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

ANS:

#### 1. Overloading User-Defined Functions

User-defined [function overloading can be achieved](https://www.educba.com/function-overloading-in-java/) in python by setting a parameter or argument value as none. So if an argument is set as none, then the function reacts in one manner when a value is not passed for this argument and in a different way when a value is not passed for the same argument.

#### 2. Overloading Default or Pre-Defined Functions

Overloading built-in functions involve defining the pre-defined function, which is expected to be overloaded in the python class as a special function. So when the pre-defined function is declared a special function in the Python class, the interpreter will use this special function as the declaration for the pre-defined call. The below example explains this process precisely.

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

ANS:

### Printing Your Objects Prettily Using str()

The str() built-in is used to cast an instance of a class to a str object, or more appropriately, to obtain a user-friendly string representation of the object which can be read by a normal user rather than the programmer. You can define the string format your object should be displayed in when passed to str() by defining the \_\_str\_\_() method in your class. Moreover, \_\_str\_\_() is the method that is used by Python when you call [print()](https://realpython.com/python-print/) on your object.

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

ANS:

The \_\_getitem\_\_ method is used for accessing list items, array elements, dictionary entries etc. slice is a constructor in Python that creates slice object to represent set of indices that the range(start, stop, step) specifies. \_\_getitem\_\_ method can be implement in a class, and the behavior of slicing can be defined inside it.

Syntax:

\_\_getitem\_\_(slice(start, stop, step))

Parameter:

* slice() : constructor to create slice object.
* start: An integer number specifying start index.It is optional and default is 0.
* stop: An integer number specifying end index.
* step: An integer number specifying the step of slicing. It is optional and  
  default is 1.

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

ANS:

Python provides the operator x += y to add two objects in-place by calculating the sum x + y and assigning the result to the first operands variable name x . You can set up the in-place addition behavior for your own class by overriding the magic “dunder” method \_\_iadd\_\_(self, other) in your class definition.

Q5. When is it appropriate to use operator overloading?

ANS:

It allows us to provide an intuitive interface to our class users, plus makes it possible for templates to work equally well with classes and built-in types. Operator overloading allows C++ operators to have user-defined meanings on user-defined types or classes.