1. What is the concept of an abstract superclass?

An abstract class is created to provide a blueprint for any other class that may implement it. This is useful in cases where a module may be used by various members of a team or a third party. A blueprint will allow for proper implementation and ensure the intended functionality remains intact.

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

The variables and the assigned values are made available to all the methods that may be created inside the class

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

This is because python does not implicitly call the \_\_init\_\_ of the parent class as one may simply not require it. Since python supports multiple inheritance it is a user’s choice as to what init method needs to be called

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

We can use super to create a function call of the parent class method inside the child class method to implement the parent method and write the additional code to augment the given code

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

Variables in the scope of class are declared inside the class and outside the function with the indent level being just one below the class declaration itself. Similarly the function scope starts after the function declaration with the indent level for the block of code inside that being one below the function.