



# Programação de Computadores

- ✓ Listas

# Conceitos abordados nesta aula

A proposta desta aula é apresentar as listas em Python e como podemos manipulá-las.

```
main.py
1  nomes = []
2  for i in range(5):
3      n = input("Digite um nome: ")
4      nomes.append(n)
5
6
7
8
9
10
11
12
13
```

The Python logo, consisting of two interlocking snakes, one blue and one yellow, is positioned to the right of the code block.

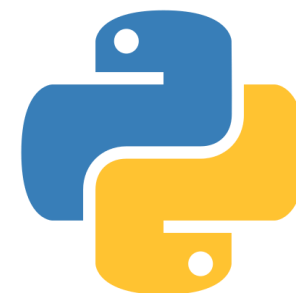
# Motivação

- ✓ Suponha que precisamos armazenar as notas de vários alunos.
- ✓ Com o conceito somente de variáveis, como faríamos para armazenar as notas de, por exemplo, 100 alunos?

main.py

```
1 nota1 = float(input("Entre com a nota do aluno 1: "))
2 nota2 = float(input("Entre com a nota do aluno 2: "))
3 nota3 = float(input("Entre com a nota do aluno 3: "))
4 ...
5 nota100 = float(input("Entre com a nota do aluno 100: "))
```

Certamente, criar 100 variáveis distintas **não** seria uma solução elegante.



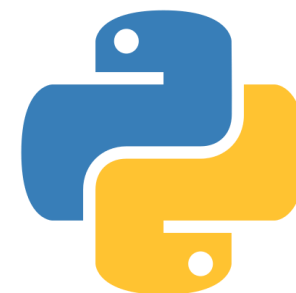
# O que são listas?

- ✓ Em Python, as listas são uma estrutura de dados muito importante e útil.
- ✓ Elas são usadas para armazenar um conjunto de valores em uma única variável. Listas são mutáveis, ou seja, podem ser modificadas após a sua criação.
- ✓ Além disso, as listas em Python são ordenadas, o que significa que cada item possui uma posição específica na lista.

```
notas = []  
for i in range(100):  
    nota = float(input('Digite a nota do {i+1}º aluno: '))  
    notas.append(nota)
```



**Armazenando notas de 100 alunos de uma turma!!!**



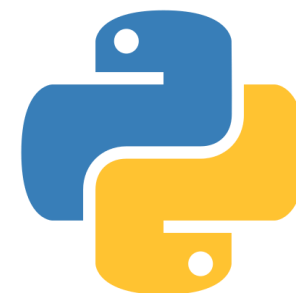
# O que são listas?

- ✓ Uma lista em Python é uma estrutura que armazena vários dados, que **podem ser de um mesmo tipo ou não**.
- ✓ Listas são construções de linguagens de programação que servem para armazenar vários dados de forma simplificada.

```
lista1 = [10, 20, 30, 40]
lista2 = ["programação", "computadores", "python"]
lista3 = ["oi", 2.0, 2, 5, "exemplo"]
```



- ✓ Características:
  - Acesso por meio de um índice inteiro.
  - Listas podem ser modificadas.
  - Pode-se incluir e remover itens de listas.

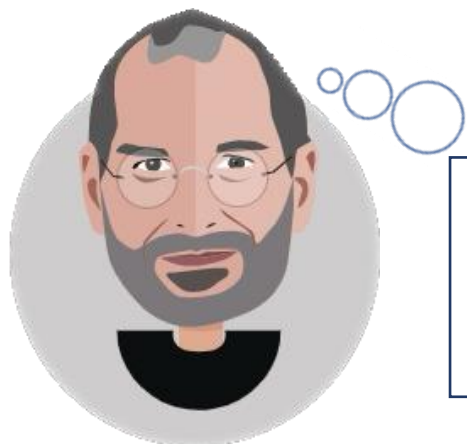


# Criação de listas em Python

Mais utilizado!

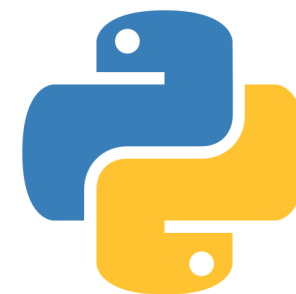
```
lista = []  
lista = list()  
lista = ['mamão', 10, 1.5, 'python']
```

Todos os exemplos produzem uma lista dinâmica



se  
Liga  
Aí

Observe que os dados armazenados nas listas não precisam ser de mesmo tipo.



