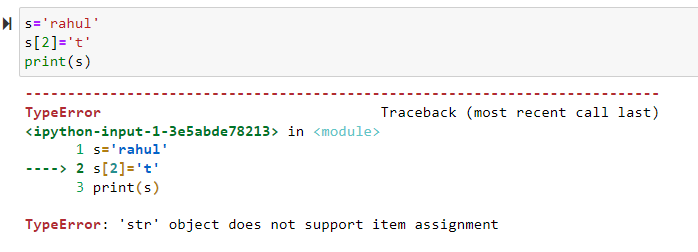
Q1. Does assigning a value to a string's indexed character violate Python's string immutability?

String is an immutable. So we can't do any updation or deletion with the string items through indexing.



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

No it doesnot violate it because string class is having an inbuilt method which is add () which takes strings and concatenate them

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

There are various ways in pytho to get some item at some index position.

1) By simply using the indexing and slicing method inside square bracket. Eg: item[1] or item[1:4] or item[3::2], etc

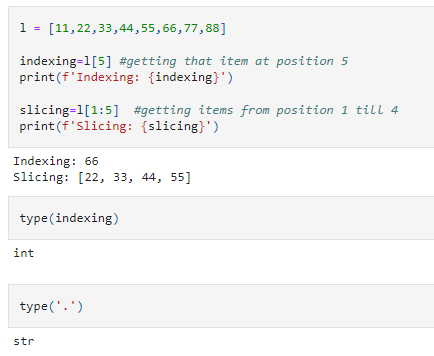
2) By using list index method. **list.index(position)**. In this way one can get the item at some index position.

3) Using built in method like **getitem ()** and **index**. We can use this inside the class

Q4. What is the relationship between indexing and slicing?

**Indexing**: It is a technique used to get the item of that particuar index position or in cas eof mutable object we can also update the value or add the new value by just giving the index position and assignment operator.

**Slicing**: It is use when we wan to extract some range of a item from some object.



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

The exact data type for the indexed character is **CHAR** but generally it is refer to as a string. Or it could that which was earlier before the indexing.

The data type of a substring produced from slicing is also a string

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

Character refers to a single letter, number, space, punctuation mark or a symbol. Its data type is actually a **CHAR** which stores character data in a fixed-length field.

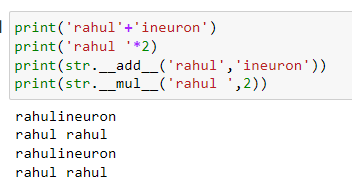
While String is a sequence of characters. A group of different different character compose a string. We enclosed it by ' ' Strings are arrays of bytes representing Unicode characters. Its type is **string**

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

In this case concatenation and a kind of multiple times of string comes into action. Concatenation needed **+** operator while multiple time needed '\*' operator.

In case of methods there are a method which do the same:

**add**and **mul**



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?

When we first check the target string using **in** or **not in** containment operator then we could get to know whether that particular target string is present or not. If it is not present the it will give True as a result else will False. Its better to check before becaue if we are not testinng it first and the substring is not present in the string then it will result in an exception

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

**contains**is a built in method which produces the Boolean values in result.

Others are also some methods which result the boolean values: 'isalnum','isalpha', 'isascii','isdecimal','isdigit','isidentifier', 'islower', 'isnumeric','isprintable', 'isspace', 'istitle', 'isupper',

The operator which used in this is **'in'** which is also called as containment operator