Funções e Métodos para Manipulação de Strings em Python

1. **`len(string)`** - Retorna o comprimento da string, ou seja, o número de caracteres.
2. **`string.upper()`** - Retorna a string toda em maiúsculas.
3. **`string.lower()`** - Retorna a string toda em minúsculas.
4. **`string.capitalize()`** - Converte o primeiro caractere da string para maiúsculo.
5. **`string.title()`** - Converte a primeira letra de cada palavra em maiúsculo.
6. **`string.strip()`** - Remove os espaços em branco no início e no final da string.
7. **`string.replace(old, new)`** - Substitui todas as ocorrências de uma substring (old) por uma nova substring (new).
8. **`string.split(separator)`** - Divide a string em uma lista, usando o `separator` como critério de separação.
9. **`string.join(iterable)`** - Junta os elementos de um `iterable` em uma única string, com um separador entre eles.
10. **`string.find(substring)`** - Retorna o índice da primeira ocorrência da `substring`. Retorna -1 se não for encontrada.
11. **`string.count(substring)`** - Conta quantas vezes uma `substring` aparece na string.
12. **`string.startswith(prefix)`** - Retorna `True` se a string começa com o prefixo especificado.
13. **`string.endswith(suffix)`** - Retorna `True` se a string termina com o sufixo especificado.
14. **`string.isalpha()`** - Retorna `True` se todos os caracteres da string são letras.
15. **`string.isdigit()`** - Retorna `True` se todos os caracteres da string são dígitos.
16. **`string.isalnum()`** - Retorna `True` se todos os caracteres da string são letras ou números.
17. **`string.isspace()`** - Retorna `True` se a string contém apenas espaços.
18. **`string.swapcase()`** - Converte caracteres maiúsculos em minúsculos e vice-versa.
19. **`string.zfill(width)`** - Adiciona zeros à esquerda da string para que ela tenha um comprimento específico.

Exemplo:

```
texto = "Olá, mundo!"
print(texto.upper())
```

Resultado:

```
OLÁ, MUNDO!
```

Fatiamento de string

```
string[início:fim:passo]
```

exemplos:

```
texto = "programação"

# Pegando os primeiros três caracteres
print(texto[:3])  # Resultado: "pro"

# Pegando os últimos três caracteres
print(texto[-3:])  # Resultado: "ção"

# Pegando do índice 2 ao 5 (excluindo o índice 5)
print(texto[2:5])  # Resultado: "ogr"
```

Reflita

1. Como você acha que os métodos de manipulação de strings podem facilitar o trabalho em projetos reais? Pode imaginar exemplos práticos?
2. Como o uso de funções como `upper`, `lower` e `capitalize` pode ser útil para padronizar textos em uma aplicação? Onde isso pode ser necessário?
3. Por que remover espaços extras com métodos como `strip` pode ser importante em uma aplicação que lida com entradas de usuários? Que problemas esses espaços podem causar?
4. Como a função `find` pode ajudar a criar ferramentas de busca? Pode imaginar algum caso prático onde seria útil buscar uma palavra dentro de uma frase?
5. Quais vantagens você vê em verificar se uma string é composta apenas por letras ou números (`isalpha`, `isdigit`)? Onde isso pode ser relevante em aplicativos ou sistemas?
6. Você consegue pensar em um cenário em que seria útil inverter a caixa dos caracteres em uma string (`swapcase`)?
7. De que forma o `len` e a contagem de caracteres pode ajudar a garantir que mensagens, nomes de usuários ou senhas sigam um padrão específico?
8. Como os métodos de manipulação de strings podem ajudar a melhorar a experiência de um usuário ao preencher formulários em uma aplicação?
9. Como o fatiamento de strings pode facilitar a extração e manipulação de partes específicas de dados em um projeto real? Você consegue pensar em algum exemplo prático em que precisaria extrair uma parte específica de uma string para usá-la em outra função ou exibição?

Exercícios

1. Escreva um programa que peça ao usuário uma palavra e exiba quantos caracteres ela possui.
2. Solicite ao usuário para digitar uma frase. Exiba essa frase em letras maiúsculas e em letras minúsculas.
3. Peça ao usuário uma frase e exiba a frase com a primeira letra em maiúsculo.
4. Solicite o nome completo do usuário e exiba-o no formato de título (primeira letra de cada nome em maiúscula).
5. Peça ao usuário para digitar uma frase com espaços antes e depois. Remova os espaços e exiba a frase limpa.
6. Solicite uma frase e uma palavra que o usuário quer substituir. Depois, peça a nova palavra e exiba a frase atualizada.
7. Peça ao usuário uma frase e uma palavra. Exiba o índice da primeira ocorrência da palavra na frase.
8. Solicite ao usuário uma frase e uma palavra. Exiba quantas vezes a palavra aparece na frase.
9. Peça ao usuário para inserir uma palavra e um prefixo. Verifique se a palavra começa com o prefixo e exiba o resultado.
10. Solicite ao usuário uma palavra e um sufixo. Verifique se a palavra termina com o sufixo e exiba o resultado.
11. Peça ao usuário uma palavra e verifique se todos os caracteres são letras. Exiba `True` ou `False`.
12. Solicite uma entrada ao usuário e verifique se ela contém apenas dígitos. Exiba `True` ou `False`.
13. Solicite uma frase ao usuário e exiba quantos espaços ela contém.
14. Peça ao usuário uma frase e exiba a frase com letras maiúsculas convertidas em minúsculas e vice-versa.
15. Peça ao usuário um número e exiba-o com 5 dígitos, preenchendo com zeros à esquerda, se necessário.
16. Peça ao usuário que insira o nome completo e exiba apenas o primeiro nome e o último sobrenome.
17. Solicite uma palavra ao usuário e exiba quantas vogais (a, e, i, o, u) ela contém.