centro universitário facvest
unifacvest

Fábrica de Software

Aula 04

Python

LISTAS

Como uma string, uma lista é uma sequência de valores. Em uma lista, os valores podem ser de qualquer tipo. Os valores de uma lista são chamados de elementos, ou algumas vezes, de itens. A forma mais simples de criar uma lista é colocando os valores entre colchetes [].

```
#Lista de números inteiros
exemplo = [10,20,30,40]
exemplo
```

```
[10, 20, 30, 40]
```

```
type(exemplo)
```

```
list
```

```
#lista de strings
exemplo2 = ['valor1','valor2','valor3','valor4']
exemplo2
```

```
['valor1', 'valor2', 'valor3', 'valor4']
```

```
type(exemplo2)
```

```
list
```

```
# lista mista
exemplo3 = ['valor10', 20, 30.5]
exemplo3
```

```
['valor10', 20, 30.5]
```

```
type(exemplo3)
```

```
list
```

Você pode colocar uma lista dentro de uma lista, trata-se de uma lista aninhada.

```
#Lista Aninhada

exemplo4 = [30,50,[10,20]]
exemplo4
```

```
[30, 50, [10, 20]]
```

Uma lista que não contém elementos é denominada lista vazia. Para criá-la é só colocar os colchetes.

```
# Lista Vazia
exemplo5 = []
exemplo5
```

```
[]
```

```
type(exemplo5)
```

```
list
```

Para acessar um elemento de uma lista, usamos a indezação. Você precisa lembrar que a contagem em python começa do zero. Assim, basta colocar a posição do elemento para acessá-lo.

Acessando os elementos pela posição

```
#Atribuindo valores à lista
lista = [10,20,30,'Valor1','Valor2','Valor3']
lista
```

```
[10, 20, 30, 'Valor1', 'Valor2', 'Valor3']
```

```
# Mostrando o elemento da posição zero
lista[0]
```

```
10
```

```
#Mostrando o último elemento
lista[5]
```

```
'Valor3'
```

```
#Mostrando o último elemento com -1
lista[-1]
```

```
'Valor3'
```

As listas são mutáveis. Podemos substituir um valor apresentando seu indice e atribuindo um novo valor a ele:

```
# Modificando a lista. Atribuindo novo valor ao índice 1
lista[1] = 2000
lista
```

```
[10, 2000, 30, 'Valor1', 'Valor2', 'Valor3']
```

Características

Qualquer expressão de números inteiros pode ser usada como índice.

Se tentar ler ou escrever um elemento que não existe, você recebe um IndexError

Se um índice tiver um valor negativo, ele conta de trás para a frente, a partir do final da lista.

O operador in também funciona como lista

Usando o operador in

```
# Operador in

'Valor2' in lista
```

```
True
```

```
'Valor 4' in lista
```

```
False
```

Operadores com listas

O operador + concatena (une) as listas e o operador * repete os valores um dado número de vezes.

Operadores

```
# Operador de concatenação +
a = [1,2,3]
b = [4,5,6]
c = a + b
```

```
c
```

```
[1, 2, 3, 4, 5, 6]
```

```
# Operador * repete a lista um dado número de vezes
p = [2,3]
p*3
```

```
[2, 3, 2, 3, 2, 3]
```

FATIAS DE LISTAS

O operador de fatias também funciona com listas:

```
# Fatias
p = [10,20,30,'Valor1','Valor2','Valor3', [1,2,3], True, False]
```

```
# Valores das posições 1 a 3
p[1:4]
```

```
[20, 30, 'Valor1']
```

Repare que ao acessar o índice, o primeiro entra no resultado e o último não. Assim, se você quer o número de 1 a 3, você deverá colocar [1:4]. Caso queira do valor 2 ao 4, deverá fazer: [2:5], assim por diante. Se você omitir o primeiro indice, a fatia começa no início. Se você omitir o segundo, a fatia vai até o final. Se você omitir ambos, a fatia é uma cópia da lista inteira.

Fatias

```
# Fatias
p = [10,20,30,'Valor1','Valor2','Valor3', [1,2,3], True, False]
```

```
# Omitindo o primeiro valor
p[:5]
```

```
[10, 20, 30, 'Valor1', 'Valor2']
```

```
# Omitindo o último valor
p[3:]
```

```
['Valor1', 'Valor2', 'Valor3', [1, 2, 3], True, False]
```

```
p[:]
```

```
[10, 20, 30, 'Valor1', 'Valor2', 'Valor3', [1, 2, 3], True, False]
```

Podemos também, atribuir novos valores a partir de fatias.

```
# Modificando os valores
p[1:3] = ['NovoValor',200]
```

```
p[:]
```

```
[10, 'NovoValor', 200, 'Valor1', 'Valor2', 'Valor3', [1, 2, 3], True, False]
```

Métodos de Listas

O python tem métodos que operam em listas. Por exemplo, o append(), o sort(), o sum(), pop(), len(), max(), min() e o count() .

