Q1. Difference between __getattr__ and __getattribute__

__getattr__ is a special method in Python that is called when you try to access an attribute that doesn't exist on an object. It's good for implementing a fallback for missing attributes.
__getattribute__ is another special method that is called whenever an attribute access occurs, regardless of whether the attribute exists or not14. It's invoked before looking at the actual attributes on the object.

Q2. Difference between properties and descriptors

Properties are a high-level application of descriptors. They provide a way to customize access to an attribute in a class. They are implemented using descriptors and are already provided for you in the standard library.
Descriptors are a low-level mechanism that lets you hook into an object's attributes being accessed. They are more flexible, but less convenient, and useful for more advanced use cases, like implementing bound methods, or static and class methods.

Q3. Key differences in functionality between __getattr__ and __getattribute__, as well as properties and descriptors

The key difference between __getattr__ and __getattribute__ is when they are called__getattr__ is only invoked if the attribute wasn't found the usual ways, while __getattribute__ is invoked before looking at the actual attributes on the object.
The key difference between properties and descriptors is their level of application. Properties are a high-level application of descriptors and are already provided for you in the standard library. Descriptors, on the other hand, are a low-level mechanism that provides more flexibility and is useful for more advanced use cases5.