

# Listas

Joyce Teixeira



# Introdução

- Iniciaremos com o seguinte problema:
  - Ler as notas de 5 estudantes, calcular e informar a média da turma.

```
soma = 0
for i in range(5):
    nota = float(input("Digite a nota do estudante: "))
    soma = soma + nota

media = soma / 5
print(f'A média é: {media}')
```

# Introdução

- Se alterarmos o problema para:
  - Além de ler as notas de 5 estudantes, calcular e informar a média, ao final, imprima as notas que são superiores à média da turma.
  - Seria possível com a resolução anterior?

A nota sempre é substituída. Não iremos conseguir verificar quais são maiores que a média.

```
soma = 0
for i in range(5):
    nota = float(input("Digite a nota do estudante: "))
    soma = soma + nota

media = soma / 5
print(f'A média é: {media}')
```

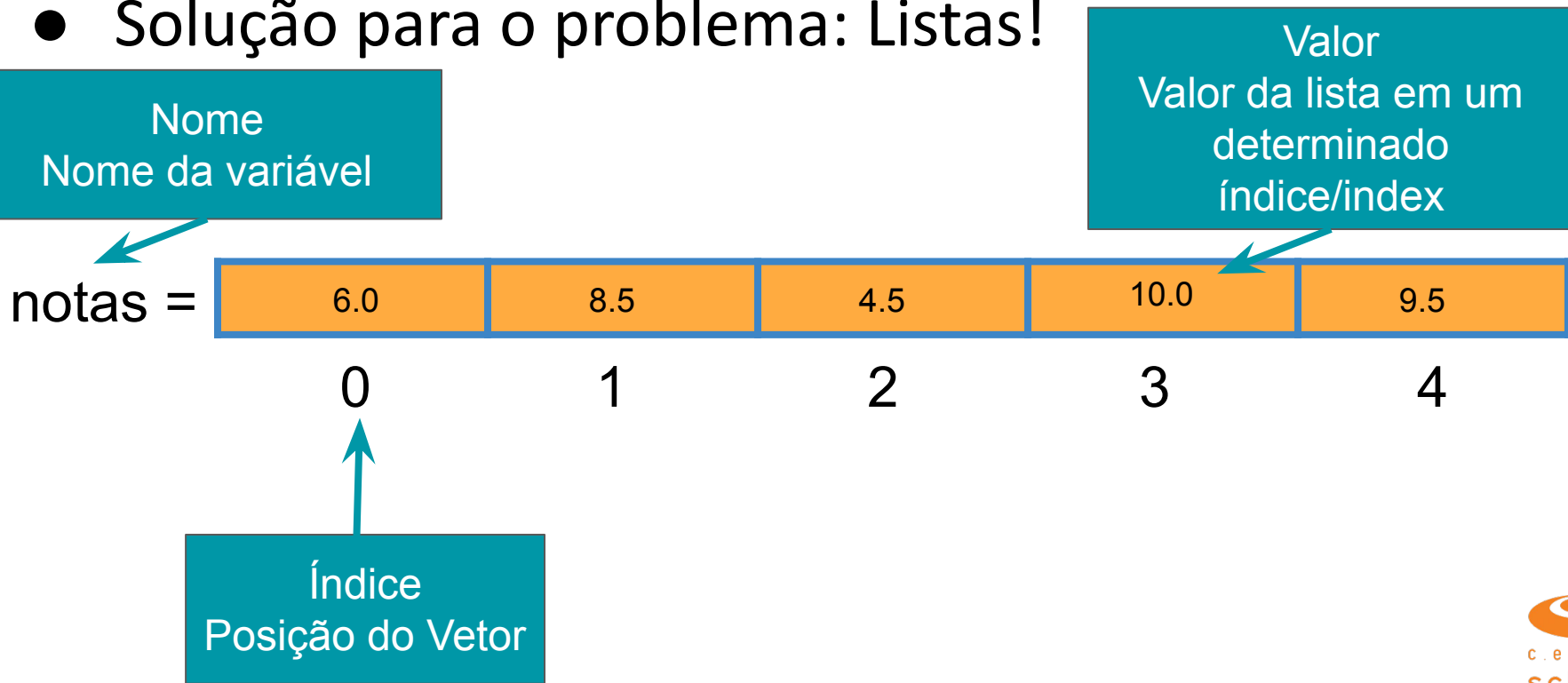


# Introdução

- Com o que sabemos hoje, como seria possível?
- Declarar 5 variáveis (nota1, nota2 ... nota5) é o único meio?
- Como podemos resolver este problema sem utilizar 5 variáveis?

