

# Programação I

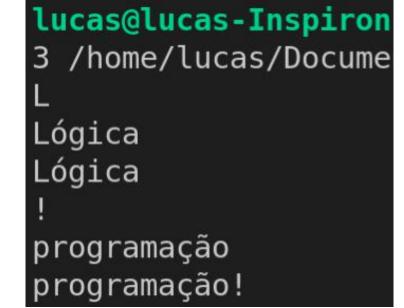
LUCAS SAMPAIO LEITE

- ☐ A inicialização de uma string pode ser tanto com aspas simples, como com aspas duplas;
- ☐ Como strings são sequências fixas de caracteres, cada caractere está alocado na memória ocupando uma posição.
- Dessa forma, é possível acessar cada caractere de uma string acessando sua respectiva posição através da utilização de colchetes [].

Exemplos:

```
main.py > ...

1     s = 'Lógica de programação!'
2     print(s[0])
3     print(s[0:6])
4     print(s[:6])
5     print(s[-1])
6     print(s[-12:-1])
7     print(s[-12:])
```



Como uma string é uma sequência de caracteres, então ela pode ser percorrida por laços de repetição (for ou while).

```
main.py > ...

1     s = 'Lógica de programação!'
2     i=0
3     while i != len(s):
4         print(s[i])
5     i += 1
```

```
main.py > ...

1     s = 'Lógica de programação!'
2     i=0
3     while i != len(s):
4         i += 1
5         print(s[-i])
```

Como uma string é uma sequência de caracteres, então ela pode ser percorrida por laços de repetição (for ou while).

```
main.py > ...

1    s = 'Lógica de programação!'
2    for i in range(len(s)):
3        print(s[i])
```

```
main.py > ...

1    s = 'Lógica de programação!'
2    for i in range(len(s)):
3        print(s[-i-1])
```

☐ Funções muito utilizadas para manipulação de strings:
 ☐ upper(): eleva todos os caracteres da string para maiúsculos;
 ☐ lower() ou casefold(): todos os caracteres da string para minúsculos;
 ☐ find(): busca um determinado caractere dentro da string e retorna sua posição;
 ☐ count(): conta o número de repetições de um caractere dentro da string;
 ☐ split(): recorta a string, transformando-a em uma lista;
 ☐ len(): retorna o tamanho da string.

```
main.py > ...
    s = 'Lógica de programação!'
    i=0
    print(s.upper())
3
    print(s.lower())
4
    print(s.find('!'))
5
    print(s.find('a'))
6
    print(s.find('de'))
    print(s.count('a'))
8
    print(s.split(' '))
9
```

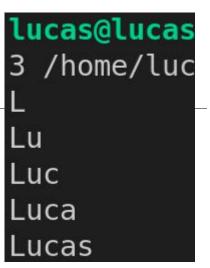
```
lucas@lucas-Inspiron-5548:~/Docum
3 /home/lucas/Documents/vscode-pr
LÓGICA DE PROGRAMAÇÃO!
lógica de programação!
21
5
7
3
['Lógica', 'de', 'programação!']
```

- Outras funções muito utilizadas para manipulação de strings:
  - □ capitalize() -> Coloca a 1º letra Maiúscula;
  - format() -> Formata uma string de acordo com os valores passados;
  - □ isalnum() -> Verifica se um texto é todo feito com caracteres alfanuméricos (letras e números) -> letras com acento ou ç são considerados letras para essa função;
  - □ isalpha() -> Verifica se um texto é todo feito de letras;
  - ☐ isnumeric() -> Verifica se um texto é todo feito por números;
  - replace() -> Substitui um texto por um outro texto em uma string;

- Outras funções muito utilizadas para manipulação de strings:
  - splitlines() -> separa um texto em vários textos de acordo com os "enters" do texto;
  - startswith() -> Verifica se a string começa com determinado texto;
  - □ strip() -> é usado para remover espaços em branco (ou outros caracteres especificados) do início e do final de uma string.
  - □ title() -> Coloca a 1ª letra de cada palavra em maiúscula;

- 1. Faça um programa que leia 2 strings e informe o conteúdo delas seguido do seu comprimento. Informe também se as duas strings possuem o mesmo comprimento e são iguais ou diferentes no conteúdo.
- 2. Faça um programa que permita ao usuário digitar o seu nome e em seguida mostre o nome do usuário de trás para frente utilizando somente letras maiúsculas. Dica: lembre-se que ao informar o nome, o usuário pode digitar letras maiúsculas ou minúsculas.

