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

\_\_getattribute\_\_: This method is called for all attribute access, whether the attribute exists or not. It is called before \_\_getattr\_\_.

\_\_getattr\_\_: This method is called only when the requested attribute is not found through the usual ways (i.e., not present in the instance's dictionary, and not found through inheritance).

Q2. What is the difference between properties and descriptors?

Properties: Properties are a high-level way of defining read-only or read-write attributes with getter and setter methods. They are created using the property decorator.

Descriptors: Descriptors provide a lower-level mechanism for defining attributes with custom behavior. They involve defining a separate class with \_\_get\_\_, \_\_set\_\_, and/or \_\_delete\_\_ methods.

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

\_\_getattribute\_\_ is a more general method that is called for every attribute access, while \_\_getattr\_\_ is specifically for handling attribute access when the attribute is not found by normal means.

Properties are a high-level way of defining getter and setter methods for attributes, while descriptors provide a more flexible and customizable mechanism for attribute access.

\_\_getattribute\_\_ and descriptors allow more fine-grained control over attribute access and modification compared to \_\_getattr\_\_ and properties.