ASSIGNMENT 10

# Q1

The main difference between \_\_getattr\_\_ and \_\_getattribute\_\_ methods is how they handle attribute access in Python classes:

\_\_getattr\_\_: This method is called when an attribute is not found using the usual lookup mechanism. It is only invoked if the attribute is not present in the instance's dictionary or in its class hierarchy. It allows us to dynamically define or provide a default value for missing attributes.

\_\_getattribute\_\_: This method is called for every attribute access, regardless of whether the attribute exists or not. It is a more general and lower-level method compared to \_\_getattr\_\_. It is often used for implementing attribute access control and customization. However, we need to be careful when implementing \_\_getattribute\_\_ as it can lead to infinite recursion if not properly handled.

# Q2

Properties and descriptors are both ways to control how attributes are accessed on an object. However, they have different implementations.

A property is a simple way to create a read-only or read-write attribute on an object. A property is implemented as a getter and setter method.

A descriptor is a more complex way to create an attribute on an object. A descriptor can be used to implement read-only, read-write, or write-only attributes. Descriptors can also be used to implement custom behaviour when an attribute is accessed.

# Q3

\_\_getattr\_\_: It is called only when an attribute is not found using the usual attribute lookup mechanism. It provides a way to dynamically define or provide a default value for missing attributes.

\_\_getattribute\_\_: It is called for every attribute access, regardless of whether the attribute exists or not. It allows us to customise attribute access and implement attribute access control.

Properties: Properties provide a high-level and intuitive way to define computed or validated attributes. They allow us to define getter, setter, and deleter methods for an attribute and are accessed like regular attributes, providing a more natural syntax for attribute access and modification.

Descriptors: Descriptors provide a lower-level mechanism for attribute access control. They allow us to define classes that can intercept attribute access and modification. Descriptors are explicitly assigned to class attributes using the descriptor protocol and can be reused across multiple attributes or classes.