1. What is the concept of an abstract superclass?

2. What happens when a class statement's top level contains a basic assignment statement?

3. Why does a class need to manually call a superclass's \_\_init\_\_ method?

4. How can you augment, instead of completely replacing, an inherited method?

5. How is the local scope of a class different from that of a function?

Answer:

1. An abstract superclass is a class that is designed to be subclassed, but not instantiated on its own. It defines a set of methods and properties that its subclasses must implement, but leaves the details of the implementation up to the subclasses. Abstract superclasses are a way of creating a common interface that multiple related classes can implement, while also providing a way to enforce consistency across those classes.
2. When a class statement's top level contains a basic assignment statement, it creates a class-level attribute that is shared by all instances of the class. This is different from an instance attribute, which is unique to each instance. Class-level attributes can be accessed using the class name followed by the attribute name, and can be modified by any instance of the class.
3. When a class inherits from another class, it needs to manually call the superclass's **init** method to properly initialize the inherited attributes and behavior. This is typically done using the "super()" function, which returns a temporary object of the superclass and allows you to call its methods. By calling the superclass's **init** method, you ensure that any necessary setup is performed before the subclass's **init** method is executed.
4. To augment an inherited method, you can define a new method in the subclass with the same name as the inherited method. Inside this new method, you can call the inherited method using the "super()" function, and then add additional functionality before or after the call to the superclass method. This technique is called method overriding, and allows you to customize the behavior of an inherited method without completely replacing it.
5. The local scope of a class is different from that of a function in several ways. First, a class defines a new namespace that is separate from the global namespace, while a function uses the global namespace by default. Second, a class can access both class-level attributes and instance attributes, while a function only has access to the local and global namespaces. Finally, a class can define methods, which are functions that operate on the class or its instances, while a function cannot.