

PROGRAMAÇÃO WEB II

Curso Técnico Integrado em Informática
Lucas Sampaio Leite



Revisão de conceitos básicos em Python

- Fatiamento de Strings:

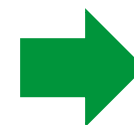
```
main.py > ...  
1  s = 'Lógica de programação!'  
2  print(s[0])  
3  print(s[0:6])  
4  print(s[:6])  
5  print(s[-1])  
6  print(s[-12:-1])  
7  print(s[-12:])
```

Fatiar (ou "slicing") uma string em Python é o processo de extrair uma subsequência de caracteres da string original. Isso é feito especificando um intervalo de índices.

Revisão de conceitos básicos em Python

- Fatiamento de Strings:

```
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:])
```



```
lucas@lucas-Inspiron  
3 /home/lucas/Docume  
L  
Lógica  
Lógica  
!  
programação  
programação!
```

Fatiar (ou "slicing") uma string em Python é o processo de extrair uma subsequência de caracteres da string original. Isso é feito especificando um intervalo de índices.

Revisão de conceitos básicos em Python

- O que seria impresso?

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

Revisão de conceitos básicos em Python

- O que seria impresso?

```
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  for i in range(len(s)):  
3      |    print(s[-i-1])
```

Revisão de conceitos básicos em Python

- O que está sendo verificado?

```
s = "Lógica de programação"
c = "L"

if c in s:
    print(f"O caractere {c} está presente na string {s}")
```

Revisão de conceitos básicos em Python

- O que está sendo verificado?

```
s = "Lógica de programação"
c = "Z"

if c not in s:
    print(f"O caractere {c} não está presente na string {s}")
```

Revisão de conceitos básicos em Python

- 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.

Revisão de conceitos básicos em Python

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

Revisão de conceitos básicos em Python

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



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

Revisão de conceitos básicos em Python

- Outras funções muito utilizadas para manipulação de strings:
 - `capitalize()` -> Coloca a 1ª 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.

Revisão de conceitos básicos em Python

- 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;

Revisão de conceitos básicos em Python

- Para converter um caractere para seu valor ASCII, usamos a função `ord()`.

```
caractere = 'A'  
valor_ascii = ord(caractere)  
print(valor_ascii) # Saída: 65
```

- Para converter um valor ASCII de volta para seu caractere correspondente, usamos a função `chr()`.

```
valor_ascii = 65  
caractere = chr(valor_ascii)  
print(caractere) # Saída: 'A'
```

Revisão de conceitos básicos em Python

- Em Python, uma lista é uma sequência mutável de n valores que podem ser de qualquer tipo (inclusive outras listas):

```
main.py > ...  
1  lista = ['Pode conter qualquer tipo de valor', 0, 5 + 8j,  
2      ['Outras listas', 'por exemplo'], 5.5]  
3  print(lista)  
4
```

```
main.py > ...  
1  lista = [1, 3, 5, 7, 9]  
2  print(lista[0])
```


Revisão de conceitos básicos em Python

- O que será impresso?

```
main.py > ...  
1  lista = [1, 3, 5, 7, 9]  
2  print(lista[0])  
3  print(lista[3])  
4  print(lista[4])  
5  print(lista[len(lista)-1])  
6  print(lista[len(lista)-3])
```

Revisão de conceitos básicos em Python

- O que será impresso?

```
main.py > ...  
1  lista = [1, 3, 5, 7, 9]  
2  print(lista[5])  
3
```

```
main.py > ...  
1  lista = [1, 3, 5, 7, 9]  
2  print(lista[len(lista)])  
3
```


Revisão de conceitos básicos em Python

- O que será impresso?
- ERRO!!! IndexError: list index out of range

```
main.py > ...  
1 lista = [1, 3, 5, 7, 9]  
2 print(lista[5])  
3
```



```
main.py > ...  
1 lista = [1, 3, 5, 7, 9]  
2 print(lista[len(lista)])  
3
```



Os índices válidos vão de 0 a 4!!!

Revisão de conceitos básicos em Python

main.py > ...

```
1 lista = [1, 3, 5, 7, 9]
2 print (lista[2:4])
```

Intervalo contendo os elementos
das posições 2 e 3

main.py > ...

```
1 lista = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
2 print ('Lista: ', lista)
3 sublist = lista[1:10:2]
4 print('Sublista: ', sublist)
```

Lista: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Sublista: [1, 3, 5, 7, 9]

Revisão de conceitos básicos em Python

```
main.py > ...  
1  lista = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]  
2  print ('Lista: ', lista)  
3  print(lista[::])  
4  print(lista[5::])  
5  print(lista[:5:])  
6  print(lista[::2])
```

