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

Abstract class is a class in which we declare attributes and methods of the class but not implemented. It is kind of blueprint of class where we decide what methods we want in our class before starting our code.

A class which inherits other abstract classes are abstract super class.

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

When first statement in class is a basic assignment statement then it is a static variable or class variable which can be accessible by any class method without pointer.

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

When we want to use attributes and methods of parent class as well then we use super() keyword to inherit the methods and attributes.

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

We can define same method in child class I.e., method overriding and call parent method in it with super() key word as super().method\_name().

class Emp(Person): ####parent class

def \_\_init\_\_(self,age,name,id,sal):

super().\_\_init\_\_(age,name)

self.id=id

self.sal=sal

print("Emp constructor called. . .")

def income(self):

return self.sal

class Manager(Emp): ##### child class

def \_\_init\_\_(self,age,name,id,sal,bonus):

super().\_\_init\_\_(age,name,id,sal)

self.bonus=bonus

print("Manager constructor called. . .")

def income(self):

total=super().income()+self.bonus

return total

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

Variables in functions are accessible throughout the function execution with the same name. But in case of class, only static variables can be accessible with same name which we need to define before constructor. But variables inside constructor are accessible only with pointer.

Local variables of functions are not accessible outside the function body but variables of class can be access with object of class.