Q1. Is an assignment operator like += only for show? Is it possible that it would lead to faster results at the runtime?

**Ans:** Speed wise there is a very marginal difference, but += is definitely a shortcut for adding in same number again and again.

Q2. What is the smallest number of statements you'd have to write in most programming languages to replace the Python expression a, b = a + b, a?

**Ans:** minimum 2 statements.

Q3. In Python, what is the most effective way to set a list of 100 integers to 0?

**Ans**:

l=[0]

l=l\*10

Q4. What is the most effective way to initialise a list of 99 integers that repeats the sequence 1, 2, 3? S If necessary, show step-by-step instructions on how to accomplish this.

**Ans:**

Using list comprehension - l=[i for i in range(1,100)]

Q5. If you're using IDLE to run a Python application, explain how to print a multidimensional list as efficiently?

**Ans:**

Print(l), will print list l which can be multidimensional.

Q6. Is it possible to use list comprehension with a string? If so, how can you go about doing it?

**Ans:**

string="hello"

l=[i for i in string]

Q7. From the command line, how do you get support with a user-written Python programme? Is this possible from inside IDLE?

**Ans:** Yes, IDLE will be a best solution approach for this, we are having debugging and other program navigation tools in IDLE.

Q8. Functions are said to be “first-class objects” in Python but not in most other languages, such as C++ or Java. What can you do in Python with a function (callable object) that you can't do in C or C++?

**Ans:**

Python functions are having no restrictions on the type of parameters or return value.

We can return multiple values from python functions.

With the help of python functions, we can create generators as well.

Q9. How do you distinguish between a wrapper, a wrapped feature, and a decorator?

**Ans:**

**Wrappers** around the functions are also knows as decorators which are a very powerful and useful tool in Python since it allows programmers to modify the behaviour of function or class. Decorators allow us to wrap another function in order to extend the behaviour of the wrapped function, without permanently modifying it. In Decorators, functions are taken as the argument into another function and then called inside the wrapper function.

Q10. If a function is a generator function, what does it return?

**Ans:** It returns the next value of sequence.

Q11. What is the one improvement that must be made to a function in order for it to become a generator function in the Python language?

**Ans:**

Using “yield” keyword.

Q12. Identify at least one benefit of generators.

**Ans:**

Memory efficient, time efficient, and local variables and their states are remembered between successive calls.