Q1. What is the purpose of Python's OOP?

Python's Object-Oriented Programming (OOP) serves to structure code using classes and objects, promoting modularity, reusability, and maintainability. It allows you to create blueprints (classes) for objects, encapsulating data and behavior, fostering code organization and abstraction.

Q2. Where does an inheritance search look for an attribute?

Inheritance search for an attribute begins within the instance itself. If not found, it looks in the class of the instance, and then in its base classes in the order they are defined. This search travels up the class hierarchy until the attribute is found or the end of the hierarchy is reached.

Q3. How do you distinguish between a class object and an instance object?

A class object is a blueprint that defines the structure and behavior of instances. It is like a template. An instance object, on the other hand, is a specific realization of a class, created based on the class blueprint. It holds unique data and can execute methods defined in the class.

Q4. What makes the first argument in a class’s method function special?

The first argument in a class's method function is conventionally named "self." It refers to the instance of the class that the method is being called on. It allows methods to access and modify instance-specific attributes and behavior, connecting the method to the instance it belongs to.

Q5. What is the purpose of the \_\_init\_\_ method?

The purpose of the \_\_init\_\_ method is to initialize the attributes of an instance when it is created. It is a constructor method that gets automatically called when a new instance of a class is created. It allows you to set up the initial state of the object by assigning values to instance variables.

Q6. What is the process for creating a class instance?

To create a class instance, you first define a class with its attributes and methods. Then, you use the class name followed by parentheses to call the class's constructor (\_\_init\_\_) method. This initializes the instance and returns it. For example: my\_instance = MyClass()

Q7. What is the process for creating a class?

To create a class, you define it using the class keyword, followed by the class name and a colon. Inside the class block, you define attributes and methods. Attributes are defined within methods, such as \_\_init\_\_, while methods are functions defined within the class body. Once the class is defined, you can create instances of it.

Q8. How would you define the superclasses of a class?

The superclasses of a class are the classes from which it inherits. In Python, a class can inherit attributes and methods from one or more parent classes (superclasses). These superclasses form the class hierarchy. The class inheriting from another class is called a subclass, and the class being inherited from is called a superclass.