# Como acessar um item em uma lista

- ✓ Pode-se acessar uma determinada posição da lista utilizando-se um índice de valor inteiro.
- ✓ A sintaxe para acesso de uma determinada posição é: **identificador[posição]**

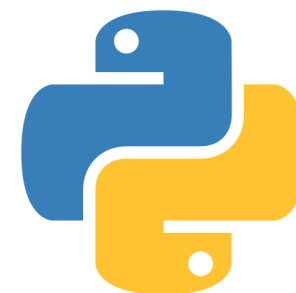
```
notas = [8.0, 5.5, 9.3, 7.6, 3.1]  
print(notas[1])
```

→ 5.5

**A primeira posição da lista tem índice 0**

Sendo  $n$  o tamanho da lista, os índices válidos para ela vão de 0 até  $n-1$ .

- ✓ A primeira posição da lista tem índice 0.
- ✓ A última posição da lista tem índice  $n-1$ .



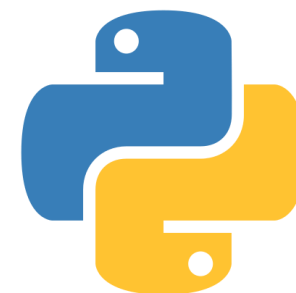
# Percorrendo uma lista

- ✓ A utilização de uma lista está associada a uma estrutura de repetição.
- ✓ Com isso podemos facilmente percorrer uma lista para consultas ou atualizações.

```
main.py
1 nomes = ['Marco', 'Eduardo', 'Mônica', 'Philippe']
2 for i in range(4):
3     print(nomes[i])
```

Exibindo os itens de uma lista!!

```
main.py
1 nomes = ['Marco', 'Eduardo', 'Mônica', 'Philippe']
2 for i in nomes:
3     print(i)
```



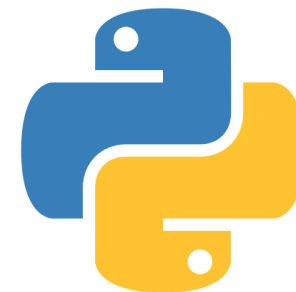


# Exemplos de aplicação

1- Faça um programa em Python que calcule a média de um aluno a partir de cinco notas previamente armazenadas em uma lista. Utilize a lista: **notas = [6, 7, 6.5, 4.8, 8]**

main.py

```
1 notas = [6,7,6.5,4.8,8]
2 soma = 0
3 for i in range(5):
4     soma = soma + notas[i]
5
6 media = soma/5
7 print("Média: %.2f" %media)
```



# Exemplos de aplicação

1- Faça um programa em Python que calcule a média de um aluno a partir de cinco notas previamente armazenadas em uma lista. Utilize a lista: `notas = [6, 7, 6.5, 4.8, 8]`

```
main.py
1 notas = [6,7,6.5,4.8,8]
2 soma = 0
3 for i in notas:
4     soma +=i
5
6 media = soma/5
7 print("Média: %.2f" %media)
```



Observe que não há somente uma forma de percorrer a lista.



# Principais métodos

11

<code>[😄, 😞].append(😬)</code>	→	<code>[😄, 😞, 😬]</code>
<code>[😄, 😞].insert(0, 😬)</code>	→	<code>[😬, 😄, 😞]</code>
<code>[😬, 😄, 😞].pop()</code>	→	<code>[😬, 😄]</code>
<code>[😬, 😄, 😞].pop(0)</code>	→	<code>[😄, 😞]</code>
<code>[😬, 😄, 🍷].remove(😄)</code>	→	<code>[😬, 🍷]</code>
<code>[🎓, 🍰, 🍷].index(🍰)</code>	→	1
<code>[🍷, 🎓, 🍰, 🍷].count(🍷)</code>	→	2
<code>[😄, 💀, 🦉, 🐼].reverse()</code>	→	<code>[🐼, 😄, 💀, 🦉]</code>
<code>[3, 7, 1, 5].sort()</code>	→	<code>[1, 3, 5, 7]</code>
<code>[🐼, 😄, 💀, 🦉].clear()</code>	→	<code>[]</code>
<code>[💀, 🤪, 🍷, 👍].len()</code>	→	4

