Q1. What is the purpose of Python OOP?

Python Object-Oriented Programming (OOP) allows you to structure your code in a way that models real-world entities using classes and objects. The purpose is to organize code into reusable components, encapsulate data and behavior, promote modularity and maintainability, and enable inheritance and polymorphism.

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

In Python, when an attribute is accessed on an object, the inheritance search looks for the attribute first in the instance's namespace, then in the class's namespace, and finally in the namespaces of its parent classes (in the order specified by the Method Resolution Order, or MRO).

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

A class object is the blueprint from which instances are created. It defines the structure and behavior of instances but does not contain specific data. An instance object, on the other hand, is a concrete realization of a class. It contains actual data and behaves according to the 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 typically named self. It refers to the instance of the class on which the method is called. This allows methods to access and manipulate the instance's attributes and methods.

Q5. What is the purpose of the init method?

The \_\_init\_\_ method is a special method in Python classes used for initialization. It is called automatically when a new instance of the class is created. The purpose of \_\_init\_\_ is to initialize the instance's attributes and perform any necessary setup.

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

To create a class instance in Python, you use the class name followed by parentheses. Optionally, you can pass arguments to the class's \_\_init\_\_ method within the parentheses if the class has one.

Q7. What is the process for creating a class?

To create a class in Python, you use the class keyword followed by the class name and a colon. Inside the class block, you define attributes and methods that describe the behavior of instances of the class.

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

The superclasses of a class are the classes from which it directly inherits. In Python, you specify superclasses by including them in parentheses after the class name in the class definition using inheritance syntax. For example, class MyClass(BaseClass):.