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

**Ans:**

An abstract class can be considered as a blueprint for other classes. It allows you to create a set of methods that must be created within any child classes built from the abstract 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:**

It's because one needs to define something that is NOT done in the base-class' \_\_init\_\_ , and the only possibility to obtain that is to put its execution in a derived-class' \_\_init\_\_ function.

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

**Ans:**

A subclass can either completely override the implementation for an inherited method or the subclass can enhance the method by adding functionality to it.

Replacing a Superclass's Method Implementation

Sometimes, a subclass will want to replace entirely its superclass's implementation of a method

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

**Ans:**

Functions do specific things, classes are specific things. Classes often have methods, which are functions that are associated with a particular class, and do things associated with the thing that the class is - but if all you want is to do something, a function is all you need.