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

Answer> n Python, a string is immutable. You cannot overwrite the values of immutable objects. However, you can assign the variable again. It's not modifying the string object; it's creating a new string object.

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

Answer> It violates the rules of how ID values and += are supposed to work - the ID values produced with the optimization in place would be not only impossible, but prohibited, with the unoptimized semantics

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

Answer> We can access characters in a String in Two ways : Accessing Characters by Positive Index Number. Accessing Characters by Negative Index Number

Q4. What is the relationship between indexing and slicing?

Answer> “Indexing” means referring to an element of an iterable by its position within the iterable. “Slicing” means getting a subset of elements from an iterable based on their indices

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

Answer> ‘str’ and A Python substring is a portion of text taken from a string in string form.

Q6. What is the relationship between string and character "types" in Python?

Q7. Identify at least two operators and one method that allow you to combine one or more smaller strings to create a larger string.

Answer> Using + operator, % operator, join() method

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?

Answer> The index() method returns the position at the first occurrence of the specified value.

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

Answer> isalnum(), isascii(), isalpha(), isdecimal(), isdigit(),islower(), isnumeric(), isprintable(),isspace(), istitle()