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

A:

The main difference between \_\_getattr\_\_ and \_\_getattribute\_\_ is that if the attribute was not found by the usual way then \_\_getattr\_\_ is used. Whereas the \_\_getattribute\_\_ is used before looking at the actual attributes on the object.

Q2. What is the difference between properties and descriptors?

A:

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\_\_().

Python's property() is the Pythonic way to avoid formal getter and setter methods in your code. This function allows you to turn class attributes into properties or managed attributes. Since property() is a built-in function, you can use it without importing anything.

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

properties and descriptors?

A:

The main difference between \_\_getattr\_\_ and \_\_getattribute\_\_ is that if the attribute was not found by the usual way then \_\_getattr\_\_ is used. Whereas the \_\_getattribute\_\_ is used before looking at the actual attributes on the object.

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\_\_().

Python's property() is the Pythonic way to avoid formal getter and setter methods in your code. This function allows you to turn class attributes into properties or managed attributes. Since property() is a built-in function, you can use it without importing anything.