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

The purpose of Python's Object-Oriented Programming (OOP) is to enable the structuring of code around objects, which bundle together both data and functions that operate on that data. This approach promotes modularity, reusability, and easier maintenance of code by organizing it into classes and objects, facilitating abstraction, encapsulation, inheritance, and polymorphism.

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

Inheritance searches for an attribute first in the instance itself, and if not found, it looks in the class it inherits from, and further up the inheritance chain

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 objects, while an instance object is a specific creation based on that blueprint. Class objects define attributes and methods, while instance objects hold unique data values and can call 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, conventionally named **self**, refers to the instance of the class on which the method is being called. It allows the method to access and manipulate the instance's attributes and methods.

Q5. What is the purpose of the __init__ method?

The **__init__** method in a class is used to initialize and set up the initial state of an instance when it's created. It allows you to assign values to attributes and perform any necessary setup tasks when a new object of the class is instantiated.

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

Creating a class instance involves two main steps:

1. Instantiation: Use the class name followed by parentheses to create an instance. This invokes the class's __init__ method, setting up initial attributes and state.
2. Initialization: The __init__ method runs, taking the instance itself (self) and any provided arguments. It initializes attributes and performs setup tasks.

Q7. What is the process for creating a class?

Creating a class involves defining its structure and behavior:

1. Class Definition: Use the class keyword followed by the class name and a colon to define the class's structure.
2. Attributes and Methods: Inside the class, define attributes (variables) and methods (functions) that describe the class's behavior.
3. Initialization: Define the __init__ method to set up initial attributes when instances are created.
4. Instance Methods: Define other methods that can be called on instances of the class.

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

The superclasses of a class, also referred to as parent classes or base classes, are the classes from which the current class inherits attributes and methods. A class can inherit from one or more superclasses, allowing it to reuse and extend the functionality defined in those superclasses.