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

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

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

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

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

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

Q7. What is the process for creating a class?

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

Answer:

A1. The purpose of Python's Object-Oriented Programming (OOP) is to provide a way to organize and structure code using objects, which are instances of classes. OOP allows for encapsulation, inheritance, and polymorphism, making code more modular, reusable, and easier to maintain.

A2. In Python, an inheritance search for an attribute starts with the instance itself, then goes up the chain of its class hierarchy, and ultimately reaches the top-level object class.

A3. A class object in Python is the blueprint for creating instances of the class, while an instance object is a specific occurrence or realization of the class, with its own unique state and behavior.

A4. The first argument in a class's method function is conventionally named "self" and refers to the instance of the class that the method is being called on. This argument is automatically passed to the method when it is called and allows the method to access and manipulate the state of the instance.

A5. The **init** method is a special method in Python classes that is called when an instance of the class is created. Its purpose is to initialize the state of the instance by setting its attributes and performing any other necessary setup.

A6. To create a class instance in Python, you first define the class, then use the class name followed by parentheses to call its constructor and create a new instance. The constructor may take arguments that are used to initialize the state of the instance.

A7. To create a class in Python, you define a new class using the "class" keyword, followed by the class name and a colon. Inside the class definition, you define the class's attributes and methods.

A8. The superclasses of a class are the classes from which it inherits attributes and methods. In Python, you can specify the superclasses of a class by including them in a comma-separated list in the class definition's parentheses. The superclasses are searched in order, with the leftmost superclass taking precedence over the others.