1. What is the concept of an abstract superclass?

Sol: In python, an abstract superclass is a class that is designed to be inherited from, but not instantiated directly. The purpose of an abstract superclass is to provide a blueprint for subclasses that inherit from it defining a common set of methods and properties that they should implement.

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

Sol: when a class statement’s top level contains an assignment statement, that statement defines a class-level variable.

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

Sol: in python a class needs to manually call a superclass’s \_\_init\_\_ method in order to initialize any attributes or methods defined in the superclass.

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

Sol: In python, we can augment instead of completely replacing an inherited method by using the ‘super()’ function to call the superclass’s method from within the subclass’s method and then modifying the result as necessary.

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

Sol: In python, the local scope of a class is different from that of a function in several ways:

1. Class level variables and methods are defined in the class scope, which is separate from the instance scope. These variables and methods are accessible to all instances of the class.
2. The class scope is accessible from within instance methods using the ‘self.\_\_class\_\_’ attribute.
3. Unlike functions classes can be nested inside other classes creating a nested scope.