Python - Modificar Strings

Em Python , uma string (objeto da classe **str**) é de tipo imutável. Um objeto imutável é aquele que pode ser modificado no local, criado na memória. Portanto, ao contrário de list , qualquer caractere na sequência não pode ser substituído, nem podemos inserir ou acrescentar caracteres a ele, a menos que usemos determinado método de string que retorne um novo objeto de string.

No entanto, podemos usar um dos truques a seguir como solução alternativa para modificar uma string.

Convertendo uma String em uma Lista

Como os objetos string e list são sequências, eles são interconversíveis. Portanto, se convertermos um objeto string em uma lista, modifique a lista pelos métodos insert() , append() ou remove() e converta a lista de volta em uma string, para recuperar a versão modificada.

Exemplo

Temos uma variável string s1 com WORD como valor. Com a função integrada list() , vamos convertê-lo em um objeto de lista l1 e inserir um caractere L no índice 3. Usamos o método join() na classe str para concatenar todos os caracteres.

```
s1="WORD"
print ("original string:", s1)
l1=list(s1)

l1.insert(3,"L")

print (l1)

s1=''.join(l1)
print ("Modified string:", s1)
```

Ele produzirá a seguinte **saída** -

```
original string: WORD
['W', 'O', 'R', 'L', 'D']
Modified string: WORLD
```

Usando o Módulo Array

Para modificar uma string, construa um objeto array . A biblioteca padrão Python inclui módulo array. Podemos ter um array do tipo Unicode a partir de uma variável string .

Exemplo

```
import array as ar
s1="WORD"
sar=ar.array('u', s1)
```

Os itens da matriz possuem um índice baseado em zero. Assim, podemos realizar operações de array como acrescentar, inserir, remover etc. Vamos inserir L antes do caractere D

```
sar.insert(3,"L")
```

Agora, com a ajuda do método tounicode(), recupere a string modificada

Exemplo

```
import array as ar

s1="WORD"
print ("original string:", s1)

sar=ar.array('u', s1)
sar.insert(3,"L")
s1=sar.tounicode()

print ("Modified string:", s1)
```

Ele produzirá a seguinte **saída** -

```
original string: WORD
Modified string: WORLD
```

Usando a classe StringIO

O módulo io do Python define as classes para lidar com fluxos. A classe StringIO representa um fluxo de texto usando um buffer de texto na memória. Um objeto StringIO obtido de uma string se comporta como um objeto File. Portanto, podemos realizar operações de leitura/gravação nele. O método getvalue() da classe StringIO retorna uma string.

Vamos usar este princípio no programa a seguir para modificar uma string.

Exemplo

```
import io

s1="WORD"
print ("original string:", s1)

sio=io.StringIO(s1)
sio.seek(3)
sio.write("LD")
s1=sio.getvalue()

print ("Modified string:", s1)
```

Ele produzirá a seguinte **saída** -

```
original string: WORD
Modified string: WORLD
```