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

Ans: - The assignment operator += is not just for show. It's a shorthand for a = a + b. It can lead to faster results at runtime because it avoids the need to look up the variable twice.

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 most programming languages, replacing the Python expression a, b = a + b, a would require two statements. For example, in C++:

```
int temp = a;
a = a + b;
b = temp;
```

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

Ans: - In Python, you can set a list of 100 integers to 0 using list comprehension:

```
my_list = [0 for _ in range(100)]
```

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: - To initialize a list of 99 integers that repeats the sequence 1, 2, 3 in Python, you can use list comprehension with the modulo operator:

```
my_list = [(i % 3) + 1 for i in range(99)]
```

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

Ans: - In IDLE, you can print a multidimensional list just like you would in any Python environment. For example, if my_list is a multidimensional list, you can print it using print(my_list)

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

Ans: - Yes, you can use list comprehension with a string in Python. For example, to create a list of all characters in a string, you can do:

```
my_string = "Hello, World!"
my_list = [char for char in my_string]
```

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

Ans: - From the command line, you can use Python's built-in help() function to get support with a user-written Python program. This is also 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++?

Ans: - In Python, functions are first-class objects, meaning they can be assigned to variables, stored in data structures, passed as arguments to other functions, and even returned as values from other functions34567. This is not typically possible in languages like C or C++.

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

Ans: - A wrapper is a function that is used to extend the behavior of another function without permanently modifying it8. A wrapped function is the function that has been extended by the wrapper. A decorator is a special type of wrapper that is applied to a function using the @decorator syntax in Python.

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

Ans: - A generator function in Python returns a special type of iterator called a generator10111213. You can iterate over this generator to retrieve the values it yields.

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 make a function a generator function in Python, you need to include at least one yield statement in its definition.

Q12. Identify at least one benefit of generators.

Ans: - One benefit of generators is that they allow you to create iterators with a very memory-efficient way. This is because they generate values on the fly rather than storing them in memory, which is especially useful when dealing with large data streams.