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

**Q2. What is the difference between properties and descriptors?**

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

**properties and descriptors?**

**SOLUTIONS**

1*. The main difference between* ***\_\_getattr\_\_*** *and* ***\_\_getattribute\_\_*** *is that* ***\_\_getattr\_\_*** *is only called when the attribute is not found by normal lookup in the object's attribute dictionary, while* ***\_\_getattribute\_\_*** *is called for every attribute access, even if the attribute exists in the object's attribute dictionary.*

*2. Properties and descriptors are both ways to define attributes with special behavior in Python. The main difference is that properties are a simpler mechanism for defining read-only or write-only attributes, while descriptors provide more fine-grained control over attribute access, allowing you to define custom behavior for getting, setting, and deleting an attribute.*

*3.* ***\_\_getattr\_\_*** *and* ***\_\_getattribute\_\_*** *are both ways to customize attribute access in Python, but* ***\_\_getattr\_\_*** *is only called for missing attributes, while* ***\_\_getattribute\_\_*** *is called for every attribute access. Properties are a simpler way to define read-only or write-only attributes, while descriptors provide more control over attribute access, allowing you to define custom behavior for getting, setting, and deleting an attribute. Properties are a specific kind of descriptor that only define getter, setter, and/or deleter methods, while descriptors can also define other methods like* ***\_\_get\_\_*** *and* ***\_\_set\_name\_\_****.*