



Estácio

ARA0066 - PARADIGMAS DE LINGUAGENS DE PROGRAMAÇÃO EM PYTHON

PROFESSOR: LUCAS SAMPAIO LEITE

Correções dos exercícios de Strings

https://github.com/lucassampaioleite/python-exercises/blob/main/005_strings.ipynb

Agenda

 Listas


Listas

- ❑ Em Python, uma lista é uma sequência mutável de n valores que podem ser de qualquer tipo (inclusive outras listas);
- ❑ Assim, uma lista pode ser entendida como um vetor de elementos que podem ser de qualquer tipo;
- ❑ As listas são exatamente iguais às strings, exceto pelo fato de strings serem imutáveis e listas serem mutáveis.

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

Listas

- ❑ Uma lista é conjunto de elementos ou um agrupamento;
- ❑ Como definir uma lista em Python?
- ❑ Usando colchetes e separando os elementos por vírgula;
- ❑ Exemplo: lista contendo números ímpares positivos menores que 10.

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

Listas

- ❑ Podemos acessar cada elemento da lista separadamente
 - ❑ Acesso indexado;
 - ❑ Usamos o nome da lista seguido da posição que queremos acessar entre colchetes.
- ❑ Exemplo: acessar e imprimir o primeiro elemento da lista:

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

- ❑ A contagem das posições da lista começa de zero:
 - ❑ Posição 0 -> primeiro elemento;
 - ❑ Posição 1 -> segundo elemento;
 - ❑ ...e assim por diante...
- ❑ Cuidado! Em uma lista com 'n' elementos, a última posição é a 'n-1'.

Listas

❑ 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])
```

Listas

```
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
```


Listas

- ❑ E se tentarmos acessar uma informação passando um índice inexistente?
- ❑ 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!!!

Listas

- ❑ Podemos adicionar novos elementos a uma lista;
- ❑ Utilizando o operador + (concatenação), adicionamos novos elementos ao final da lista.
- ❑ Exemplo:

```
main.py > ...  
1  lista = [1, 3, 5, 7, 9]  
2  print('Lista antes da primeira concatenação: ', lista)  
3  lista = lista+[11, 13]  
4  print('Lista após a primeira concatenação: ', lista)  
5  lista += [15, 17, 19]  
6  print('Lista após a segunda concatenação: ', lista)
```



```
Lista antes da primeira concatenação: [1, 3, 5, 7, 9]  
Lista após a primeira concatenação: [1, 3, 5, 7, 9, 11, 13]  
Lista após a segunda concatenação: [1, 3, 5, 7, 9, 11, 13, 15, 17, 19]
```

Listas

- ❑ Listas são mutáveis, logo, podemos alterar os valores dos seus elementos;
- ❑ Exemplo:

```
main.py > ...  
1  lista = [1, 3, 5, 7, 9]  
2  print('Lista antes das alterações: ', lista)  
3  lista[0] = 0  
4  lista[2] = 4  
5  print('Lista após as alterações: ', lista)
```



```
Lista antes das alterações: [1, 3, 5, 7, 9]  
Lista após as alterações: [0, 3, 4, 7, 9]
```

Listas

- ❑ O fatiamento de listas é a extração de um subconjunto dos elementos da lista:
 - ❑ Notação: Mesma do acesso indexado, usando colchetes [];
 - ❑ Porém, ao invés de especificar apenas uma posição, será informado um intervalo de posições;
 - ❑ Como definir um intervalo?
 - ❑ Índice da posição inicial, seguido de ':', seguido do índice da posição final:
 - ❑ Obs1: Posição inicial: intervalo fechado;
 - ❑ Obs2: Posição final: intervalo aberto.

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

Intervalo contendo os elementos das posições 2 e 3

Listas

- ❑ O fatiamento de uma lista retorna uma lista;
- ❑ Isso quer dizer que podemos armazenar o resultado do fatiamento de uma lista em uma segunda variável:

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



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

Listas

- ❑ Podemos definir um intervalo entre os elementos do subconjunto resultante do fatiamento;
- ❑ A seguinte notação é utilizada para isso:
 - ❑ Lista [<inicio> : <fim> : <intervalo>]
 - ❑ Logo, o fatiamento se aplica da posição <inicio> até imediatamente antes da posição <fim> pulando elementos de <intervalo> em <intervalo>

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



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

Listas

- ❑ No fatiamento, os valores de `lista[a:b:c]`, os valores de `a`, `b` e `c` podem ser omitidos;
- ❑ Quando omitidos, Python aplica o seguinte padrão:
 - ❑ `a = 0` (inicia sempre da posição 0);
 - ❑ `b = n` (termina sempre no último elemento);
 - ❑ Sendo `n` o número de elementos da lista;
 - ❑ `c = 1`.
 - ❑ O intervalo de variação dos elementos é de um em um (não pula elementos).

Listas

❏ Exemplos:

