Q1. What is the relationship between classes and modules?

🡪- Classes can be defined within a module. You can have one or more classes in a module, along with other code.

- Modules can import classes defined in other modules to make use of their functionality.

- Classes provide a way to organize and encapsulate code within a single object, while modules provide a way to organize and encapsulate code within a file.

- Modules serve as a higher-level organizational unit, allowing you to group related classes, functions, and variables together.

- Modules can be imported into other modules or scripts, making their contents accessible for use.

Q2. How do you make instances and classes?

🡪 For creating a class instance. we call a class by its name and pass the arguments which its \_\_init\_\_ method accepts.

For creating a class, we use the Class keyword. Class keyword is followed by classname and semicolon.

Q3. Where and how should be class attributes created?

🡪 Class attributes or Class level Attributes belong to the class itself. these attributes will be shared by all the instances of the class. Hence these attributes are usually created/defined in the top of class definition outside all methods.

Q4. Where and how are instance attributes created?

🡪Instances attributes are passed to the class when an object of the class is created. Unlike class attributes, instance attributes are not shared by all objects of the class. instead, each object maintains its own copy of instance attributes at object level. whereas in case of class attributes all instances of class refer to a single copy.

Q5. What does the term "self" in a Python class mean?

🡪 self represents the instance of the class (it represents the object itself). By using the “self” keyword we can access the attributes and methods of the class with in the class in python. It binds the attributes with the given arguments.

Q6. How does a Python class handle operator overloading?

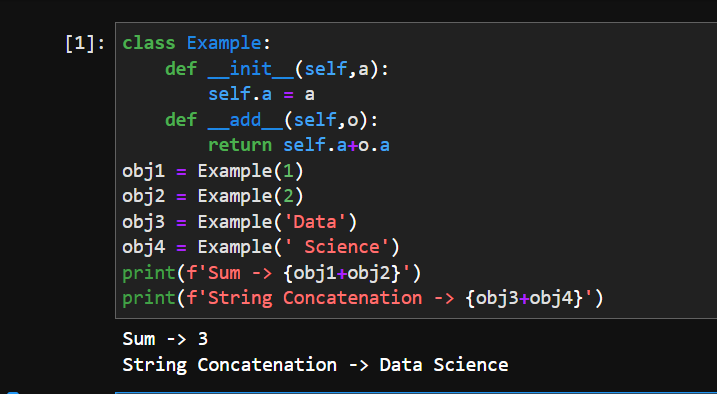
🡪Operator overloading allows classes to define the behaviour of operators such as `+`, `-`, `\*`, `/`, `==`, `!=`, and many others. By implementing special methods, also known as magic methods, a class can specify how operators should work with instances of that class.

Q7. When do you consider allowing operator overloading of your classes?

🡪 When we want to have different meaning for the same operator according to the context we use operator overloading.

Q8. What is the most popular form of operator overloading?

🡪 The most popular form of operator overloading in python is by special methods called Magic methods. Which usually beign and end with double underscore \_\_<method name>\_\_.



Q9. What are the two most important concepts to grasp in order to comprehend Python OOP code?

🡪 Classes and objects are the two concepts to comprehend python OOP code as more formally objects are entities that represent instances of general abstract concept called class