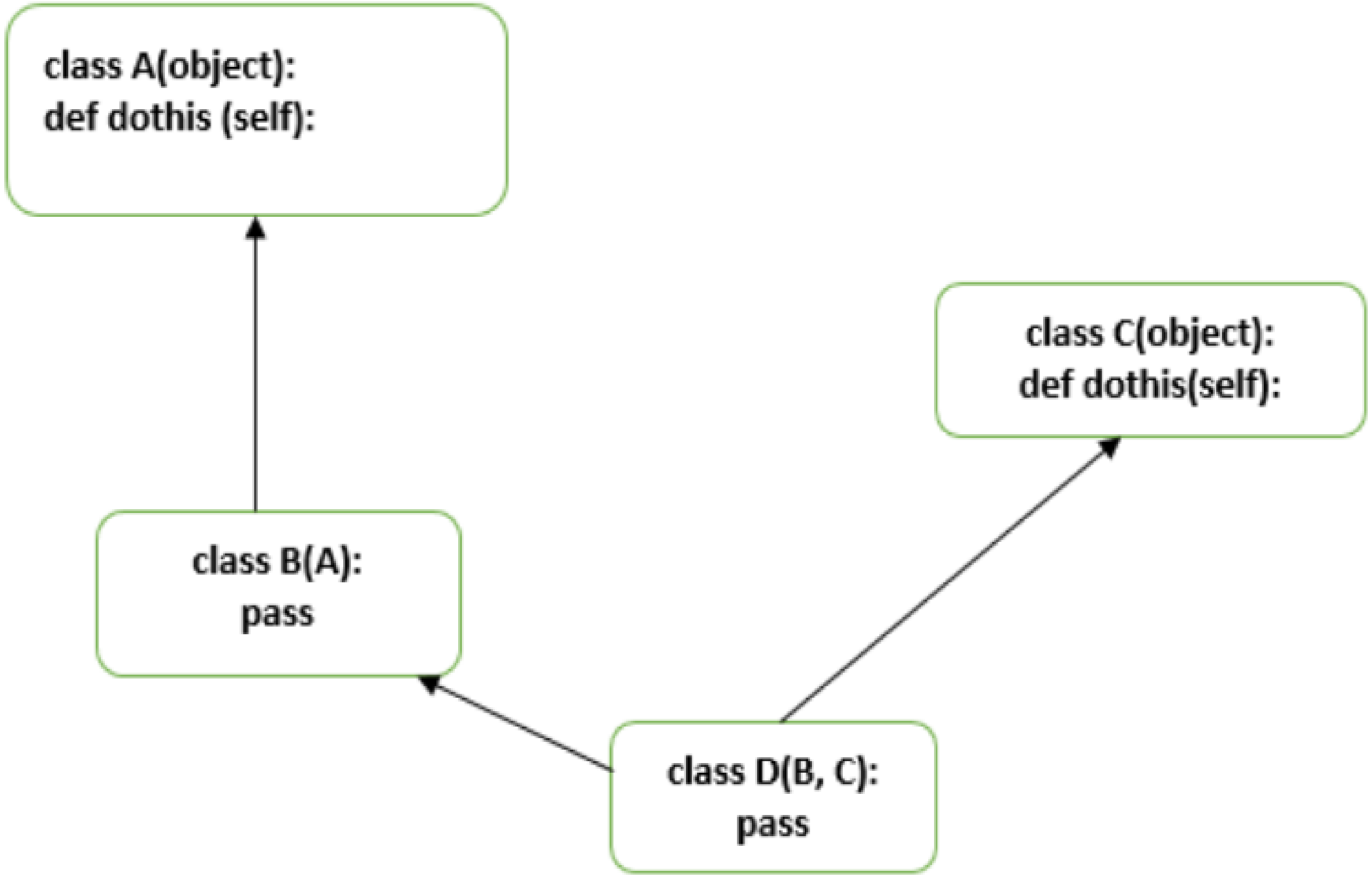
Syntax:

```
Class_name.mro()
```

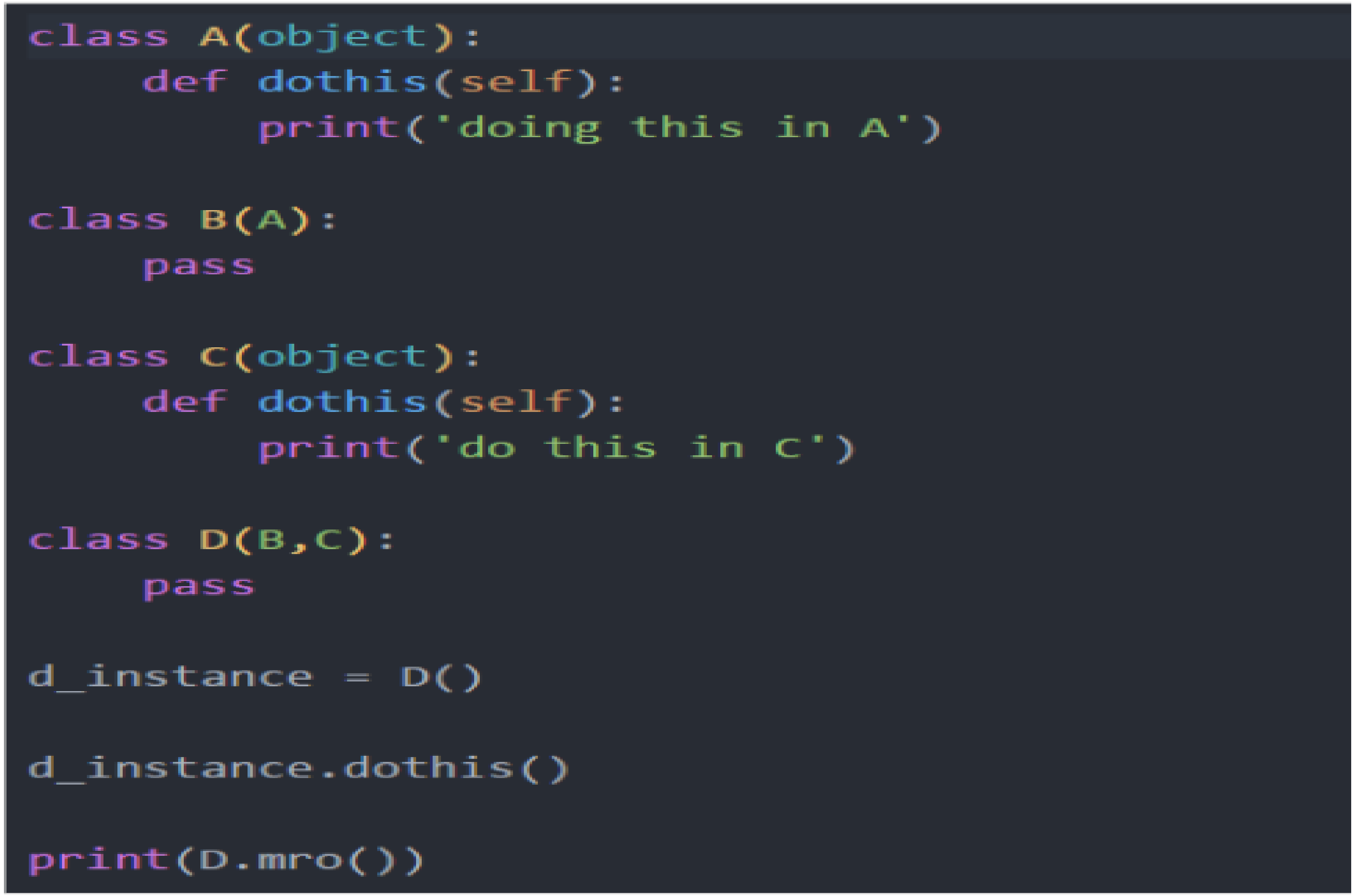
That's is why in the below diagram the Python searches the dothis() method first in class A. So the method resolution order in the below example will be

Mro- D->B->A->C

Look at the below multiple inheritance diagram:

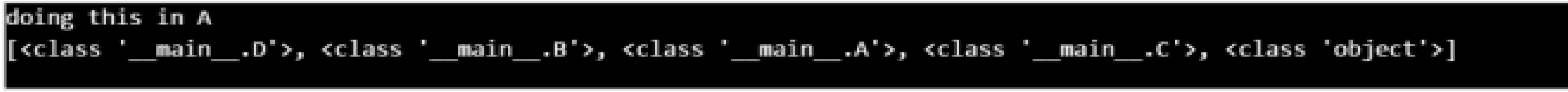


Let’s go through an example to understand the “mro” feature of an Python.



