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

Ans. \_\_iter\_\_ and \_\_next\_\_ are the operator overloading methods in python that support iteration and are collectively called iterator protocol.

\_\_iter\_\_ returns the iterator object and is called at the start of loop in our respective class.

\_\_next\_\_ is called at each loop increment, it returns the incremented value. Also Stopiteration is raised when there is no value to return.

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

Ans. \_\_str\_\_ and \_\_repr\_\_ are two operator overloading methods that manage printing.

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

Ans. In a class use of slice() in \_\_getitem\_\_ method is used for intercept slice operation. This slice method is provided with start integer number, stop integer number and step integer number.

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

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

Ans. In a class \_\_iadd\_\_ method is used for this in-place operation. This function is used to assign the current value and add them. This operator does x+=y operation.

Q5. When is it appropriate to use operator overloading?

Ans. Operator overloading is appropriate to provide a special meaning of an operator for a user-defined data type. When we need the specification of user-defined implementation for operations wherein one or both operands are of user-defined class or structure type, operator overloading is needed.