1. What is the concept of an abstract superclass?

🡪 Abstract superclass serves as a blueprint or template for its subclasses and defines common attributes and behaviours that the subclasses can inherit and override. The purpose of an abstract superclass is to provide a common interface and functionality to its subclasses while allowing each subclass to implement its own specific details. It encapsulates the shared characteristics and behaviour that are common to multiple related classes.

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

🡪 When a class statement's top level contains a basic assignment statement, it typically defines a class-level variable. This means that the variable is shared among all instances of the class.

A basic assignment statement within the top level of a class is used to define class-level variables or class attributes. These variables are associated with the class itself rather than with specific instances of the class.

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

🡪 If a child class has \_\_init\_\_ method, then it will not inherit the \_\_init\_\_ method of the parent class. in other words, the \_\_init\_\_ method of the child class overrides the \_\_init\_\_ method of the parent class. so we have to manually call a parent superclass's \_\_init\_\_ using super() method

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

🡪 To augment, or extend, an inherited method without completely replacing it, you can use method overriding. Method overriding allows a subclass to provide its own implementation of a method inherited from its superclass while still retaining the original behaviour of the superclass method.

super() method can be used to augment, instead of completely replacing, an inherited method.

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

🡪 A Variable which is defined inside a function is local to that function. it is accessible from the point at which it is defined until the end of the function, and exists for as long as the function is existing.

Similarly a variable inside of a class also has a local variable scope. Variables which are defined in the class body (but outside all methods) are called as class level variables or class attributes. they can be referenced by there bare names within the same scope, but they can also be accessed from outside this scope if we use the attribute access operator (.). on a class or an instance of the class.