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

Ans: \_\_getattr\_\_ will intercept the request over unknown attribute and will be assigning zero to the unknown attribute and returning it to show without raising any exception.

If you have \_\_getattribute\_\_ method in your class, Python invokes this method for every attribute regardless whether it exists or not. ‘if’ condition looks for the mentioned substring at the beginning of every attribute’s name and raise an exception AttributeError when it gets the condition.

Q2. What is the difference between properties and descriptors?

Ans: Python descriptors are created to manage the attributes of different classes which use the object as reference. In descriptors we used three different methods that are \_\_getters\_\_(), \_\_setters\_\_(), and \_\_delete\_\_(). If any of those methods are defined for an object, it can be termed as a descriptor. Normally, Python uses methods like getters and setters to adjust the values on attributes without any special processing. It’s just a basic storage system. Sometimes, You might need to validate the values that are being assigned to a value. A descriptor is a mechanism behind properties, methods, static methods, class methods, and super().

Property()is a built-in function that creates and returns a property object. A property object has three methods, getter(), setter(), and delete(). property() function in Python has four arguments property(fget, fset, fdel, doc), fget is a function for retrieving an attribute value. fset is a function for setting an attribute value. fdel is a function for deleting an attribute value. doc creates a docstring for attribute. A property object has three methods, getter(), setter(), and delete() to specify fget, fset and fdel individually.

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

Ans: \_\_getattr\_\_ :

1. \_\_getattr\_\_ gets called if there is no attribute in the instance.
2. It’s invoked “last”, if Python can’t find that attribute.(lowest priority)

\_\_getattribute\_\_ :

1. \_\_getattribute\_\_ gets called all the times, whether there is the attribute or not.
2. It’s invoked “first”(highest priority) — it actually “intercepts” every lookup.

Descriptors are a low-level mechanism that lets you hook into an object's attributes being accessed. Properties are a high-level application of this; that is, properties are implemented using descriptors.