ASSIGNMENT - 14

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 can lead to faster results at runtime. For certain mutable types like lists, dictionaries, or arrays, using += to modify the object in place can be more efficient than creating a new object with the result of the addition.

Q2. What is the smallest number of statements you&#39;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 might require more statements due to the absence of tuple unpacking or multiple assignments in some languages. For instance, in languages without direct tuple assignment, you might need separate statements to achieve the same effect.

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 using list comprehension:

repeated\_sequence = [i % 3 + 1 for i in range(99)]

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 repeat the sequence 1, 2, 3, you can use list comprehension in Python:

repeated\_sequence = [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: To efficiently print a multidimensional list in IDLE or any Python environment, you can use nested loops or list comprehension to iterate through the elements and print them systematically.

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 characters from a string:

my\_string = "Hello"

char\_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: To get support with a user-written Python program from the command line, you might refer to Python's built-in help system by using python -m pydoc your\_module\_name. In IDLE, you can get help using the help() function or by using the pydoc module.

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 passed around as arguments, returned from other functions, assigned to variables, and dynamically created at runtime. This flexibility is not as common in languages like C or C++.

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

Ans: In Python, functions are first-class objects, meaning they can be passed around as arguments, returned from other functions, assigned to variables, and dynamically created at runtime. This flexibility is not as common in languages like C or C++.

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

Ans: If a function is a generator function in Python, it returns an iterator when called. The yield keyword in the function makes it a generator, allowing it to yield values one at a time instead of returning all values at once.

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: The improvement required for a function to become a generator function in Python is to use the yield keyword instead of return to produce a series of values over multiple calls.

Q12. Identify at least one benefit of generators.

Ans: One benefit of generators in Python is their memory efficiency. They produce values on the fly, allowing you to work with large datasets without needing to store the entire sequence in memory at once. This feature is especially useful for handling large or infinite sequences.