**1. What is the concept of an abstract superclass?**

* **Abstract Superclass**:
  + A class that cannot be instantiated directly and is meant to be subclassed.
  + Defines a common interface or structure for its subclasses.
  + Often contains abstract methods (methods without implementation) that must be implemented by subclasses.
  + Example:

from abc import ABC, abstractmethod

class Animal(ABC): # Abstract superclass

@abstractmethod

def sound(self):

pass

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

* **Basic Assignment Statement**:
  + Creates a **class attribute**.
  + Shared by all instances of the class.
  + Example:

class MyClass:

class\_attr = 42 # Class attribute

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

* **Reason**:
  + To ensure that the superclass's initialization logic is executed.
  + Without calling super().\_\_init\_\_(), the superclass's \_\_init\_\_ method is not executed, and its attributes are not initialized.
  + Example:

class SuperClass:

def \_\_init\_\_(self):

self.value = 42

class SubClass(SuperClass):

def \_\_init\_\_(self):

super().\_\_init\_\_() # Manually call superclass's \_\_init\_\_

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

* **Augmenting a Method**:
  + Call the superclass's method using super() and add additional logic.
  + Example:

class SuperClass:

def greet(self):

print("Hello from SuperClass")

class SubClass(SuperClass):

def greet(self):

super().greet() # Call superclass's method

print("Hello from SubClass") # Add additional logic

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

* **Class Local Scope**:
  + Contains class attributes and methods.
  + Variables defined at the top level of the class are class attributes.
  + Example:

class MyClass:

class\_attr = 42 # Class attribute

* **Function Local Scope**:
  + Contains local variables defined within the function.
  + Variables are accessible only within the function.
  + Example:

def my\_function():

local\_var = 42 # Local variable

**Difference**:

* Class attributes are shared across all instances of the class.
* Function local variables are temporary and exist only during the function's execution.