**FACULDADE DE COMPUTAÇÃO E INFORMÁTICA**

**CIÊNCIA DA COMPUTAÇÃO E SISTEMAS DE INFORMAÇÃO**

**Linguagem de Programação I**

**Aula 04: Listas em Python – Parte I**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | http://images.flatworldknowledge.com/ketchen/ketchen-fig05_x001.jpg | Nossos objetivos nesta aula são:   * Entender o que é uma lista na programação de computadores. * Conhecer quais as operações comumente efetuadas nas listas * Criar e usar listas em Python. | | Macintosh HD:Users:anacris:Desktop:Captura de Tela 2017-02-19 às 18.57.35.png | A referência para esta aula está no **Capítulo 4** do livro:  DIERBACH, C. *Introduction to Computer Science Using Python: A Computational Problem Solving Focus.* 1st Edition, New York: Wiley, 2012. | |
|  |

# O que é uma lista?

Nós utilizamos as listas frequentemente no nosso dia-a-dia: lista de compras, lista de convidados, lista de tarefas etc. É uma maneira de organizar os dados.

Abaixo nós temos um exemplo de uma lista de compras. Note que o primeiro elemento da lista é o *Cereal*, o segundo é o *Leite*, o terceiro é a *Banana* etc.

|  |
| --- |
| ***Lista de compras*** |
| 0 - Cereal |
| 1 - Leite |
| 2 - Banana |
| 3 - Maçã |
| 4 - Iogurte |

A ordem em que os elementos aparecem na lista pode ser a ordem em que o autor planeja pegar os itens no mercado, ou pode ser simplesmente a ordem em que ele anotou os itens à medida que foi se lembrando.

Na programação de computadores é usual numerar os índices a partir do 0, por isso a nossa contagem na lista de compras começa do 0.

Na programação de computadores, o conceito de lista é similar ao do nosso dia-a-dia. Uma lista é uma **estrutura de dados** que organiza seus elementos de forma linear, ou seja, dentro dela há o primeiro elemento, o segundo, o terceiro e assim por diante.

Uma lista é uma estrutura de dados linear que mantém a posição de seus elementos.

# Operações em uma lista

As operações que normalmente são realizadas em uma lista são: recuperar, substituir, inserir, remover e acrescentar.

## Recuperar (*retrieve*)

É comum querermos recuperar o item que está em uma determinada posição na lista. Na nossa lista de compras, o item que está na posição de índice 2 é a *Banana*.

|  |
| --- |
| ***Lista de compras*** |
| 0 - Cereal |
| 1 - Leite |
| **2 - Banana** |
| 3 - Maçã |
| 4 - Iogurte |

## Substituir (*replace*)

Se trocarmos o item na posição 3 da lista por *Mamão* estaremos realizando uma **substituição**.

|  |  |
| --- | --- |
| ***Antes*** | ***Depois*** |
| 0 - Cereal | 0 - Cereal |
| 1 - Leite | 1 - Leite |
| 2 - Banana | 2 - Banana |
| 3 - Maçã | **3 - Mamão** |
| 4 - Iogurte | 4 - Iogurte |

## Inserir (*insert*)

Suponha que o autor da lista se lembrou de que ele deve pegar mais uma fruta (*Melão*) antes de ir para a seção de iogurtes. Neste caso, ele irá **inserir** na posição 4 da lista o novo item.

|  |  |
| --- | --- |
| ***Antes*** | ***Depois*** |
| 0 - Cereal | 0 - Cereal |
| 1 - Leite | 1 - Leite |
| 2 - Banana | 2 - Banana |
| 3 - Maçã | 3 - Maçã |
| 4 - Iogurte | **4 - Melão** |
|  | 5 - Iogurte |

Note que o item que antes estava na posição número 4 (*Iogurte*) passou agora para a posição de número 5, e o número total de itens aumentou de 5 para 6.

## Remover (*remove*)

Se o autor da lista não precisar mais de *Leite*, ele pode remover o item da posição número 1.

|  |  |
| --- | --- |
| ***Antes*** | ***Depois*** |
| 0 - Cereal | 0 - Cereal |
| **1 - Leite** | 1 - Banana |
| 2 - Banana | 2 - Maçã |
| 3 - Maçã | 3 - Melão |
| 4 - Melão | 4 - Iogurte |
| 5 - Iogurte |  |

Note que todos os itens que estavam abaixo do *Leite* na lista "subiram" uma posição.

## Acrescentar (*append*)

Caso o autor da lista queira apenas adicionar mais um item à lista (*Mel*) sem que isto tenha que ser feito em uma posição específica, o mais fácil é acrescentá-lo no final da lista.

|  |  |
| --- | --- |
| ***Antes*** | ***Depois*** |
| 0 - Cereal | 0 - Cereal |
| 1 - Banana | 1 - Banana |
| 2 - Maçã | 2 - Maçã |
| 3 - Melão | 3 - Melão |
| 4 - Iogurte | 4 - Iogurte |
|  | **5 - Mel** |

# Listas em Python

Uma lista em Python é uma estrutura de dados linear, mutável (ou seja, pode ser modificada), de comprimento variável, que permite tipos diferentes de elementos. Ela é representada por uma lista de elementos separados por vírgula dentro de colchetes.

Exemplos de listas:

[1, 10, 3]  
['cereal', 'banana', 'maçã', 'melão', 'iogurte']  
[5, 'maçã', True, 10]  
[]

Na última linha destes exemplos temos somente um par de colchetes ([]), o que representa uma lista vazia.

