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?

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

In Python, object-oriented Programming (OOPs) is a programming that can use objects and classes in it. Its main purpose to implement real-world entities like inheritance, polymorphism, encapsulation in the programming. The main use of OOPs is to create reusable code and bind the data and the functions that work on that 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?

The first occurrence of attribute by looking in object, then in all classes above it, from bottom to top and left to right. An attribute fetches are simply tree searches. The term inheritance is applied because objects lower in a tree inherit attributes attached to objects higher in that tree. As the search proceeds from the bottom up, in a sense, the objects linked into a tree are the union of all the attributes defined in all their tree parents, all the way up the tree.

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

A class object can have a only one object which contains static variables and it can be created at the time of class creation.

There can be multiple instance objects and we need to call instance objects explicitly by calling class name along with parenthesis. In Instance object \_\_init\_\_ method gets called implicitly by passing first argument as the object itself.

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

The first argument in method is self-variable that refers to the object itself.

The reason you need to use self. is because Python does not use special syntax to refer to instance attributes. Python decided to do methods in a way that makes the instance to which the method belongs be passed automatically, but not received automatically: the first parameter of methods is the instance the method is called on.

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

. "\_\_init\_\_" is a reserved method in python classes. It is called as a constructor in object oriented terminology. This method is called when an object is created from a class and it allows the class to**initialize the attributes of the class**.

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

To create instances of a class, we call the class using class name and pass in whatever arguments its \_\_init\_\_ method accepts. Class variable would be accessed using class name as follows –

e.g.

Class dog:

Pass

a= dog();

Q7. What is the process for creating a class?

We can give any name which is relevant to your problem statement to Class name.

Class Student:

Pass

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

A superclass is the**class from which many subclasses can be created**. The subclasses inherit the characteristics of a superclass. The superclass is also known as the parent class or base class. Car, Truck and Motorcycle are all subclasses of the superclass Vehicle. They all inherit common attributes from vehicle such as speed, colour etc. while they have different attributes also i.e Number of wheels in Car is 4 while in Motorcycle is 2.