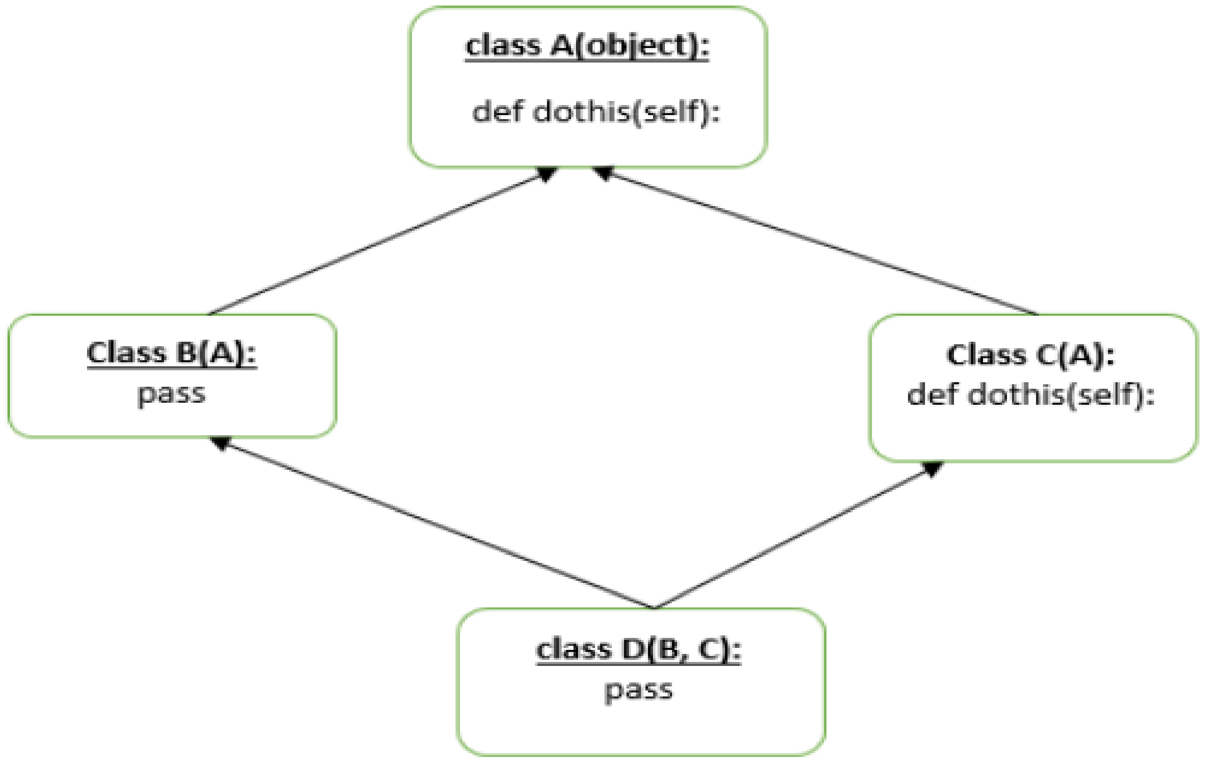
```python
class A(object):
    def dothis(self):
        print('doing this in A')

class B(A):
    pass

class C(object):
    def dothis(self):
        print('do this in C')

class D(B,C):
    pass

d_instance = D()

d_instance.dothis()

print(D.mro())
```