# Principais métodos

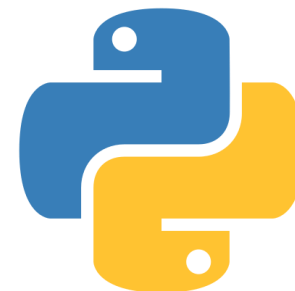
[ 😊, 😞 ].append( 😬 )

[ 😊, 😞, 😬 ]



se  
Liga  
Aí

O método **append()** adiciona um valor (n) ao final da lista!!!



```
▷ nomes = ['Marco', 'João', 'Maria']  
  nomes.append('Giulianna')  
  nomes  
[1] ✓ 0.1s  
... ['Marco', 'João', 'Maria', 'Giulianna']
```

# Principais métodos

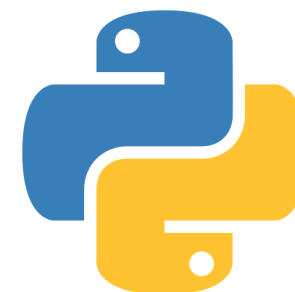
[😄, 😞].insert(0, 😟)

[😟, 😄, 😞]



se  
Liga  
Aí

O método **insert(pos, item)** adiciona um item à posição (pos) da lista!!



```
names = ['Marco', 'João', 'Maria']
names.insert(1, 'Giulianna')
names

[1] ✓ 0.1s

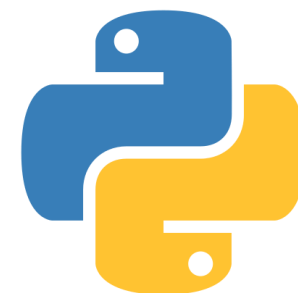
['Marco', 'Giulianna', 'João', 'Maria']
```

# Exemplos de aplicação

2- Vamos criar um programa em Python que solicite ao usuário o nome de 5 pessoas e armazene em uma lista. Em seguida o programa deve solicitar ao usuário um número de 0 a 4, correspondendo ao índice, e o programa deverá mostrar nome armazenado nesse índice.

main.py

```
1 nomes = [ ]
2 for i in range(5):
3     n = input("Digite um nome: ")
4     nomes.append(n)
5 print(nomes)
6 n=int(input("Digite um número: "))
7 print(nomes[n])
```



# Lista: outras funções

- ✓ A função **len()** retorna o tamanho da lista (nº de elementos):

```
nomes = ["Marco", "Maria", "João"]  
len(nomes)
```

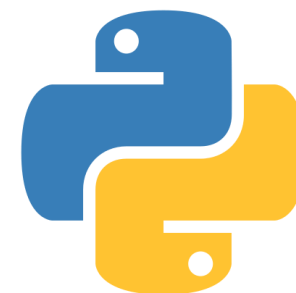
6



**Retorna a quantidade de elementos da lista**

- ✓ É muito comum usar a função **len** junto com o laço for para percorrer todas as posições de uma lista:

```
main.py  
1 notas = [8.0, 5.5, 9.3, 0.5, 3.1]  
2 for i in range(len(notas)):  
3     print(notas[i])  
4
```



# Exemplos de aplicação

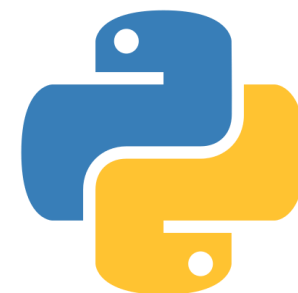
16

3- Faça um programa em Python que calcule e mostre a média de uma quantidade indeterminada de números inteiros digitados pelo usuário. Para sair o usuário deverá digitar 0. Use lista e exiba no final os números digitados.

```
num = []
soma = 0

while True:
    n = int(input('Digite um número inteiro: '))
    if n==0:
        break
    num.append(n)
    soma+=n

media = soma/len(num)
print(f'{media:.2f}')
print(num)
```





# Exemplos de aplicação

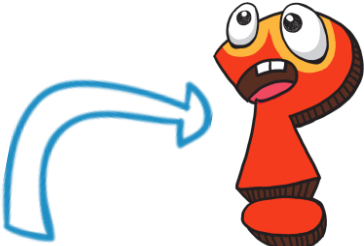
17

3- Faça um programa em Python que calcule e mostre a média de uma quantidade indeterminada de números inteiros digitados pelo usuário. Para sair o usuário deverá digitar 0. Use lista e exiba no final os números digitados.

```
num = []

while True:
    n = int(input('Digite um número inteiro: '))
    if n==0:
        break
    num.append(n)

media = sum(num)/ len(num)
print(f'{media:.2f}')
print(num)
```



se  
Liga  
Aí

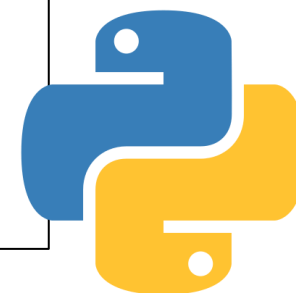
O Python tem uma função nativa dedicada à soma de todos os elementos de uma lista: a função sum()