```
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])
```


Listas

❏ Exemplos:

```
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]
```

Listas

- ❑ Também é possível utilizar fatiamento para alterar uma lista;
- ❑ Mesma regra para definir o subconjunto da lista que será alterado é utilizado;
- ❑ No entanto, a lista e o seu subconjunto são definidos à esquerda do operador de atribuição (=).

O código atribui um subconjunto vazio à fatia que vai da posição 3 até antes da posição 7

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

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

Listas

- ❑ Note que no código abaixo que o fatiamento também pode funcionar para remover um elemento, ou subconjunto de elementos.

O código remove o terceiro elemento (posição 2);

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

Listas

- ❑ Note que no código abaixo que o fatiamento também pode funcionar para remover um elemento, ou subconjunto de elementos.

O código remove o terceiro elemento (posição 2);

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

A substituição não precisa ser por uma lista vazia!!!!

Listas

- ❑ Remoção também pode ser feita por meio do comando del;
- ❑ Especificamos as posições dos elementos a serem removidos por meio de colchetes.

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



```
Lista antes remoção:  [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]  
Lista após primeira remoção:  [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]  
Lista após primeira remoção:  [1, 2, 3, 4, 5, 6, 7, 8, 9]  
Lista após terceira remoção:  [1, 2, 8, 9]
```


Listas

- ❑ O método `append` adiciona elementos no fim da lista;

```
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]
```

Listas

- ❑ O tamanho de uma lista é obtido por meio do método `len()`.

```
main.py > ...  
1  lista = ['Esta lista é heterogênea', 0, 1, 1.5, 2.8,  
2      |      2, 3, 4, 5, 6, 7, 8, 8.55, 9, 10, True]  
3  print ('Lista hererogênea: ', lista)  
4  print ('O tamanho da lista é: ', len(lista))
```



```
Lista hererogênea: ['Esta lista é heterogênea', 0, 1, 1.5, 2.8, 2,  
3, 4, 5, 6, 7, 8, 8.55, 9, 10, True]  
O tamanho da lista é: 16
```

Listas

- ❑ Um elemento de uma lista também pode ser uma outra lista:
- ❑ Nesse caso, temos listas aninhadas!

```
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))
```



```
Lista heterogênea aninhada: ['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]  
O tamanho da lista é: 15
```


Listas

❏ Exemplo para acessar elementos da lista aninhada:

```
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])-1])  
8  print ('O tamanho da sublista é: ', len(lista[2]))
```

Listas

- ❑ Exemplo para acessar elementos da lista aninhada:

```
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])-1])  
8  print ('O tamanho da sublista é: ', len(lista[2]))
```



```
Lista heterogênea aninhada: ['Esta lista é heterogênea', 0, [0.5, 0  
.75, 1, 1.25, 1.75], 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 último elemento da posição 2 é: 1.75  
5
```

Exercícios

1. Faça um Programa que leia 5 números inteiros, armazene-os em uma lista e imprima essa lista na tela.
2. Faça um programa que leia 10 números inteiros. Cada numero par deve ser armazenado em uma lista de pares e cada impar tem que ser armazenado em uma lista de impares. Ao término do programa imprima as duas listas.
3. 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.
4. Modifique o programa da questão 3 para que o usuário indique a quantidade de alunos que será utilizada no programa. Assim antes de começar a leitura de idades e alturas, o programa deve solicitar ao usuário o quantitativo de alunos.
5. 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.

Mais exercícios...

6. 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
7. Faça um programa que receba a temperatura média de cada mês do ano e armazene-as em uma lista. Após isto, calcule a média anual das temperaturas e mostre todas as temperaturas acima da média anual, e em que mês elas ocorreram (mostrar o mês por extenso: 1 – Janeiro, 2 – Fevereiro, . . .).

Mais exercícios...

8. Em uma competição de salto em distância cada atleta tem direito a cinco saltos. O resultado do atleta será determinado pela média dos cinco valores restantes. Você deve fazer um programa que receba o nome e as cinco distâncias alcançadas pelo atleta em seus saltos e depois informe o nome, os saltos e a média dos saltos. O programa deve ser encerrado quando não for informado o nome do atleta. A saída do programa deve ser conforme o exemplo abaixo:

```
Atleta: Rodrigo Curvêllo

Primeiro Salto: 6.5 m
Segundo Salto: 6.1 m
Terceiro Salto: 6.2 m
Quarto Salto: 5.4 m
Quinto Salto: 5.3 m

Resultado final:
Atleta: Rodrigo Curvêllo
Saltos: 6.5 - 6.1 - 6.2 - 5.4 - 5.3
Média dos saltos: 5.9 m
```

Dúvidas???



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