Solution Review: String Transformation

This lesson will give a detailed review of how to update a string.

```
WE'LL COVER THE FOLLOWINGSolution: Use len() and concatenation(+) Operation
```

Solution: Use len() and concatenation(+) Operation

- Use len(str) to calculate the length of string str
- Concatenate value at a certain position in the string using the concatenation operation

Given a string 'str', use the following piece of code to transform the string

```
str = str[:position] + character_to_insert + str[position:]
```

The character needs to be inserted where the position is in the code.

Have a look at the following illustration to get an insight on how to update the length of a string using concatenation operation.

```
abc
1 of 7
```

```
s=s[:1]+s[0]+s[1:]
aabc
```

```
s=s[:1]+s[0]+s[1:]
aaabc

3 of 7
```

```
s=s[:3]+s[3]+s[3:] aaabb
```

```
s=s[:6]+s[6]+s[6:]
aaabbbcc

6 of 7
```

```
s=s[:6]+s[6]+s[6:]
aaabbbccc
7 of 7
```



The following python code shows how to transform the string.

```
def getStr(s):
    s=s[:1] + s[0] + s[1:]# Transform the string
    s=s[:1] + s[0] + s[1:]
    s=s[:3] + s[3] + s[3:]
```

```
s=s[:6] + s[6] + s[6:]
s=s[:6] + s[6] + s[6:]
# Update the length of string
strlen = len(s)
return [s, strlen]

print(getStr("abc"))
print(getStr("xyz"))
```

Let's solve another problem using strings.