Q1. What is the relationship between classes and modules?

Sol: In object-oriented programming both classes and modules are used to organize and encapsulate related pieces of code. A class is a blueprint for creating objects. It defines the properties and behaviors that an object of that class will have. A module is a collection of related functions, variables and classes. Modules can be used to organize classes as a class can be defined within a module.

Q2. How do you make instances and classes?

Sol:

We can define classes by class keyword

Ex: class MyClass:

def \_\_init\_\_(self,name):

self.name=name

def greet(self):

print(f”Hello,{self.name}!”)

making instances

EX: obj1 = MyClass(“Mahesh”)

Q3. Where and how should be class attributes created?

Sol: class attributes are created inside the class block but outside of any method, typically below the ‘\_\_init\_\_’ method.

EX: class Car:

color = "red

def \_\_init\_\_(self, make, model):

self.make = make

self.model = model

def description(self):

print(f"This car is a {self.color} {self.make} {self.model}.")

Q4. Where and how are instance attributes created?

Sol: instance attributes are created inside the ‘\_\_init\_\_’ method of a class

EX: class Car:

def \_\_init\_\_(self, make, model, color):

self.make = make

self.model = model

self.color = color

def description(self):

print(f"This car is a {self.color} {self.make} {self.model}.")

Q5. What does the term "self" in a Python class mean?

Sol: In python the ‘self’ keyword is used as the first parameter to a class method to refer to the instance of the class on which the method is called.

Q6. How does a Python class handle operator overloading?

Sol: In python classes can overload operators by implementing special methods that are called when certain operators are used on instances of the class. These special methods have names that start and end with double underscores, such as ‘\_\_add\_\_’ for the addition operator(‘+’)

Q7. When do you consider allowing operator overloading of your classes?

Sol: Operator overloading can be useful in certain situations where it makes the code more readable, intuitive and easier to maintain. We can allow operator overloading when there may Mathematical or scientific calculations, container or collection classes and custom data types.

Q8. What is the most popular form of operator overloading?

Sol: The most popular form of operator overloading in python is probably the arithmetic operators such as ‘+’,’-‘,’\*’,’/’ and ‘\*\*’.

Q9. What are the two most important concepts to grasp in order to comprehend Python OOP code?

Sol: the two most important concepts to grasp in order to comprehend Python OOP code are:

1. Classes and objects
2. Inheritance and polymorphism