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

The main purpose of OOP is to bind both data and function that work on that together as a single unit so that no other part of the code can access this data. The use OOP is useful for:

1. Easy troubleshooting of problems whenever error occurs
2. Repeated use of code through inheritance
3. Flexibility of using functions in polymorphism

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

An inheritance occurs when a child class is formed; hence automatically the search attribute will be of parent class.

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

An instance object is unique to its own instances whereas a class object is for all methods and instances inside that class.  Instance variables are variables whose value is assigned inside a constructor or method with self whereas class variables are variables whose value is assigned in the class.

class room:

    # Class Variable

    room\_width = 15

room\_len = 35

    # The init method or constructor

    def \_\_init\_\_(self, bed\_size, table\_type, sofa\_type):

        # Instance Variable

        self.bed\_size = bed\_size

        self.table\_type = table\_type

self.sofa\_type = sofa\_type

Room1= room(‘king’,’large’,’L-shaped’)

print(“room details are:)

print(“room width:”, Room1.room\_width)

print(“room length:”, Room1.room\_len)

print(“bed size:”, Room1. bed\_size)

print(“table size:”, Room1. table\_size)

print(“sofa size:”, Room1. sofa\_size)

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

Since class is a blueprint for objects, the first argument allows the access to the attributes and methods of each object. Multiple number of objects are given access through this argument (self). If there is no self argument the same class could not hold information of one or more objects that are created.

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

Its main objective is it allows class to initialize the initial state of object instances created. The \_\_init\_\_() method is called the constructor and is always called when an object is created.

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

class room:

    # Class Variable

    room\_width = 15

room\_len = 35

    # The init method or constructor

    def \_\_init\_\_(self, bed\_size):

        # Instance Variable

        self.bed\_size = bed\_size

To create instances of a class, you call the class using class name and pass in whatever arguments its *\_\_init\_\_* method accepts.

**Room1= room(king\_size) #creating instances of class**

print(“room details are:)

print(“room width:”, Room1.room\_width)

print(“room length:”, Room1.room\_len)

print(“bed size:”, Room1. bed\_size)

**Q7. What is the process for creating a class?**

Type class and add any keyword followed by :. For creating new class from existing we use inheritance concept.

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

A class that is derived from another class is called a subclass (also a derived class, extended class, or child class). The class from which the subclass is derived is called a superclass (also a base class or a parent class).