Revisão de conceitos básicos em Python

```
main.py > ...  
1  lista = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]  
2  print ('Lista: ', lista)  
3  print(lista[::])  
4  print(lista[5::])  
5  print(lista[:5:])  
6  print(lista[::2])
```



```
Lista:  [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]  
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]  
[5, 6, 7, 8, 9, 10]  
[0, 1, 2, 3, 4]  
[0, 2, 4, 6, 8, 10]
```

Revisão de conceitos básicos em Python

```
main.py > ...  
1  lista = [0, 1, 2, 3, 4, 5, 6, 7, 8]  
2  print ('Lista da antes da inserção: ', lista)  
3  lista.append(9)  
4  print ('Lista após a primeira inserção: ', lista)  
5  lista.append(10)  
6  print ('Lista após a segunda inserção: ', lista)
```



```
Lista da antes da inserção: [0, 1, 2, 3, 4, 5, 6, 7, 8]  
Lista após a primeira inserção: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]  
Lista após a segunda inserção: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

Revisão de conceitos básicos em Python

- O método `insert(índice, valor)` insere um elemento em uma posição específica da lista.

```
lista = ["a", "b", "c", 0, 1, 2, 3]
lista.insert(1, "IF")
lista.insert(-1, "Baiano")
lista.insert(len(lista), "Senhor do Bonfim")
print(lista)
```


Revisão de conceitos básicos em Python

- O método insert(índice, valor) insere um elemento em uma posição específica da lista.

```
lista = ["a", "b", "c", 0, 1, 2, 3]
lista.insert(1, "IF")
lista.insert(-1, "Baiano")
lista.insert(len(lista), "Senhor do Bonfim")
print(lista)
```



```
['a', 'IF', 'b', 'c', 0, 1, 2, 'Baiano', 3, 'Senhor do Bonfim']
```

Revisão de conceitos básicos em Python

- O método `pop()` realiza uma remoção com base em seu índice. Esta função retorna o elemento removido.
- O método `remove()` remove um elemento com base em seu valor.

```
lista = ["a", "b", "c", 0, 1, 2, 3]  
print(lista.pop(0))  
lista.remove(0)  
print(lista)
```



```
a  
['b', 'c', 1, 2, 3]
```


Revisão de conceitos básicos em Python

```
main.py > ...  
1  lista = ['Esta lista é heterogênea', 0, [0.5, 0.75, 1, 1.25, 1.75],  
2      2, 2.8, 3, 4, 5, 6, 7, 8, 8.55, 9, 10, True]  
3  print ('Lista heterogênea aninhada: ', lista)  
4  print ('O tamanho da lista é: ', len(lista))  
5  print ('O elemento da posição 2 é: ', lista[2])  
6  print ('O primeiro elemento da posição 2 é: ', lista[2][0])  
7  print ('O último elemento da posição 2 é: ', lista[2][len(lista[2])])  
8  print ('O tamanho da sublista é: ', len(lista[2]))
```

Revisão de conceitos básicos em Python

```
lista = ['Esta lista é heterogênea', 0, [0.5, 0.75, 1, 1.25, 1.75],  
        2, 2.8, 3, 4, 5, 6, 7, 8, 8.55, 9, 10, True]  
print('Lista heterogênea aninhada: ', lista)  
print('O tamanho da lista é: ', len(lista))  
print('O elemento da posição 2 é: ', lista[2])  
print('O primeiro elemento da posição 2 é: ', lista[2][0])  
print('O último elemento da posição 2 é: ', lista[2][len(lista[2])-1])  
print('O tamanho da sublista é: ', len(lista[2]))
```



```
Lista heterogênea aninhada: ['Esta lista é heterogênea', 0, [0.5, 0.75, 1, 1.25, 1.7  
5], 2, 2.8, 3, 4, 5, 6, 7, 8, 8.55, 9, 10, True]  
O tamanho da lista é: 15  
O elemento da posição 2 é: [0.5, 0.75, 1, 1.25, 1.75]  
O primeiro elemento da posição 2 é: 0.5  
O último elemento da posição 2 é: 1.75  
O tamanho da sublista é: 5
```

Revisão de conceitos básicos em Python

- Algumas funções para operar sobre listas com valores numéricos:
 - `sum(lista)`: retorna a soma de todos os elementos da lista;
 - `min(lista)`: retorna o valor mínimo armazenado;
 - `max(lista)`: retorna o valor máximo armazenado;
 - `lista.count(valor)`: retorna o número de ocorrências de valor em lista:

```
lista = [10, 1.5, 2, 13, 4, 85, 10]  
print(sum(lista))  
print(max(lista))  
print(min(lista))  
print(lista.count(10))
```



```
125.5  
85  
1.5  
2
```


Revisão de conceitos básicos em Python

```
lista = [10, 1.5, 2, 13, 4, 85, 10]
lista.sort()
print(lista)

lista = ["Ziraldo", "Lucas", "Marcos", "Ana"]
lista.sort(reverse=True)
print(lista)
```