Output



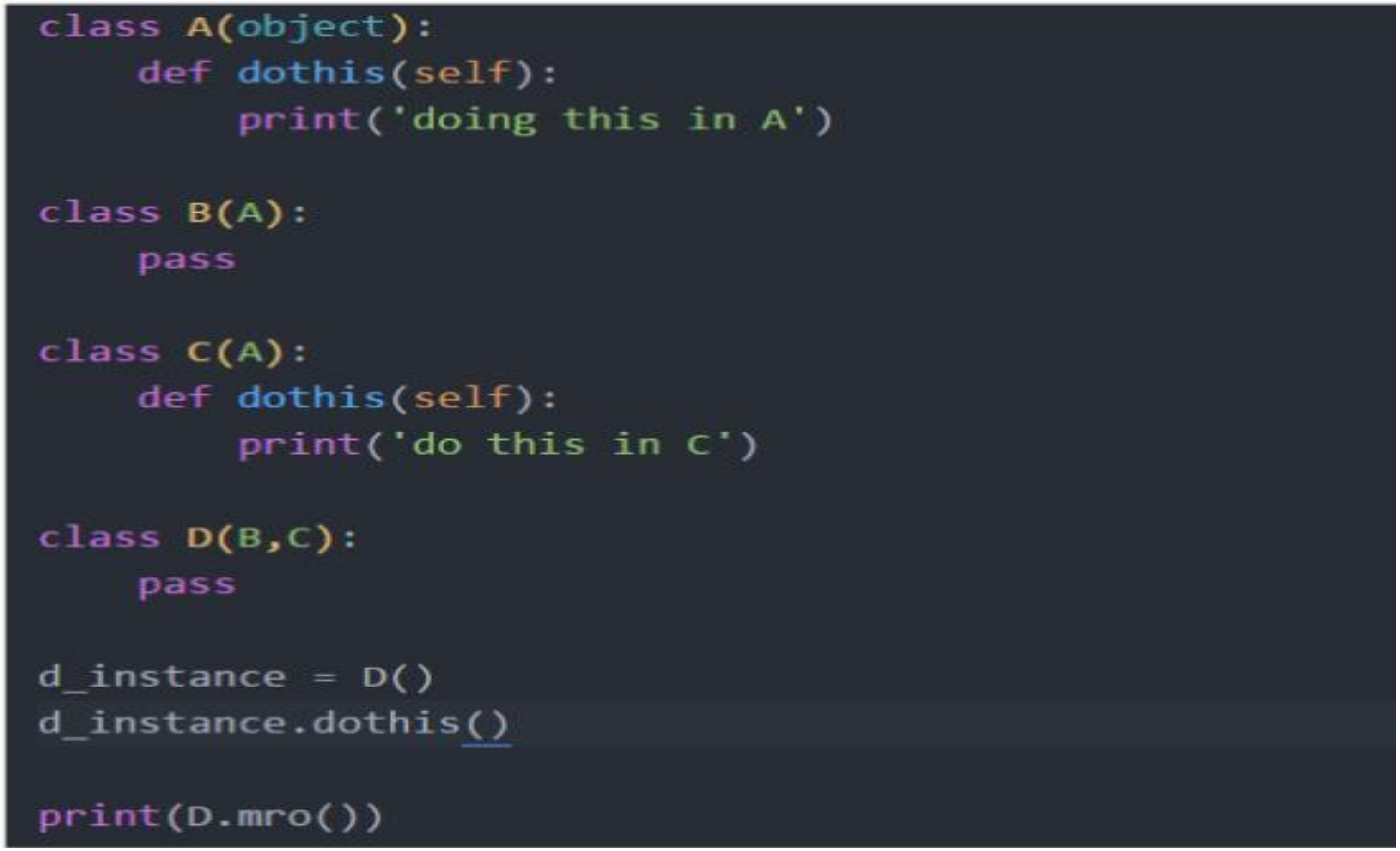
```
doing this in A
[<class '__main__.D'>, <class '__main__.B'>, <class '__main__.A'>, <class '__main__.C'>, <class 'object'>]
```

Let's take another example of "diamond shape" multiple inheritance.



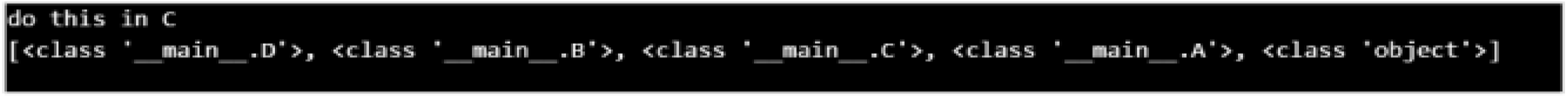
Above diagram will be considered ambiguous. From our previous example understanding "method resolution order" .i.e. mro will be D->B->A->C->A but it's not. On getting the second A from the C, Python will ignore the previous A. so the mro will be in this case will be D->B->C->A.

