Q1. Which two operator overloading methods can you use in your classes to support iteration?

Q2. In what contexts do the two operator overloading methods manage printing?

Q3. In a class, how do you intercept slice operations?

Q4. In a class, how do you capture in-place addition?

Q5. When is it appropriate to use operator overloading?

SOLUTIONS

*1. The two operator overloading methods that can be used in classes to support iteration are* ***\_\_iter\_\_()*** *and* ***\_\_next\_\_()****.*

*2. The* ***\_\_str\_\_()*** *method is used to manage printing in the context of converting an object to a string representation, while the* ***\_\_repr\_\_()*** *method is used to manage printing in the context of providing a detailed and unambiguous representation of an object for debugging and development purposes.*

*3. To intercept slice operations in a class, you can define the* ***\_\_getitem\_\_()*** *method with the appropriate logic to handle the slicing parameters.*

*4. To capture in-place addition in a class, you can define the* ***\_\_iadd\_\_()*** *method.*

*5. Operator overloading can be appropriate when you want to define custom behaviors for operators that are applied to instances of your class, allowing you to create more expressive and intuitive APIs for your objects. However, it should be used judiciously and with care, as overuse can lead to code that is difficult to read and understand.*