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

**Ans:**

\_\_iter\_\_ and \_\_next\_\_

The \_\_iter\_\_ returns the iterator object and is implicitly called at the start of loops. The \_\_next\_\_ method returns the next value and is implicitly called at each loop increment. \_\_next\_\_ raises a StopIteration exception when there are no more value to return, which is implicitly captured by looping constructs to stop iterating.

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

**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))

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

**Ans:**

Python in its definition provides methods to perform inplace operations, i.e doing assignment and computation in a single statement using “operator” module. For example,

x += y is equivalent to x = operator.iadd(x, y)

1. iadd() :- This function is used to assign and add the current value. This operation does “a+=b” operation. Assigning is not performed in case of immutable containers, such as strings, numbers and tuples.

**Q5. When is it appropriate to use operator overloading?**

**Ans:**

Operator Overloading means giving extended meaning beyond their predefined operational meaning. For example operator + is used to add two integers as well as join two strings and merge two lists. It is achievable because '+' operator is overloaded by int class and str class