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

Ans: - The purpose of Python's Object-Oriented Programming (OOP) is to implement real-world entities like inheritance, polymorphisms, encapsulation, etc. in programming1. The main concept of OOPs is to bind the data and the functions that work on that data together as a single unit so that no other part of the code can access this data.

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

Ans: - An inheritance search in Python looks for an attribute first in the instance object, then in the class the instance was created from, and if not found there, it continues to look in any class that the initial class inherits from

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

Ans: - In Python, an instance object and a class object are essentially the same thing. When you create an instance of a class, that instance is an object3. However, we generally use the term "instance" when we want to discuss the behavior of instances of a specific class or class.

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

Ans: - The first argument in a class's method function in Python is special because it is a reference to the instance of the class. It is typically named self in Python4. It allows you to access or modify the instance's attributes and call other methods.

Q5. What is the purpose of the __init__ method?

Ans: - The purpose of the __init__ method in Python is to initialize the instance's state5. It is automatically called when an instance of the class is created. It allows you to set up attributes with initial values and perform any other setup steps necessary.

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

Ans: -. The process for creating a class instance in Python involves calling the class as if it were a function, passing any arguments that the __init__ method accepts. This creates a new instance of the class.

Q7. What is the process for creating a class?

Ans: - The process for creating a class in Python involves using the class keyword, followed by the class name and a colon. Then, you define the class body, which includes any methods and attributes.

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

Ans: - The superclasses of a class in Python are defined in the class definition. When you define a class, you can specify one or more superclasses in parentheses after the class name. The class then inherits the attributes and methods of the superclasses