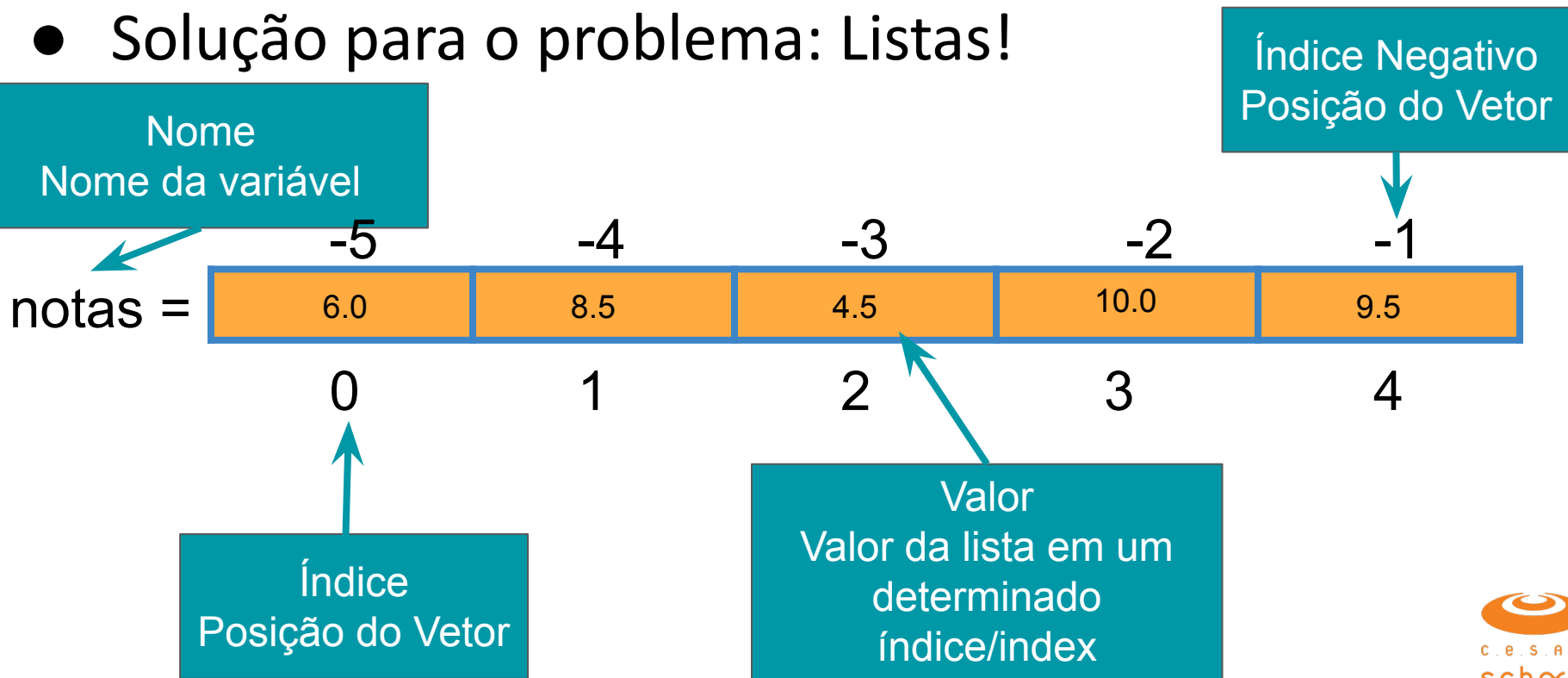
# Listas

- Solução para o problema: Listas!



