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

**using \_\_iter\_\_ and \_\_next\_\_ methods.**

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

# Python program to show use of

# + operator for different purposes.

print(1 + 2)

# concatenate two strings

print("Geeks"+"For")

# Product two numbers

print(3 \* 4)

# Repeat the String

print("Geeks"\*4)

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

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

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.

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

import operator

# using iadd() to add and assign value

x = operator.iadd(2, 3);

Q5. When is it appropriate to use operator overloading?

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. You might have noticed that the same built-in operator or function shows different behavior for objects of different classes, this is called *Operator Overloading*.