Podemos inicializar uma variável com uma lista fazendo uso do operador atribuição.

lst = [1, 10, 3]

Após a atribuição, o conteúdo da lista pode ser acessado utilizando a variável.

Se quisermos imprimir todos os elementos da lista que é acessada através desta variável, basta utilizar a função **print**:

lista\_compras = ['cereal', 'leite', 'banana', 'maçã', 'iogurte']  
print(lista\_compras)

Para saber o número de itens em uma lista, usamos a função **len**:

lista\_compras = ['cereal', 'leite', 'banana', 'maçã', 'iogurte']  
n = len(lista\_compras)  
print('Número de itens: ', n)

## Operações nas listas em Python

Para **recuperar** (*retrieve*) elementos nós acessamos a lista através do seu índice, que é colocado entre colchetes logo após o nome da variável.

lista\_compras = ['cereal', 'leite', 'banana', 'maçã', 'iogurte']  
print('Elemento na posição de índice 1:', lista\_compras[1])

A contagem dos índices em uma lista em Python inicia-se em 0 (zero), e não em 1. Assim, se uma lista possui 5 itens, o primeiro elemento é acessado utilizando o índice 0, o último elemento é acessado utilizando o índice 4, e não há elemento com índice 5.

lista\_compras = ['cereal', 'leite', 'banana', 'maçã', 'iogurte']  
print('Primeiro item:', lista\_compras[0])  
print('Último item:', lista\_compras[4])

Podemos acessar vários elementos da lista em uma única linha de código e fazer operações com eles. Por exemplo:

lista\_notas = [8.0, 7.5, 7.0, 9.9]  
soma = lista\_notas[0] + lista\_notas[1] + lista\_notas[2] + lista\_notas[3]  
print('Soma:', soma)

Para **substituir** (*replace*), deve-se fazer a atribuição de um novo valor para uma posição específica da lista.

lista\_compras = ['cereal', 'leite', 'banana', 'maçã', 'iogurte']  
print('Antes:', lista\_compras)  
lista\_compras[3] = 'mamão'  
print('Depois:', lista\_compras)

A operação de **inserir** (*insert*) um elemento é realizada utilizando o método **insert** da lista. Para executá-lo deve-se fazer a chamada utilizando a **notação do ponto**. Para isso, escreve-se:

* o **nome da variável** que acessa a lista seguido por um **ponto** (.);
* o nome do método (**insert**);
* entre parênteses, o número do índice da posição onde deve ser realizada a inserção e o valor a ser inserido.

lista\_compras = ['cereal', 'leite', 'banana', 'mamão', 'iogurte']  
print('Antes:', lista\_compras)  
lista\_compras.insert(4, 'melão')  
print('Depois:', lista\_compras)

Para **remover** (*remove*) um elemento, deve-se utilizar a sentença **del** seguida do nome da variável e o índice do elemento que será removido.

lista\_compras = ['cereal', 'leite', 'banana', 'mamão', 'melão', 'iogurte']  
print('Antes:', lista\_compras)  
del lista\_compras[1]  
print('Depois:', lista\_compras)

A operação de **acrescentar** (*append*) um elemento no final da lista é feita utilizando o método **append**, informando qual o elemento que deve ser adicionado.

lista\_compras = ['cereal', 'banana', 'mamão', 'melão', 'iogurte']  
print('Antes:', lista\_compras)  
lista\_compras.append('mel')  
print('Depois:', lista\_compras)

Além disso, em Python temos à disposição o método **sort** (para alterar a posição dos elementos na lista de forma que fiquem em ordem crescente):

lista\_compras = ['cereal', 'banana', 'mamão', 'melão', 'iogurte', 'mel']  
print('Antes:', lista\_compras)  
lista\_compras.sort()  
print('Depois:', lista\_compras)

No caso deste exemplo, como os elementos da lista são strings, o método **sort** colocou os elementos em ordem alfabética.

Há também o método **reverse**, que inverte a posição de todos os elementos.

lista\_compras = ['cereal', 'banana', 'mamão', 'melão', 'iogurte', 'mel']  
print('Antes:', lista\_compras)  
lista\_compras.reverse()  
print('Depois:', lista\_compras)

# Exercícios

1. Quais serão os resultados apresentados se tentarmos executar as seguintes linhas em um interpretador Python?

>>> lst = [10, 20, 30]  
>>> lst  
???  
>>> lst[0]  
???  
>>> lst[0] = 5  
>>> lst  
???  
>>> del lst[2]  
>>> lst  
???  
>>> lst.insert(1,15)  
>>> lst  
???  
>>> lst.append(40)  
>>> lst  
???

**2)** Qual será a faixa dos valores dos índices para uma lista de 10 elementos?

* 1. 0-9
  2. 0-10
  3. 1-10
  4. 1-9

**3)** Qual dos seguintes itens NÃO é uma operação comum em listas?

* 1. recuperação
  2. substituição
  3. entrelaçamento
  4. inserção
  5. remoção

**4)** Qual será o resultado se inserirmos na lista abaixo o valor 50 na posição de índice 2?

|  |  |
| --- | --- |
| ***índice*** | ***elemento*** |
| 0 | 35 |
| 1 | 15 |
| 2 | 45 |
| 3 | 28 |

**5)** Escreva uma função não recursiva que receba como parâmetro uma lista de inteiros e retorne como resposta a soma de todos os elementos da lista.

**6)** Escreva uma função **recursiva** que receba como parâmetro uma lista de inteiros e retorne como resposta a soma de todos os elementos da lista.