1. What is the concept of an abstract superclass?

**Ans:** Abstract Super Class. A common superclass for several subclasses. Factor up common behavior. Define the methods they all respond to. Methods that subclasses should implement are declared abstract.

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

**Ans:** An assignment statement evaluates the expression list (remember that this can be a single expression or a comma-separated list, the latter yielding a tuple) and assigns the single resulting object to each of the target lists, from left to right.

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

**Ans:** The main reason for always calling base class \_init\_\_ is that base class may typically create member variable and initialize them to defaults. So if you don't call base class init, none of that code would be executed and you would end up with base class that has no member variables.

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

**Ans:** We can augment by using args and kwargs.

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

**Ans:**Local (or function) scope is the code block or body of any Python function or lambda expression. This Python scope contains the names that you define inside the function. These names will only be visible from the code of the function.