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

A1. Python's Object-Oriented Programming (OOP) allows for the creation of objects that have properties and behaviors. This allows for code to be organized into modular, reusable components that can be easier to understand and maintain.

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

A2. When Python is searching for an attribute in an inheritance hierarchy, it first looks in the current object instance, then in the class of the instance, and finally in any superclasses of the class.

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

A3. A class object in Python represents the class definition itself, while an instance object is a specific instance of that class. In other words, the class object describes the properties and methods that all instances of the class will have, while an instance object represents a unique set of values for those properties.

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

A4. The first argument in a class's method function is conventionally called "self" and refers to the instance of the class that the method is being called on. This allows the method to access and modify the instance's properties and behaviors.

Q5. What is the purpose of the init method?

A5. The init method in a Python class is a special method that is called when a new instance of the class is created. Its purpose is to initialize the instance's properties with any values that are passed as arguments, and to perform any other setup tasks that are required.

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

A6. To create a new instance of a class in Python, you simply call the class like a function, passing any required arguments. For example, if you have a class called "Person", you can create a new instance of it by calling "my\_person = Person(name='Alice', age=30)".

Q7. What is the process for creating a class?

A7. To create a new class in Python, you use the "class" keyword, followed by the name of the class and a colon. Inside the class block, you define any properties and methods that the class should have.

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

A8. The superclasses of a class in Python are the classes that it inherits from, either directly or indirectly. To define the superclasses of a class, you include them in the class definition using parentheses after the class name. For example, if you have a class called "Child" that inherits from a class called "Parent", you would define it like this: "class Child(Parent):".