**Assignment - 10**

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

Ans: The main difference between \_\_getattr\_\_ and \_\_getattribute\_\_ lies in their purpose and behavior:

* \_\_getattr\_\_(self, name): This method is called when an attribute lookup fails. It allows you to define a fallback behavior for accessing attributes that are not directly defined in the class. It is only invoked when the requested attribute is not found via the normal lookup process.
* \_\_getattribute\_\_(self, name): This method is called for every attribute access, regardless of whether the attribute exists or not. It is invoked before looking up the attribute in the instance's dictionary or class hierarchy. Be cautious when implementing \_\_getattribute\_\_ as it can lead to infinite recursion if not used carefully.

Q2. What is the difference between properties and descriptors?

Ans: Properties and descriptors are both ways to define custom behavior for attribute access, but they differ in their implementation and usage:

* Properties: Properties are a built-in feature of Python that allow you to define methods to get, set, and delete attributes. They are defined using the property() function or by using the @property decorator. Properties are typically used to add custom behavior to attribute access while maintaining a simple syntax.
* Descriptors: Descriptors are objects that define how attribute access is handled at the class level. They implement the \_\_get\_\_, \_\_set\_\_, and \_\_delete\_\_ methods to customize attribute access. Descriptors are more flexible than properties and can be used to implement more complex behaviors, such as validation or lazy loading.

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

Ans: The key differences in functionality between \_\_getattr\_\_, \_\_getattribute\_\_, properties, and descriptors can be summarized as follows:

* \_\_getattr\_\_: Used to handle attribute access for undefined attributes. Called only when the attribute is not found through normal lookup.
* \_\_getattribute\_\_: Called for every attribute access, allowing customization of all attribute access behavior. Use with caution to avoid infinite recursion.
* Properties: Provide a simple way to define custom getter, setter, and deleter methods for attribute access. Primarily used for adding behavior to individual attributes.
* Descriptors: Allow more fine-grained control over attribute access at the class level. Can be used to define behavior that applies to multiple attributes or to implement more complex attribute access patterns.