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

\_\_getattr\_\_ and \_\_getattribute\_\_ are both methods in Python that deal with attribute access, but they serve different purposes:

* \_\_getattr\_\_: This method is called when an attribute that doesn't exist is accessed. It allows you to dynamically create or retrieve attributes that are not directly defined in the object.
* \_\_getattribute\_\_: This method is called for every attribute access, whether the attribute exists or not. It is a more general and powerful method, used for custom attribute access and can be overridden to customize attribute behavior.

Q2. What is the difference between properties and descriptors?

Properties and descriptors are both techniques for controlling attribute access in Python, but they have different levels of granularity and use cases:

* Properties: Properties are a high-level way to define computed attributes. They allow you to define getter, setter, and deleter methods using the @property, @<attribute>.setter, and @<attribute>.deleter decorators. Properties are defined on a per-attribute basis and provide a clean interface for attribute access.
* Descriptors: Descriptors are a lower-level mechanism for controlling attribute access. They involve defining classes with \_\_get\_\_, \_\_set\_\_, and \_\_delete\_\_ methods, which can be used to customize attribute behavior on a per-class basis. Descriptors offer more control and flexibility but require more involved coding.

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

* \_\_getattr\_\_ vs. \_\_getattribute\_\_:
* \_\_getattr\_\_: Called only when an attribute is not found through normal lookup.
* \_\_getattribute\_\_: Called for every attribute access, whether the attribute exists or not.

Properties vs. Descriptors:

**Properties:**

Defined using @property, @<attribute>.setter, and @<attribute>.deleter decorators.

Used to define computed attributes that have custom getter, setter, and deleter methods.

Provides a clean and intuitive syntax.

**Descriptors:**

Involves defining classes with \_\_get\_\_, \_\_set\_\_, and \_\_delete\_\_ methods.

Used to customize attribute access behavior at a lower level.

Offers more flexibility and control but requires more involved coding.

Both \_\_getattr\_\_ and properties are used for attribute access customization at a higher level, whereas \_\_getattribute\_\_ and descriptors offer more advanced and fine-grained control over attribute access behavior.