**Q1. What is the meaning of multiple inheritance?**

When a class is derived from more than one base class it is called multiple Inheritance. The derived class inherits all the features of the base classes

**Q2. What is the concept of delegation?**

If we need to inherit only few properties not all the properties then we can use delegation. Inheritance will inherit all the propertied , functions and methods. If we want to use few of the properties from parent then we can use delegation.

**Q3. What is the concept of composition?**

one class is container and other class is content and if you delete the container object then all of its contents objects are also deleted. Composition models a “has a” relationship between “Composite” and “Component”.

>>> class **Salary**:

def \_\_init\_\_(self, pay, bonus):

   self.pay=pay

self.bonus=bonus

    def **annual\_salary**(self):

return (self.pay\*12) + self.bonus

>>> class Employee:

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

self.name=name

        self.age=age

        self.obj\_salary=Salary(pay, bonus)

def total\_salary(self):

return self.**obj\_salary.annual\_salary**()

>>> emp = Employee('max', 25, 15000, 10000)

>>> print(emp.total\_salary())

output:-

190000

In our example Employee class acts like Container and Salary class inside the Employee class acts like content in it. Composition represents part-of relationship. In our eg. Salary is part of Employee.

**Q4. What are bound methods and how do we use them?**

A bound method is the one which is dependent on the instance of the class as the first argument. It passes the instance as the first argument which is used to access the variables and functions. In Python 3 and newer versions of python, all functions in the class are by default bound methods

# Python code to demonstrate

# use of bound methods

class A:

    def func(self, arg):

        self.arg = arg

        print("Value of arg = ", arg)

  obj = A()

# bound method

print(obj.func)

**Ex : 2**

class Car:

    # Car class created

    gears = 5

    # a class method to change the number of gears

    @classmethod

    def change\_gears(cls, gears):

        cls.gears = gears

# instance of class Car created

Car1 = Car()

print("Car1 gears before calling change\_gears() = ", Car1.gears)

Car1.change\_gears(6)

print("Gears after calling change\_gears() = ", Car1.gears)

# bound method

print(Car1.change\_gears)

The above code is an example of a [classmethod](https://www.geeksforgeeks.org/classmethod-in-python/). A class method is like a bound method except that the class of the instance is passed as an argument rather than the instance itself. Here in the above example when we call Car1.change\_gears(6), the class ‘Car’ is passed as the first argument.

**Q5. What is the purpose of pseudoprivate attributes?**

It is not the private attribute , this is intended **to avoid namespace collisions in instances, not to restrict access to names in genera**