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

Ans. A=A+1 evaluates to finding A, adding 1 to it. Then storing the value again in variable A. This expression makes Python to look for memory holder of a twice. But A+=1 simply means value of A is to incremented by 1. As memory address has to be identified once, += leads to faster operation.

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 number of lines required to write above code in languages other Python will be 4, two for assigning initial values for variables a and b, and two for reassignment i.e. a=a+b and b=a.

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 by using repition operator(\*) or by using list comprehension.

METHOD:

list\_zero=[0]\*100

print(list\_zero)

OUTPUT:

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 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.

METHOD:

my\_list = [1,2,3]\*33

print(my\_list)

OUTPUT:

[1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3]

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

Ans.

METHOD:

my\_list = [[1,1],[2,2],[3,3],[4,4],[5,5]]

for x in range(len(my\_list)):

for y in range(len(my\_list[x])):

print(my\_list[x][y],end=" ")

OUTPUT:

1 1 2 2 3 3 4 4 5 5

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.

METHOD:

my\_list = [ele for ele in 'iNeuron']

print(my\_list)

OUTPUT:

['i', 'N', 'e', 'u', 'r', 'o', 'n']

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

Ans. Get support with a user-written Python Programme: Start a command prompt (Windows) or terminal window (Linux/Mac). If the current working directory is the same as the location in which you saved the file, you can simply specify the filename as a command-line argument to the Python interpreter.

Get support with a User-written Python Program from IDLE: You can also create script files and run them in IDLE. From the Shell window menu, select File → New File. That should open an additional editing window. Type in the code to be executed. From the menu in that window, select File → Save or File → Save As… and save the file to disk. Then select Run → Run Module. The output should appear back in the interpreter.

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. The tasks which can be performed with the functions in python are:

-- A function is an instance of the Object type.

-- You can store the function in a variable.

-- You can pass the function as a parameter to another function.

-- You can return the function from a function.

-- You can store them in data structures such as hash tables, lists,

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

Ans. Wrappers around the functions are also knows as decorators. Wrapper functions can be used as an interface to adapt to the existing codes, so as to save you from modifying your codes back and forth.

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

Ans. A generator is a function that returns an object (iterator) which we can iterate over (one value at a time).

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. Generator is a written as normal function but uses yield keyword to return values instead of return keyword.

Q12. Identify at least one benefit of generators.

Ans. Memory is saved as the items are produced when required, unlike normal Python functions. This fact becomes very important when you need to create a huge number of iterators. This is also considered as the biggest advantage of generators. Can be used to produce an infinite number of items.