Q1. What is the difference between **getattr** and **getattribute**?

Answer:

getattribute has a default implementation, but **getattr** does not. This has a clear meaning: since **getattribute** has a default implementation, while **getattr** not, clearly python encourages users to implement **getattr** . Python getattr() is a built-in function that is used to return the value of an attribute of a specific object/instance. The Python getattr() function is used to obtain an object's attribute value and also provides the option of executing the default value if the attribute is not available. This method should return the (computed) attribute value or raise an AttributeError exception. In order to avoid infinite recursion in this method, its implementation should always call the base class method with the same name to access any attributes it needs, for example, object.

Q2. What is the difference between properties and descriptors?

Answer:The Cliff's Notes version: descriptors are a low-level mechanism that lets you hook into an object's attributes being accessed. Properties are a high-level application of this; that is, properties are implemented using descriptors.

Q3. What are the key differences in functionality between **getattr** and **getattribute**, as well as properties and descriptors?

Answer: **getattribute** has a default implementation, but **getattr** does not. This has a clear meaning: since **getattribute** has a default implementation, while **getattr** not, clearly python encourages users to implement **getattr** .The getattr() function returns the value of the specified attribute from the specified object.