INHERITANCE

1. What is inheritance ?

Inheriting the attributes and methods of a base class into a derived class is called Inheritance.

Syntax:

```
class BaseClass:
  # Body of BaseClass

class DerivedClass(BaseClass):
  # Body of DerivedClass
```

Example:

```
class Shape:
  unitOfMeasurement = 'centimetre'

class Square(Shape):
  def __init__(self):
    # The attribute unitOfMeasurement has been inherited from the class Shape to this class Square
    print("Unit of measurement for this square: ", self.unitOfMeasurement)

square = Square()
```

2. What is multiple inheritance ?

Mechanism in which a derived class inherits from two or more base classes is called a multiple inheritance

Syntax:

```
class baseClassOne:
  # Body of baseClassOne

class baseClassTwo:
  # Body of baseClassTwo
```

```
class derivedClass(baseClassOne, baseClassTwo):
  # Body of derived class
```

Example:

```
class OperatingSystem:
  multiTasking = True

class Apple:
  website = 'www.apple.com'

class MacOS(OperatingSystem, Apple):
    def __init__(self):
      if self.multiTasking is True:
        # The class MacOS has inherited 'multitasking' attribute from the class OperatingSystem and 'website' attribute from the class 'Apple'
        print("MacOS is a multitasking operating system. Visit {} for more details".format(self.website))

mac = MacOS()
```

3. What is multilevel inheritance ?

Mechanism in which a derived class inherits from a base class which has been derived from another base class is called a multilevel inheritance

Syntax:

```
class BaseClass:
  # Body of baseClass

class DerivedClassOne(BaseClass):
  # Body of DerivedClassOne

class DerivedClassTwo(DerivedClassOne):
```

```
        # Body of DerivedClassTwo
```

Example:

```
class Apple:
  website = 'www.apple.com'

class MacBook(Apple):
  deviceType = 'notebook computer'

class MacBookPro(MacBook):
  def __init__(self):
    # This class inherits deviceType from the base class MacBook. It also inherits website from base class of MacBook, which is Apple.
    print("This is a {}. Visit {} for more details".format(self.deviceType, self.website))

macBook = MacBookPro()
```

5. What is an abstract base class ?

A base class which contains abstract methods that are to be overridden in its derived class is called an abstract base class. They belong to the abc module.

Example:

```
from abc import ABCMeta, abstractmethod
class Shape(metaclass = ABCMeta):
    @abstractmethod
    def area(self):
        return 0

class Square(Shape):
def area(self, side)
    return side * side
```

6. What are the naming conventions used for private, protected and public members ?

Private -> __memberName
Protected -> _memberName
Public -> memberName

7. How is name mangling done for private members by Python ?

Name mangling is done by prepending the member name with an underscore and class name.

_className__memberName