```
[1.5, 2, 4, 10, 10, 13, 85]
['Ziraldo', 'Marcos', 'Lucas', 'Ana']
```

Revisão de conceitos básicos em Python

```
1  tupla = (0, 1, 2, 'muito parecido com listas')
2  print(tupla)
3
4  # Parenteses são opcionais .
5  tupla = 0, 1, 2, 'muito parecido com listas'
6  print(tupla)
```



```
(0, 1, 2, 'muito parecido com listas ')  
(0, 1, 2, 'muito parecido com listas ')
```

De forma geral as tuplas são mais eficientes que as listas pois consomem menos recursos computacionais (memória), por serem estruturas mais simples (Borges, 2014).

Revisão de conceitos básicos em Python

- Tuplas são imutáveis!!!!
- Manipulação por índice, função len():

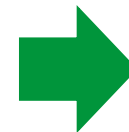
```
32 x = (1,2,3)
33 print(x)
34 print(x[0])
35 print(x[0:2])
36 print(len(x))
37 x[0]=2
```



```
(1, 2, 3)
1
(1, 2)
3
Traceback (most recent call last):
  File "/home/lucas/Documents/vscode-projects/lucas.py", line 37,
in <module>
    x[0]=2
TypeError: 'tuple' object does not support item assignment
```

Revisão de conceitos básicos em Python

```
8   tupla = 1,  
9   print(tupla)  
10  print(type(tupla))  
11  
12  tupla = 1  
13  print(tupla)  
14  print(type(tupla))
```

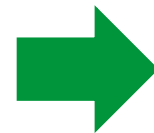


```
(1,)  
<class 'tuple'>  
1  
<class 'int'>
```

Revisão de conceitos básicos em Python

- Conversões entre Listas, Tuplas e Strings:

```
16  lista = [0,1,2,3,4]
17  print(lista)
18  tupla = tuple(lista)
19  print(tupla)
20  lista = list(tupla)
21  print(lista)
```



```
[0, 1, 2, 3, 4]
(0, 1, 2, 3, 4)
[0, 1, 2, 3, 4]
```


Revisão de conceitos básicos em Python

- Conversões entre Listas, Tuplas e Strings:

```
1  tupla = (0, 1, 2, 'muito parecido com listas')
2  print(tupla)
3  string = str(tupla)
4  print(string)
5  print(type(string))
```



```
(0, 1, 2, 'muito parecido com listas')
(0, 1, 2, 'muito parecido com listas')
<class 'str'>
```

Exercícios de revisão

1. Dados três valores distintos, fazer um programa que, após a leitura destes dados, coloque-os em ordem crescente (não usar listas ou ordenação).
2. Escreva um programa que imprime todos os numeros de 0 até 50, incluindo-os.
3. Escreva um programa para contar a quantidade de números pares entre dois números quaisquer fornecidos pelo usuário? Obs: Considerar intervalo crescente e decrescente.
4. Escreva um programa para calcular o fatorial de um número fornecido pelo usuário.

Exercícios de revisão

5. 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

6. Faça um programa em que troque todas as ocorrências 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 usuário.

- Obs: Não utilize a função `replace()`.

Exercícios de revisão

7. Faça um programa que armazene as idades e as alturas de 4 alunos. Seu programa deve exibir quantos alunos com mais de 13 anos possuem uma altura inferior à altura média dentre todos os alunos.
8. Modifique o programa da questão 3 para que o programa funcione para qualquer quantidade de alunos. Assim, durante a leitura das idades e alturas o usuário poderá inserir um valor negativo para indicar que deseja interromper a leitura dos dados.

Exercícios de revisão

9. Escreva um programa que compare duas tuplas e imprima:
- a) uma lista com os valores comuns às duas tuplas
 - b) uma lista com os valores que só existem na primeira tupla
 - c) uma lista com os valores que existem apenas na segunda tupla
 - d) uma lista com com os elementos não repetidos das duas tuplas.

Site da disciplina



Prof. Lucas Sampaio Leite

Bem-vindo ao meu portfólio de disciplinas ministradas! Aqui você encontra os materiais atualizados das disciplinas que ministro atualmente no Instituto Federal Baiano.

Disciplinas

→ **Curso Técnico Integrado em Informática**

Lógica e Linguagem de Programação

Programação Web II

Curso Técnico Subsequente em Informática

Lógica e Linguagem de Programação

Projeto Integrador I

Curso Técnico Integrado em Agroindústria

Informática Básica

Para mais informações sobre pesquisa e áreas de interesse, acesse meu [Currículo Lattes](#).



Link: <https://lucassampaioleite.github.io/portfolio/>

LÓGICA E LINGUAGEM DE PROGRAMAÇÃO

Curso Técnico Integrado em Informática
Lucas Sampaio Leite

