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

Python's Object-Oriented Programming (OOP) allows you to structure your code into reusable and modular components. It provides a way to define objects that can encapsulate data (attributes) and behaviors (methods), promoting code organization, readability, and maintainability. OOP principles like inheritance, encapsulation, and polymorphism help in building complex systems efficiently.

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

When an attribute is accessed on an instance, the inheritance search for that attribute follows the Method Resolution Order (MRO).

The search order is as follows:

1. **Instance itself**: Python first looks for the attribute in the instance's own dict.
2. **Class of the instance**: If not found, it checks the class of the instance.
3. **Parent classes**: It then searches through the base classes of the instance's class, following the MRO.
4. **Superclasses**: Finally, it continues up the inheritance chain through all parent classes, following the MRO, until the attribute is found or the chain ends.

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

A class object defines the structure and behavior, while an instance object is a concrete realization of that structure with its own data.

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

The first argument in a class’s method function is special because it refers to the instance of the class on which the method is being called. By convention, this argument is named self.

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

The \_\_init\_\_ method in Python is a special method, also known as a constructor. Its primary purpose is to initialize a new instance of a class when it is created. This method allows you to set up the initial state of an object by assigning values to instance variables or performing any other setup actions required.

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

Define the class.

Call the class with any necessary arguments.

The instance is created and initialized.

Use the instance.

Q7. What is the process for creating a class?

Start with class and name your class.

Define \_\_init\_\_ if you need to initialize attributes.

Add methods to define the behavior of instances.

Include variables if needed (either class-level or instance-level).

Use the class by creating instances and interacting with them.

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

We define the superclasses (also known as parent classes) of a class by specifying them in parentheses after the class name in the class definition. This establishes an inheritance relationship, where the new class (subclass) inherits attributes and methods from the superclass(es)