ANSWER:-

1. Classes in python are templates for creating objects. They contain variables and functions which define the class objects. At the same time, modules are python programs that can be imported into another python program.
2. The first step to create an instance method is to [create a Python class](https://codefather.tech/blog/python-class-definition/) using the class keyword. **The class keyword allows to define a new class in Python. By defining a new class you create a new data type.** Once you create a class you can define an instance method inside that class using a similar syntax you would use to define a Python function (using the **def keyword**).
3. Class attributes are defined directly within the class definition and are shared among all instances of the class. They can be accessed using the class name and through an instance of the class.
4. Instance attributes are created within the \_\_init\_\_ method of the class. They are unique to each object and are assigned values specific to that instance.
5. The term “self” refers to the instance of the class that is currently being used. It is customary to use “self” as the first parameter in instance methods of a class. Whenever you call a method of an object created from a [class](https://www.geeksforgeeks.org/python-classes-and-objects/), the object is automatically passed as the first argument using the “self” parameter.
6. **Operator Overloading**  means giving extended meaning beyond their predefined operational meaning. For example operator + is used to add two integers as well as join two strings and merge two lists.
7. We might allow want operator overloading in a class **when it makes the code more intuitive, readable, and easier to use** by allowing users to perform operations using standard operators.
8. The most popular form of operator overloading in Python is to give extended meaning beyond their predefined operational meaning.
9. To comprehend Python OOP code, it is essential to understand the

following two concepts:

**Classes**: A class is a blueprint for creating objects that have similar attributes and methods. It is a user-defined data type that encapsulates data and functions that operate on that data.

**Objects**: An object is an instance of a class. It is a unique entity that has a state and behavior. The state of an object is represented by its attributes, and the behavior is represented by its methods.

Understanding these two concepts is crucial to writing and comprehending Python OOP code. [Once you have a good grasp of these concepts, you can move on to other OOP concepts such as inheritance, polymorphism, encapsulation, and abstraction](https://www.geeksforgeeks.org/python-oops-concepts/).