**Assignment 4**

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

Ans: We can overload \_\_iter\_\_(self) and \_\_next\_\_(self) methods in the class to support iteration.

\_\_iter\_\_() method is used to convert the argument into iterator and then by using the \_\_next\_\_() method to produce next values.

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

Ans: Two operator overloading methods which manage printing are \_\_str\_\_() and \_\_repr\_\_(). \_\_str\_\_() method is used to describe a class’s object which is more user friendly than the \_\_repr\_\_() method which is more developer friendly. \_\_repr\_\_() method is generally used for logging or debugging purposes.

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

Ans: Slice operations can be intercepted by using \_\_setitem\_\_(self) and \_\_getitem\_\_() methods in class.

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

Ans: To capture the in-place addition in a class we can use \_\_iadd\_\_() method, which will perform the in-place addition and return the modified result.

Q5. When is it appropriate to use operator overloading?

Ans: Operator overloading is used to modify the already existing operator functions. It is very useful if we have a really simple modification to the operator function but it can easily become confusing if it is very complex. Hence, it becomes hard to understand for other developers.