# Exemplos de aplicação

4- Faça um programa que leia n notas, mostre as notas e a média.

```
notas = []
soma = 0
n = int(input('Entre com o número de notas: '))
for i in range(n):
    nota = float(input(f'Entre com a {i+1}ª nota: '))
    notas.append(nota)
    soma+=nota
print(notas)

soma = 0
media = soma/n
print(f'{media:.2f}')
```



# Lista: outras funções

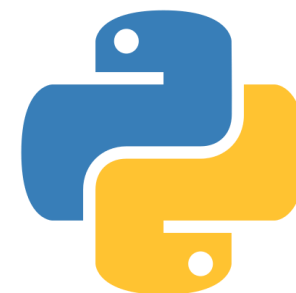
[😬, 😄, 🤖].remove(😄)

[😬, 🤖]

- ✓ A função **remove(item)** remove o primeiro item encontrado na lista cujo valor é igual a item.

```
nomes = ["Marco", "Maria", "João"]  
len(nomes)  
nomes.remove("Marco")  
nomes
```

➞ ['Maria', 'João']

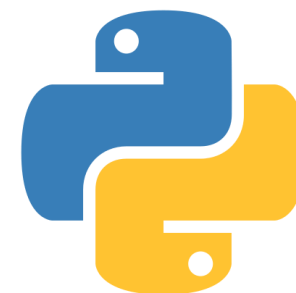


# Lista: outras funções

- ✓ A função **enumerate** gera uma tupla(\*) em que o primeiro valor é o índice e o segundo é o elemento da lista sendo enumerada.

```
nomes = ["Marco", "Maria", "João"]  
for x, e in enumerate(nomes)  
    print(f"[{x+1}]- {e}")
```

```
➞ [1]- Marco  
   [2]- Maria  
   [3]- João
```



(\*)Tuplas são similares às listas, porém são imutáveis!

# Resumo dos métodos

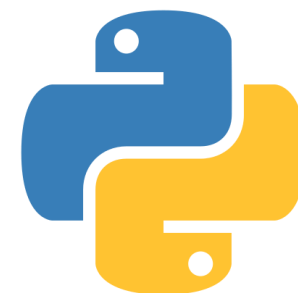
Método	Parâmetros	Descrição
<b>append</b>	item	Acrescenta um novo item no final da lista
<b>insert</b>	posição, item	Insere um novo item na posição dada
<b>pop</b>	nenhum	Remove e retorna o último item
<b>pop</b>	posição	Remove e retorna o item da posição
<b>sort</b>	nenhum	Ordena a lista
<b>reverse</b>	nenhum	Ordena a lista em ordem reversa
<b>index</b>	item	Retorna a posição da primeira ocorrência do item
<b>count</b>	item	Retorna o número de ocorrências do item
<b>remove</b>	item	Remove a primeira ocorrência do item
<b>enumerate</b>	nenhum	Exibe o índice da lista sendo enumerada



# Exemplos de aplicação

---

5- Faça um programa em Python que leia o nome e duas notas de n alunos e calcule a média. O usuário deverá digitar o número do aluno e o programa exibirá a média e o resultado, sabendo que o critério para aprovação é média igual ou maior que 6.0.



