Q1. What is the difference between \_\_getattr\_\_ and \_\_getattribute\_\_?

**Ans:**

getattribute: Is used to retrieve an attribute from an instance. It captures every attempt to access an instance attribute by using dot notation or getattr() built-in function.

getattr: Is executed as the last resource when attribute is not found in an object. You can choose to return a default value or to raise AttributeError.

Q2. What is the difference between properties and descriptors?

**Ans:**

Properties - We can bind getter, setter (and deleter) functions with an attribute name, using the built-in property function. When we do this, each reference to an attribute looks like simple, direct access, but invokes the appropriate function of the object.

Descriptors - We can bind getter, setter (and deleter) functions into a separate class. We then assign an object of this class to the attribute name. When we do this, each reference to an attribute looks like simple, direct access, but invokes an appropriate function of the Descriptor object.

Q3. What are the key differences in functionality between \_\_getattr\_\_ and \_\_getattribute\_\_, as well as properties and descriptors?

**Ans:** \_\_getattribute\_\_ is used to get attribute from a class.

\_\_getattr\_\_, is the last resource to use when an attribute not found in class, and you can return any specific thing or raise attribute error in this case.

Descriptor – We can bind getter, setter (and deleter) functions into a separate class. We then assign an object of this class to the attribute name. below is an example:

class Celsius( object ):

def \_\_init\_\_( self, value=0.0 ):

self.value= float(value)

def \_\_get\_\_( self, instance, owner ):

return self.value

def \_\_set\_\_( self, instance, value ):

self.value= float(value)

class Farenheit( object ):

def \_\_get\_\_( self, instance, owner ):

return instance.celsius \* 9 / 5 + 32

def \_\_set\_\_( self, instance, value ):

instance.celsius= (float(value)-32) \* 5 / 9

class Temperature( object ):

celsius= Celsius()

farenheit= Farenheit()

>>>

oven= Temperature()

>>>

oven.farenheit= 450

>>>

oven.celsius

232.22222222222223

Property - We can bind getter, setter (and deleter) functions with an attribute name, using the built-in property function. When we do this, each reference to an attribute looks like simple, direct access, but invokes the appropriate function of the object. Below is example:

class Temperature( object ):

def fget( self ):

return self.celsius \* 9 / 5 + 32

def fset( self, value ):

self.celsius= (float(value)-32) \* 5 / 9

farenheit= property( fget, fset )

def cset( self, value ):

self.cTemp= float(value)

def cget( self ):

return self.cTemp

celsius= property( cget, cset, doc="Celsius temperature" )

>>>

oven= Temperature()

>>>

oven.farenheit= 450

>>>

oven.celsius

232.22222222222223