Let’s create an example based on above diagram:



```python
class A(object):
    def dothis(self):
        print('doing this in A')

class B(A):
    pass

class C(A):
    def dothis(self):
        print('do this in C')

class D(B,C):
    pass

d_instance = D()
d_instance.dothis()

print(D.mro())
```

Output



```
do this in C
[<class '__main__.D'>, <class '__main__.B'>, <class '__main__.C'>, <class '__main__.A'>, <class 'object'>]
```

Simple rule to understand the above output is- if the same class appear in the method resolution order, the earlier appearances of this class will be remove from the method resolution order.

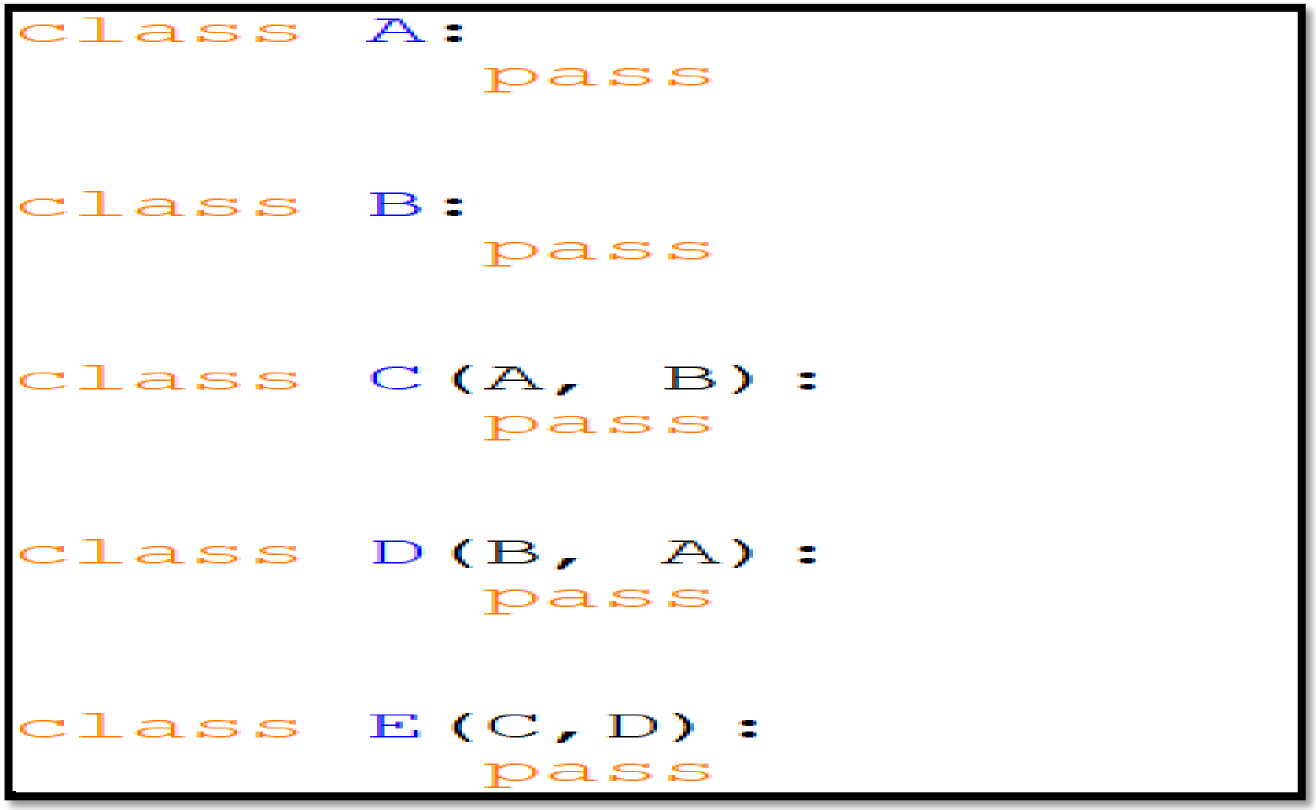
In conclusion:

- Any class can inherit from multiple classes
- Python normally uses a “depth-first” order when searching inheriting classes.
- But when two classes inherit from the same class, Python eliminates the first appearances of that class from the mro.

Practical Question

1-draw UML diagram for the following python program

2- what is the output of **MRO function for class (E)**



```
class  A:
        pass


class  B:
        pass


class  C(A, B):
        pass


class  D(B, A):
        pass


class  E(C,D):
        pass
```