Strings em Python

Índice

Explicações Adicionais

- **Slicing Strings:** Extrai partes específicas de uma string usando índices e passos.
- **Modify Strings** Strings são imutáveis; você cria novas strings com as modificações desejadas.
- **Concatenate Strings** Junta strings usando o operador `+` ou o método `join()`.
- **Format Strings** Incorpora variáveis e valores dentro de uma string usando `format()` ou f-strings.
- **Escape Characters** Inclui caracteres especiais em uma string usando a barra invertida (`\`).
- **String Methods** Utiliza métodos incorporados para manipular e processar strings.

Slicing Strings

O slicing (fatiamento) permite extrair partes de uma string usando a sintaxe `string[início:fim:passo]`.

```
# Código
text = 'Python Programming'

# Fatiamento básico
sub_text1 = text[7:18]     # 'Programming'
sub_text2 = text[:6]       # 'Python'
sub_text3 = text[7:]       # 'Programming'
sub_text4 = text[-11:-1]   # 'Programming'

# Fatiamento com passo
sub_text5 = text[::2]      # 'Pto rgamn'

print(sub_text1)
print(sub_text2)
print(sub_text3)
print(sub_text4)
print(sub_text5)

# Saída
# Programming
# Python
# Programming
# Programming
# Pto rgamn
```

Modify Strings

Strings em Python são imutáveis, então você não pode alterar uma string existente diretamente. Em vez disso, você pode criar uma nova string com as modificações desejadas.

```
# Código
original_string = 'Hello World'
modified_string = original_string.replace('World', 'Python')

print(modified_string)

# Saída
# Hello Python
```

Concatenate Strings

Você pode concatenar strings usando o operador `+` ou o método `join()` para unir uma lista de strings.

```
# Código
string1 = 'Hello'
string2 = 'World'
concatenated_string = string1 + ' ' + string2

# Usando join()
words = ['Hello', 'World']
joined_string = ' '.join(words)

print(concatenated_string)
print(joined_string)

# Saída
# Hello World
# Hello World
```

Format Strings

A formatação de strings permite incorporar variáveis e valores dentro de uma string de maneira flexível.

Em Python, você pode usar o método `format()` ou f-strings para formatação.

```
# Código
name = 'Alice'
age = 30
formatted_string = 'Name: {}, Age: {}'.format(name, age)

print(formatted_string)

# Saída
# Name: Alice, Age: 30
```

```
# Código
formatted_string = f'Name: {name}, Age: {age}'

print(formatted_string)

# Saída
# Name: Alice, Age: 30
```

Escape Characters

Escape characters são usados para incluir caracteres especiais em uma string. O caractere de escape em Python é a barra invertida (`\`).

```
# Código
single_quote = 'I\'m learning Python.'
double_quote = "He said, \"Hello!\""
newline = 'First line\nSecond line'
tabbed = 'Column1\tColumn2'

print(single_quote)
print(double_quote)
print(newline)
print(tabbed)

# Saída
# I'm learning Python.
# He said, "Hello!"
# First line
# Second line
# Column1    Column2
```

String Methods

Python oferece uma ampla gama de métodos para trabalhar com strings. Aqui estão alguns dos métodos mais comuns:

```
# Código
text = '   Python Programming   '

# strip() - Remove espaços em branco no início e no final
stripped_text = text.strip()

# upper() - Converte para maiúsculas
upper_text = text.upper()

# lower() - Converte para minúsculas
lower_text = text.lower()

# find() - Encontra a posição da primeira ocorrência de uma substring
position = text.find('Programming')

# replace() - Substitui parte da string
replaced_text = text.replace('Programming', 'Coding')

# split() - Divide a string em uma lista
words = text.split()

print(stripped_text)
print(upper_text)
print(lower_text)
print(position)
print(replaced_text)
print(words)

# Saída
# Python Programming
# PYTHON PROGRAMMING
# python programming
# 7
#    Python Coding
# ['Python', 'Programming']
```