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

object-oriented Programming (OOPs) is a programming paradigm that uses objects and classes in programming. It aims **to implement real-world entities like inheritance, polymorphisms, encapsulation, etc.** **in the programming**

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

### An inheritance search looks for an attribute first in the instance object, then in the class the instance was created from, then in all higher superclasses, progressing from left to right (by default). The search stops at the first place the attribute is found.

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

### Classes are a kind of factory for creating multiple instances. Classes also support operator overloading methods, which instances inherit, and treat any functions nested in the class as methods for processing instances.

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

### It always receives the instance object that is the implied subject of the method call. It’s usually called 'self' by convention.

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

### If the \_\_init\_\_ method is coded or inherited in a class, Python calls it automatically each time an instance of that class is created. class New(): def \_\_init\_\_(self, arg1, arg2): self.first\_var = arg1 self.second\_var = arg2

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

### You create a class instance by calling the class name as though it were a function; any arguments passed into the class name show up as arguments two and beyond in the \_\_init\_\_ constructor method. x = ClassName() y = AnotherClass(arg1, arg2)

Q7. What is the process for creating a class?

### class ClassName():

### def functionname():

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

### They are classes which are used to inherit from. class Son(Mother, Father): ... In this case Mother and Father are superclasses for Son subclass.