**Python Advanced Assignment 12**

Q1. Does assigning a value to a string’s indexed character violate Python’s string immutability?

Ans-) Yes, it violates the string immutability as strings are immutable objects in Python.

Q2. Does using the += operator to concatenate strings violate Python’s string immutability? Why or why not?

Ans-) No, it doesn't violate the string immutability as the += operator creates a new string object that is the concatenation of the two original strings.

Q3. In Python, how many different ways are there to index a character?

Ans-) There is only one way to index a character in Python, which is by using an integer index within square brackets.

Q4. What is the relationship between indexing and slicing?

Ans-) Indexing and slicing are both ways to access a substring from a larger string in Python. Indexing returns a single character at the specified index, while slicing returns a substring consisting of a range of characters specified by a start index and an end index.

Q5. What is an indexed character’s exact data type? What is the data form of a slicing-generated substring?

Ans-) An indexed character has a string data type. A slicing-generated substring is also a string object.

Q6. What is the relationship between string and character “types”; in Python?

Ans-) In Python, characters are represented as single-character strings, so there is no distinct character type. Strings are a sequence of characters.

Q7. Identify at least two operators and one method that allow you to combine one or more smaller

strings to create a larger string.

Ans-) The + operator and the += operator can be used to concatenate two or more strings. The join() method can be used to join a sequence of strings into a single string.

Q8. What is the benefit of first checking the target string with in or not in before using the index

method to find a substring?

Ans-) The benefit of first checking the target string with in or not in before using the index method to find a substring is that it avoids a ValueError if the substring is not found. The in or not in operator returns a Boolean value indicating whether the substring is present in the target string or not, without raising an exception.

Q9. Which operators and built-in string methods produce simple Boolean (true/false) results?

Ans-) The in and not in operators produce Boolean results indicating whether a substring is present in a target string or not. The startswith() and endswith() methods return True or False based on whether the target string starts or ends with a specified substring.