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

Ans:

- Implements inheritance, polymorphisms, encapsulation, etc in programming.
- Easy to understand and efficient programs.
- Reusable code.
- Safe and secure data.
- Polymorphism allows the same interface for different objects.

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

Ans: Inheritance search looks for an attribute in a bottom to top, looking for the lowest occurrence of the attribute and then looks in the class it is generated from, to all super classes listed in its class header.

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

Ans: Updating class obejcts updates instance objects whereas any changes to instance objects doesn't affect class objects. A class object is passed as a method's first argument while in case of an instance object, it is automatically passed as the first argument to the instance method.

4. What makes the first argument in a class's method function special?

Ans: the first parameter of a function in class is always a reference to the current instance of the class. By convention, this argument is always named self. The calling process is automatic while the receiving process is not (its explicit).

5. What is the purpose of the __init__ method?

Ans: __init__ serves as a constructor in oop. It is called when an object is created from a class and allows the class to initialize the attributes of the class

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

Ans: call the class using class name and pass arguments its __init__ method accepts

7. What is the process for creating a class?

Ans: Use keyword class

Example: class New:

```
x = 0
```

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

Ans: Superclass is given as a arugment to the derived class

Example: class Newer(New):

Here New is the superclass and new