1. What is the concept of an abstract superclass?

Ans- 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:- We can do it by writing

instance.method(args...) which is automatically translated by Python into this equivalent form:

class.method(instance, args...)

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

Ans:- In class, when we are calling local variables, we are using