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

OOP is the practice of writing code in the form of small objects and classes. This allows the code written to be reusable and easier to understand. These are the main concepts of OOP:

1. Classes: a class is a collection of objects, a logical entity that contains some attributes and methods
2. Objects: objects are the real-world entities associated with the classes and methods.
3. Polymorphism: This allows multiple methods and functions to have similar name but different signature that can perform a different operation
4. Inheritance: Inheritance allows for reusability of code and addition of more functionality of code to existing classes and methods without having to modify the original code.
5. Encapsulation: OOP allows for creation of wrapping the data and methods for the data in one unit. This allows for imposing restrictions to code which can increase security. Public, Private and Protected tiers of objects and variables can be created to specify the access level so that sensitive data cannot be accessed or modified by an outside entity either by accident or malicious intent.
6. Data Abstraction: this is used to hide unnecessary or sensitive code details from the user. Code can still be used and implemented but the details and actual code is hidden using abstract classes

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

An inheritance search looks for an attribute inside the instance object first, followed by the class the instance was created from and then in all the super classes the class may have inherited

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

A class object is declared inside the class but outside any of the function implementations

An instance object is created inside the \_\_init\_\_ function of the class and are determined when an instance is created

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

The first argument in a method is the instance of the class itself. The word ‘self’ is conventionally used for this purpose. This allows the class access to all attributes and objects in python. This is used to associate the attributes of the object to the class using the instance of the class which in this case is ‘self’

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

The \_\_init\_\_ method of the class is used to initialize the data members of the class when an instance of the class is created. It can also contain a collection of statements that are executed as soon as an instance of the class is created

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

classname(parameters)

where classname is the Name of the class whose instance is to be created and parameters are any that are to be passed that were declared in the \_\_init\_\_ method of the class

Q7. What is the process for creating a class?

class classname:

def \_\_init\_\_(self, \*args):

self.args = args

where class and def are keywords used for creation of classes and methods and init method is used for passing parameters required for creating instance of the class

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

A superclass of a class is the class which is inherited by the sub class. This would allow for access to all the methods and variables of the superclass to the sub class.