3. Faça um programa que solicite uma string ao usuário e em seguida a imprima em formato de escada.



4. Altere o programa anterior de modo que a escada seja invertida.

lucas@luca 3 /home/lucas Lucas ucas cas as

- 5. Um palíndromo é uma seqüência de caracteres cuja leitura é idêntica se feita da direita para esquerda ou vice-versa. Por exemplo: OSSO e OVO são palíndromos. Em textos mais complexos os espaços e pontuação são ignorados. A frase SUBI NO ONIBUS é o exemplo de uma frase palíndroma onde os espaços foram ignorados. Faça um programa que leia uma seqüência de caracteres, mostre-a e diga se é um palíndromo ou não.
- 6. Faça uma função que recebe uma string que representa uma cadeia de DNA e gera a cadeia complementar. A entrada e saída de dados deve ser feita pelo programa principal.
  - Exemplo:
  - Entrada: AATCTGCAC
  - Saída: TTAGACGTG

- 7. Faça um programa que leia uma data de nascimento no formato dd/mm/aaaa e imprima a data com o mês escrito por extenso.
  - **Exemplo:**
  - Data = 20/02/1995
  - Resultado gerado pelo programa:
  - □ Você nasceu em 20 de fevereiro de 1995

- 8. Escreva um programa que leia duas strings. Verifique se a segunda ocorre dentro da primeira e imprima a posição de início.
  - 1º string: AABBEFAATT
  - 2ª string: BE
  - Resultado: BE encontrado na posição 3 de AABBEFAATT
- 9. Escreva um programa que leia duas strings e gere uma terceira com os caracteres comuns às duas strings lidas.
  - 1º string: AAACTBF
  - 2ª string: CBT
  - Resultado: CBT
  - A ordem dos caracteres da string gerada não é importante, mas deve conter todas as letras comuns a ambas.

- 10. Conta espaços e vogais. Dado uma string com uma frase informada pelo usuário (incluindo espaços em branco), conte: quantos espaços em branco existem na frase. quantas vezes aparecem as vogais a, e, i, o, u.
- 11. Escreva um programa que leia uma string e imprima quantas vezes cada caractere aparece nessa string.
  - String: TTAAC
  - Formato de saída:
  - T: 2x
  - A: 2x
  - C: 1x

- 12. Número por extenso. Escreva um programa que solicite ao usuário a digitação de um número até 99 e imprima-o na tela por extenso.
- 13. Faça um programa que leia uma palavra e some 1 no valor ASCII de cada caractere da palavra. Imprima a string resultante.
  - Dica: O Python disponibiliza 2 funções que são bastante uteis quando estamos trabalhando com o sistema ASCII. A primeira é a função ord(), que recebe uma letra como parâmetro e retorna o código ASCII da mesma. A segunda função, é a chr(), onde passamos o código ASCII e nos é retornado a respectiva letra.
  - □ Tabela ASCII: https://www.ime.usp.br/~pf/algoritmos/apend/ascii.html

- 14. Faça um programa que solicite ao usuário uma string e modifique a string para que todos os caracteres fiquem em maiúsculas.
  - Obs: Não utilize a função upper(). Utilize a tabela ASCII.
- 15. Faça um programa em que troque todas as ocorrencias de uma letra L1 pela letra L2 e da L2 pela L1 em uma string. A string e as letras L1 e L2 devem ser fornecidas pelo usuario.
  - Obs: Não utilize a função replace().

### Dúvidas???



Fonte: https://institutoseculoxxi.com.br/duvidas-entramos-em-contato-com-voce/