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

It all depends on the data type. Assigning booleans is faster than first comparing them. But that may not be true ...

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 = Python number method min() returns the smallest of its arguments: the value closest to negative infinity.

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

Ans = Simple loop with append : my\_list = [] for i in range(50): my\_list.append(0)

Simple loop with += : my\_list = [] for i in range(50): my\_list += [0]

List comprehension: my\_list = [0 for i in range(50)]

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.

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

Ans = In Python, you can start indexing from the end of an iterable. This is known as negative indexing. This means you can use both positive and negative indexes to access iterables.

A

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

List comprehension in Python is an easy and compact syntax for creating a list from a string or another list. It is a very concise way to create a new list by performing an operation on each item in the existing list. List comprehension is considerably faster than processing a list using the for loop.

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

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 behavior of function or class. Decorators allow us to wrap another function in order to extend the behavior of the wrapped function, without permanently modifying it

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

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 = It is fairly simple to create a generator in Python. It is as easy as defining a normal function, but with a yield statement instead of a return statement. If a function contains at least one yield statement (it may contain other yield or return statements), it becomes a generator function.

Q12. Identify at least one benefit of generators.

Ans = They provide a continuous flow of power, especially since much of our daily lives revolve around electricity. Even when the power is out, you can run appliances like air conditioners, refrigerators, heaters, washers/dryers and lights when you have a generator.