O método append() adiciona um valor na última posição da lista. Você sempre deverá colocar a lista e o ponto, por tratar-se de um método.Como você pode perceber, o método `append()` aceita apenas um argumento: o elemento que você deseja adicionar. Este elemento pode ser qualquer tipo de dado

Append()

```
t = ['a','b','c','d']
t
```

```
['a', 'b', 'c', 'd']
```

```
t.append('E')
t
```

```
['a', 'b', 'c', 'd', 'E']
```

O método sort() ordena uma lista. Esse método é usado para ordenar uma lista diretamente, o que significa que ela realiza a **mutação** da lista ou que a modifica diretamente sem criar cópias adicionais.

Sort()

```
p = ['d','f','b','a','c','e']
p
```

```
['d', 'f', 'b', 'a', 'c', 'e']
```

```
p.sort()
p
```

```
['a', 'b', 'c', 'd', 'e', 'f']
```

A soma dos números na lista é necessária em todos os lugares. Python fornece uma função embutida sum() que resume os números da lista.

Sum()

```
p = [20,30,40,50]
sum(p)
```

```
140
```

pop() é uma função embutida no Python que remove e retorna o último valor da lista ou o valor de índice fornecido.

pop()

```
list1 = [ 1, 2, 3, 4, 5, 6 ]
list1.pop()
print(list1)
```

```
[1, 2, 3, 4, 5]
```

A função len() é uma das funções internas do `Python`. Ela retorna o comprimento de um objeto, por exemplo, ela pode retornar o número de itens em uma lista. Você pode usar a função com muitos tipos de dados diferentes.

len()

```
lista = ['José','Tadeu','Marcos','Felipe','Felisteu']
lista
```

```
['José', 'Tadeu', 'Marcos', 'Felipe', 'Felisteu']
```

```
len(lista)
```

```
5
```

```
lista2 = [2,5,6,5,8,9,4,2,3,1,5,6,8]
len(lista2)
```

```
13
```

A max() função retorna o maior item em um iterável ou o maior de dois ou mais argumentos.

max()

```
lista = [2,6,18,52,6,9,86,24,4,56,25,74]
lista
```

```
[2, 6, 18, 52, 6, 9, 86, 24, 4, 56, 25, 74]
```

```
max(lista)
```

```
86
```

A função Python min() retorna o menor valor ou o menor item em um iterável passado como seu parâmetro.

min()

```
lista = [2,6,18,52,6,9,86,24,4,56,25,74]
lista
```

```
[2, 6, 18, 52, 6, 9, 86, 24, 4, 56, 25, 74]
```

```
min(lista)
```

```
2
```

A função count() retorna a quantidade de vezes que um mesmo elemento está contido numa lista. Essa é uma excelente maneira que evita a implementação de um Laço de Repetição em busca de elementos iguais.

count()

```
lista = [2,6,18,52,6,9,86,6,24,4,56,25,6,74,55,6,33,26,6,48,87,6,42,55,23,32,14,6,44,42,8]
```

```
# Quantas vezes o número 6 aparece na lista?
lista.count(6)
```

```
8
```

```
#Quantas vezes o número 55 aparece na lista?
lista.count(55)
```

```
2
```

EXERCÍCIOS

1) Vamos fazer uma lista de compras. Peça ao usuário 3 produtos, coloque em uma lista e mostre a lista final em um print**.** (batata,leite e pão)

2) Adicione mais produto à lista sem apagá-la. Adicione: banana, chocolate, cenoura,ervilha,café,sucrilhos,arroz,vinagre,salgadinho e pimenta.

3) Você é estranho. Não consegue ficar sem colocar a lista em ordem alfabética. Faça isso então.

4) Você é a Dory. Esqueceu se colocou pinhão e cenoura na sua lista. Verifique se estão lá.

5) Crie uma lista com: 3 strings, 4 números inteiros, 3 números float, duas listas aninhadas.

6) Crie uma lista com: 2 strings, 3 números inteiros, 2 números float, sendo que os elementos devem ser dados pelo usuário e impressos no final com um print.

7) Faça uma lista com os números a seguir e verifique :

5,4,8,9,6,54,52,68,102,48,201,55,60,31,4,50,8,4,33,123,87,66,2,4,82,102,44,6,32,26,4,14,25

a) Verifique se há o número 23 na lista

b) Quantas vezes o número 4 aparece?

c) Coloque a lista em ordem crescente

d) Coloque a lista em ordem decrescente

e) Verique a quantidade total de números

f) Verifique a soma dos números

g) Mostre o valor máximo e o valor mínimo

h) Sorteie uma número aleatório da lista e imprima com print