# Listas

- Solução para o problema: Listas!

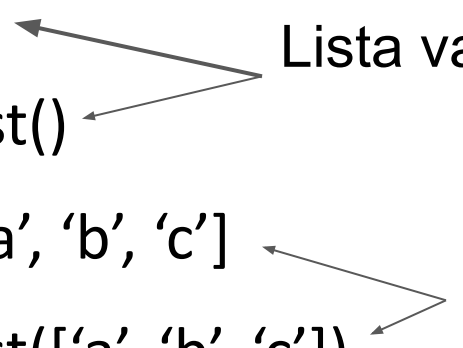


# Listas

- Em Python, uma lista é representada como uma sequência de objetos separados por vírgula e dentro de colchetes []:
- `notas = [6.0, 8.5, 4.5, 10.0, 9.5]`

# Criando Listas

- Há algumas maneiras de criar uma lista:

- `lista = []`
  - `lista = list()`
  - `lista = ['a', 'b', 'c']`
  - `lista = list(['a', 'b', 'c'])`
- Lista vazia
- Lista iniciada
- 



# Acessando Itens da Listas

```
notas = [6.0, 8.5, 4.5, 10.0, 9.5]
```

```
print(notas[1])      8.5
```

```
print(notas[3])      10.0
```

```
print(notas[-1])     9.5
```

```
print(notas[-5])     6.0
```

# Acessando Itens da Listas

- Como somar itens da lista?

```
notas = [6.0, 8.5, 4.5, 10.0, 9.5]  
  
notas2 = notas[2] + notas[4]  
print(notas2) 14
```

# Acessando Itens da Listas - Limites :

- Podemos acessar elementos da lista usando limites, definido

por :

○ lista[inicio:fim]



Sempre incluso no intervalo

Não incluso no intervalo

The diagram shows the list slicing syntax 'lista[inicio:fim]'. An arrow points from the text 'Sempre incluso no intervalo' to the colon ':' in the slice notation. Another arrow points from the text 'Não incluso no intervalo' to the 'fim' (end) part of the slice notation.

- notas = [6.0, 8.5, 4.5, 10.0, 9.5]

○ notas[2:4]    [4.5,10.0]

# Acessando Itens da Listas - Limites :

```
notas = [6.0, 8.5, 4.5, 10.0, 9.5]
```

```
print(notas[1:3])
```

```
print(notas[0:4])
```

```
print(notas[0:-1])
```

```
print(notas[0:5])
```

```
print(notas[-3:-1])
```

```
print(notas[:])
```

```
print(notas[2:])
```

```
print(notas[:2])
```

```
print(notas[-2:])
```

```
print(notas[:-2])
```

[8.5, 4.5]

[6.0, 8.5, 4.5, 10.0]

[6.0, 8.5, 4.5, 10.0]

[6.0, 8.5, 4.5, 10.0, 9.5]

[4.5, 10.0]

[6.0, 8.5, 4.5, 10.0, 9.5]

[4.5, 10.0, 9.5]

[6.0, 8.5]

[10.0, 9.5]

[6.0, 8.5, 4.5]

# Concatenação e Multiplicação de Listas

- É possível concatenar listas por meio do operador de adição **+** e multiplicá-las por um **inteiro**, o que gerará várias cópias dos seus itens.

# Concatenação e Multiplicação de Listas

- 

```
animais_domesticos = ['cachorro', 'gato', 'coelho']  
animais_selvagens = ['leao', 'zebra', 'girafa']  
  
animais = animais_domesticos + animais_selvagens  
  
print(animais)
```

```
['cachorro', 'gato', 'coelho', 'leao', 'zebra', 'girafa']
```

# Concatenação e Multiplicação de Listas

- 

```
animais_domesticos = ['cachorro', 'gato', 'coelho']  
animais_selvagens = ['leao', 'zebra', 'girafa']  
  
animais = animais_domesticos + animais_selvagens  
  
animais_total = animais * 3  
  
print(animais_total)
```

```
['cachorro', 'gato', 'coelho', 'leao', 'zebra', 'girafa', 'cachorro', 'gato',  
'coelho', 'leao', 'zebra', 'girafa', 'cachorro', 'gato', 'coelho', 'leao',  
'zebra', 'girafa']
```

