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

Ans **+=** adds a number/string to a variable, changing the variable itself in the process. Meaning, ( += ) adds the value of the right operand to a variable and assigns the result to the variable.

Not really it result in faster performance because it take lesser time to operate and we dont require a new

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  In C programming it takes minimum 3 number of statement.

c=a+b

a=c

b=a

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

Ans By using list comprehension in reverse order

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

print(l)

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 l=[i for i in range(1,4)]

l2=[]

for i in range(33):

l2.extend(l)

print(l2,'\n')

print(f'Number of integers ares: {len(l2)}')

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

Ans It is very simple procedure. We first need a variable and assign it a multidimensional list.

a = [[2, 4, 6, 8, 10], [3, 6, 9, 12, 15], [4, 8, 12, 16, 20]]

print(a)

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

Ans. l=[i for i in 'Srinath']

print(l)

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

Ans help(print)

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 In python there is a beautiful things which we can do with the function. It is called first class function because of the following reasons:

1) It can be use as an argument

2) It can be use in return by some other function

3) It can be act as variable and we can assign some value to it

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

Ans **Wrappers**: It also refers to decorators. It allows programmers to modify the behavior of function or class.

**Wrapper Function**: Wrapper functions can be used as an interface to adapt to the existing codes, so as to save from modifying the codes back and forth. It helps in allowing some codes to run repeatedly.

**Decorators**: It allows 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?

Ans Generator function is a kind of a iterable function which knows that what i have generated the last time and what it will give me the next time. Using return or print statement we can simply get every element which we have iterated upon immediately at once. But what if we want that we get the element one by one as per our requirement. Here generator comes.

We can also say that Generator is something which help in creating iterator sort of thing. We use **yield** keyword in place of **return** when we are dealing with generators. With the help of generator only it is not necessary to create an iterator from an iterables. We can make any thing an iterator object. It do not stores all the items at once. Whatever the value we need, it catches and throw it out. Means whenever we need a next value to print we just call a next function over its object and it will give that value.

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 Use yield statement instead of print inside the function, to make it a generator function

Q12. Identify at least one benefit of generators.

Ans One of the advantages of the generator over the iterator is that elements are generated dynamically. Since the next item is generated only after the first is consumed, it is more memory efficient than the iterator.