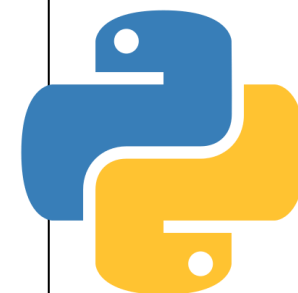
# Exemplos de aplicação

```
medias = []
nomes = []

x = int(input('Digite a quantidade de alunos: '))
for i in range(x):
    nome = input('Digite o nome do aluno: ')
    n1 = float(input(f'Qual a 1ª nota do {nome}? '))
    n2 = float(input(f'Qual a 2ª nota do {nome}? '))
    media = (n1+n2) / 2
    medias.append(media)
    nomes.append(nome)

print(10*'-')
n = int(input('Digite o nº do aluno que deseja exibir: '))
result = 'APROVADO' if medias[n] >= 6.0 else 'REPROVADO'

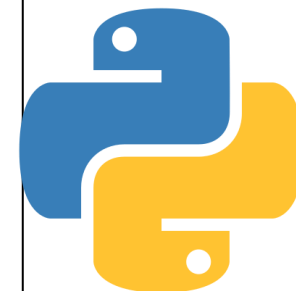
print('O aluno {nomes[n]} foi {result} com média {medias[n]:.2f}')
```



# Exemplos de aplicação

```
medias = []
nomes = []
x = int(input('Digite a quantidade de alunos: '))
for i in range(x):
    nome = input('Digite o nome do aluno: ')
    n1 = float(input(f'Qual a 1ª nota do {nome}? '))
    n2 = float(input(f'Qual a 2ª nota do {nome}? '))
    media = (n1+n2) / 2
    medias.append(media)
    nomes.append(nome)

print(10*'-')
n = input('Digite o nome do aluno que deseja exibir: ')
if n in nomes:
    i = nomes.index(n)
    result = 'APROVADO' if medias[i] >= 6.0 else 'REPROVADO'
    print(f'0 aluno {nomes[i]} foi {result} com média {medias[i]:.2f}')
else:
    print('Aluno não encontrado!')
```

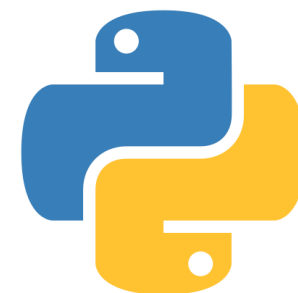




# Exemplos de aplicação

---

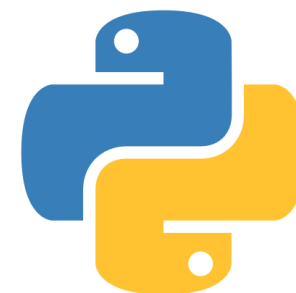
6- Vamos criar um programa em Python que solicite ao usuário o nome de 5 pessoas, armazene em uma lista e exiba os nomes digitados e o tamanho da lista. Em seguida o programa deve solicitar ao usuário um nome, e o programa deverá remover o nome armazenado na lista, exibir os nomes digitados e o tamanho da lista.



# Exemplos de aplicação

main.py

```
1  #Criação da lista
2  nomes = [] #ou nomes = list()
3  #Armazenar valores na lista
4  for i in range(5):
5      n = input("Digite um nome: ")
6      nomes.append(n) #adiciona no final da lista
7      #nomes.insert(i, n)
8
9  print(nomes) #mostra os itens da lista
10 print(len(nomes)) #quantidade de itens da lista
11 #exclui um item da lista
12 nome = input("Digite um nome para remover da lista: ")
13 if nome in nomes:
14     nomes.remove(nome) #remove o nome da lista
15     print(nomes) #mostra os itens da lista
16     print(len(nomes)) #mostra o tamanho da lista
17 else:
18     print("Nome não encontrado!")
```



# Exemplos de aplicação

---

7- Vamos fazer um programa em Python que controle a utilização de 5 salas do cinema CINEMARKO. O programa deverá ter as seguintes funcionalidades:

- Uma lista deverá armazenar os lugares vagos por sala: `lugaresVagos = [10, 5, 6, 8, 0]`, respectivamente para as sala 1, 2, 3, 4 e 5.
- O usuário deverá digitar o número da sala e a quantidade de ingressos que deseja comprar, ou zero para encerrar o programa.
- O programa deverá verificar se a venda é possível antes de concretizá-la, informando quando não há lugares disponíveis para venda.
- Caso a compra seja efetivada, atualizar o número de lugares livres e exibir na tela.

