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

***Ans***: No, the += operator is not just for show. In fact, it can lead to faster results at runtime in certain cases.

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***: temp = a

a = b

b = temp

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

***Ans***:

The most effective way to set a list of 100 integers to 0 in Python is to use the list slicing notation to replace all elements of the list with 0.

my\_list = [1, 2, 3, 4, 5, ... , 100] # create a list of 100 integers

my\_list[:] = [0] \* 100 # replace all elements of the list with 0

print(my\_list)

output:

[0, 0, 0, 0, 0, 0, 0, 0, 0, ……, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

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***:

The most effective way to initialize a list of 99 integers that repeats the sequence 1, 2, 3 in Python is to use the list slicing notation along with the itertools module.

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

***Ans***:

To print a multidimensional list efficiently in Python using IDLE, you can use a nested loop to iterate over the rows and columns of the list and print each element on the same line.

my\_list = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]

for row in my\_list:

for col in row:

print(col, end=" ")

print()

output:

1 2 3

4 5 6

7 8 9

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

***Ans***:

Yes, it is possible to use list comprehension with a string in Python.

my\_string = "hello world"

my\_list = [char for char in my\_string]

print(my\_list)

output: ['h', 'e', 'l', 'l', 'o', ' ', 'w', 'o', 'r', 'l', 'd']

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

***Ans***:

you can get support with a user-written Python program by running the program with the --help or -h option. This option is commonly used to display a help message that explains how to use the program and provides information about its command line arguments.

in IDLE, you can provide help to users by adding comments to your code that explain how to use the program and provide information about its functions and variables. You can also use the help () function to provide interactive help from within the IDLE environment.

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***:

Followings are a few things of what you can do in Python with functions that you can't do in C or C++:

1. Functions as arguments: you can create a function that takes another function as an argument and applies it to a list of values.
2. Functions as return values: you can create a function that returns a function that applies a specified mathematical operation to a given input value.
3. Function decorators: you can create a function decorator that logs the input and output of a function.

4. Closures: This allows you to create functions that have state and can be used to implement more complex behaviour.

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

***Ans***:

1.a wrapper provides a simple and convenient interface to a more complex or low-level function or class,

2.a wrapped feature extends or modifies the behaviour of a function or method without changing its source code.

3.a decorator modifies or extends the behaviour of a function or method by returning a modified version of the function as output.

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

***Ans***:

A generator function returns an iterator object that can be used to generate a sequence of values, one at a time, using the yield keyword.

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***:

To convert a regular function into a generator function in Python, the only improvement that needs to be made is to replace the return statement with a yield statement.

Q12. Identify at least one benefit of generators.

***Ans***:

Generators offer a powerful and flexible way to generate sequences of values in a memory-efficient and performant way, making them a valuable tool for data processing, scientific computing, and other applications.