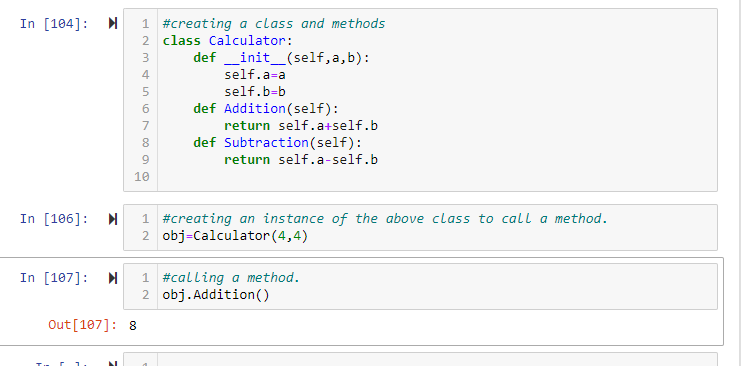
Q1. What is the relationship between classes and modules?

Ans: **modules(.py file) can contain multiple classes. In order to create an instance of a class, first module needs to be imported.**

Q2. How do you make instances and classes?

Ans: **To construct a class, ‘class’ keyword has to be used. Below is an example of how to create a class and an instance of the class.**



Q3. Where and how should be class attributes created?

Ans:  **attributes should be created when we use a constructor like below. These attributes must be passed into the instance of the class.**

Graphical user interface, text, application, email

Description automatically generated

**Here ‘a’ and ‘b’ are attributes which must be passed in while creating an instance of the class.**

Q4. Where and how are instance attributes created?

Ans: **Instance attributes are created while creating an instance of a class.**

Graphical user interface, text, application

Description automatically generated

**The highlighted part in the above screenshot shows how attribute values are passed in while creating an instance of the class.**

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

**Ans**: **‘self’ is a pointer which helps the instance to connect to the class and its methods.**

Q6. How does a Python class handle operator overloading?

Ans: **This operator overloading is handled by the built-in classes only like list, string or number and each class tends to behave differently with operator overloading such as ‘+’ operator.**

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

Ans: **when you want the operator to work differently for different classes.**

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

Ans:  **‘+’ operator is the most popular example of operator overloading where it behaves differently for object of different classes. For example, this operator can be used to merge two lists or join two strings or add two integers.**

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

Ans: a) **Classes and Objects**

b) **Inheritance and Polymorphism**