# Exemplos de aplicação

```
main.py
1  lugaresVagos=[10,2,3,4,0]
2  x=1
3  print("Bem vindos ao CINEMARKO")
4  for s in lugaresVagos:
5      print("Sala %d: %d lugares vagos"%(x,s))
6      x+=1
7  while True:
8      sala = int(input("Escolha uma sala (0 para sair): "))
9      if sala==0:
10         print("Até logo")
11         break
12     elif sala>len(lugaresVagos):
13         print("Sala inválida!!\n")
14     elif lugaresVagos[sala-1]==0:
15         print("Desculpe! Sala lotada!\n")
```

# Exemplos de aplicação

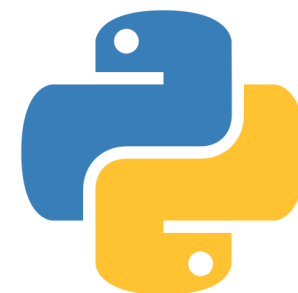
```
16     else:
17         compra = int(input("Quantos ingressos você deseja (%d vagos) :"%lugaresVagos[sala-1]))
18         if compra>lugaresVagos[sala-1]:
19             print("Desculpe! Número de ingressos indisponível\n!!")
20         elif compra<=0:
21             print("Número inválido\n!!")
22         else:
23             lugaresVagos[sala-1]-=compra
24             print("%d ingressos vendidos! Bom filme"%compra)
25             break
26     print("Utilização das salas:")
27     for x,s in enumerate(lugaresVagos):
28         print("Sala %d - %d lugar(es) vago(s)"%(x+1,s))
```

# Material Complementar

---

<https://www.devmedia.com.br/como-trabalhar-com-listas-em-python/37460>

<http://devfuria.com.br/python/listas/>



# Alguma dúvida????

---





# Exercícios de aplicação

---





# Exercício

---

- 1- Faça um programa em Python que contenha 3 listas com os nomes: valores, par e impar. Solicite N números inteiros ao usuário e armazene-os na lista chamada valores (utilize como critério de parada se o usuário deseja continuar).
- Após a obtenção dos dados, na lista par armazene apenas os números pares da lista valores e na lista ímpar os números ímpares. É obrigatório o uso de estrutura de repetição e listas.
  - Exiba os números armazenados nas 3 listas.

# Exercício

2- Faça um programa em Python que solicite ao usuário a placa e o valor da multa de 15 carros. As informações obtidas devem ser armazenadas em 2 listas distintas (observe que cada lista poderá ter apenas 15 itens armazenados e que na posição  $i$  das duas listas ficarão armazenados: a placa  $i$  e o valor de venda  $i$ , veja exemplo abaixo).

É obrigatório o uso de estrutura de repetição e listas. Calcule e mostre o valor médio de todas as multas e quantos carros possuem o valor de multa maior ou igual a R\$300.00, para isso utilize os dados armazenados nas listas descritas anteriormente e estrutura de repetição.

0	AAA-1234
1	CCC-1234
2	AAA-1234
3	DDD-1234
...	
14	BBB-1234

0	880.41
1	1467.35
2	293.47
3	293.47
...	
14	2934.70

# Exercício

---

3- Faça um programa em Python que solicite ao usuário o dia da semana e o volume de chuva correspondente a 10 dias. As informações obtidas devem ser armazenadas em 2 listas distintas (observe que cada lista poderá ter apenas 10 itens armazenados e que na posição  $i$  das duas listas ficarão armazenados: o dia da semana  $i$  e o volume de chuva  $i$ ). É obrigatório o uso de estrutura de repetição e listas.

Em seguida, calcule e mostre o volume médio de chuva apenas do dia de semana igual a quarta-feira e a soma total do volume de chuva, para isso utilize os dados armazenados nas listas. É obrigatório o uso de estrutura de repetição e das listas do exercício descritas anteriormente.

# Exercício

---

4- Criar um programa em Python que leia os dados necessários para cadastrar os nomes de N alunos em uma lista, em outra lista as respectivas notas dos alunos e em uma terceira lista o seu curso (ccp ou tads). Observe que na posição i das três listas ficarão guardados: o nome do aluno i, a nota do aluno i e o curso do aluno i.

Resolva os seguintes itens:

- a) Calcule e visualize a quantidade de alunos do curso de tads.
- b) Calcule e visualize a média das notas dos N alunos.
- c) Quantos alunos estão com a nota acima da média.

# Exercícios

---

5- Faça um programa em Python que solicite ao usuário, enquanto o mesmo desejar, números e armazene-os em uma lista.

Após a entrada de dados, somar os valores da lista, calcular e mostrar a média.

Calcule e mostre quantos números armazenados na lista estão acima da média.

# Créditos

Esta aula foi elaborada com base no material produzido e cedido gentilmente pelos **Professores Alcides, Lédon, Amilton e Cristiane.**





*That's all Folks!*