# Tamanho da Lista

- O comprimento de uma lista, ou o número de itens que a compõem, pode ser obtido a partir da função **len()**

```
animais = ['cachorro', 'gato', 'coelho']  
  
print(len(animais))
```



# Mínimo, Máximo e Soma

- As funções **min()**, **max()** e **sum()**, encontram o menor valor da lista, maior valor da lista e realiza a soma de todos os elementos da lista, respectivamente.

# Mínimo, Máximo e Soma

```
notas = [6.0, 8.5, 4.5, 10.0, 9.5]

print("A menor nota foi:", min(notas))
print("A maior nota foi:", max(notas))
print("A soma de todas as notas foi:", sum(notas))
print("A média da turma foi:", sum(notas)/len(notas))
```

```
A menor nota foi: 4.5
A maior nota foi: 10.0
A soma de todas as notas foi: 38.5
A média da turma foi: 7.7
```

# Adição de Elementos na Lista

- Há duas maneiras de adicionar elementos na lista:
  - **append()** → adiciona um novo elemento no final da lista;
  - **insert()** → adiciona um novo elemento em uma posição (index) específica.

# Adição de Elementos na Lista

```
animais_domesticos = ['cachorro', 'gato', 'coelho']  
animais_domesticos.append('hamster')  
print(animais_domesticos)
```

```
['cachorro', 'gato', 'coelho', 'hamster']
```

# Adição de Elementos na Lista

```
animais_domesticos = ['cachorro', 'gato', 'coelho']  
  
animais_domesticos.append('hamster')  
animais_domesticos.insert(1, 'peixe')  
print(animais_domesticos)
```

```
['cachorro', 'peixe', 'gato', 'coelho', 'hamster']
```

# Verificando Item na Lista

- in

```
animais_domesticos = ['cachorro', 'gato', 'coelho']  
animais_selvagens = ['leao', 'zebra', 'girafa']  
  
print('leao' in animais_domesticos)  
print('girafa' in animais_selvagens)
```

```
False  
True
```

# Procurando Elemento e Posição na Lista

- A função `index()` procura um elemento na lista e retorna seu `index`.

```
animais_domesticos = ['cachorro', 'gato', 'coelho']  
|  
print(animais_domesticos.index('gato'))
```

# Remoção de Elementos da Lista

- Utilizando a função **pop()** é possível remover o último elemento da lista e também um elemento específico, informando o index.



# Remoção de Elementos da Lista

```
animais_domesticos = ['cachorro', 'peixe', 'gato', 'coelho', 'hamster']  
animais_domesticos.pop()  
  
print(animais_domesticos)
```

```
['cachorro', 'peixe', 'gato', 'coelho']
```

# Remoção de Elementos da Lista

```
animais_domesticos = ['cachorro', 'peixe', 'gato', 'coelho', 'hamster']  
animais_domesticos.pop(2)  
  
print(animais_domesticos)
```

```
['cachorro', 'peixe', 'coelho', 'hamster']
```

# Remoção de Elementos da Lista

- A função **remove()** é utilizada para remover um item a partir do seu valor.

# Remoção de Elementos da Lista

```
animais_domesticos = ['cachorro', 'peixe', 'gato', 'coelho', 'hamster']  
animais_domesticos.remove('coelho')  
  
print(animais_domesticos)
```

```
['cachorro', 'peixe', 'gato', 'hamster']
```

# Ordenando e Invertendo Listas

- A função **sort()** ordena a lista em ordem crescente para números e em ordem lexicográfica para strings;
- A função **reverse()** inverte a posição dos itens.

# Ordenando e Invertendo Listas

```
nomes = ['Maria', 'Ana', 'Cássio', 'Thiago', 'Zuleide', 'Elen']  
nomes.sort()  
print(nomes)  
['Ana', 'Cássio', 'Elen', 'Maria', 'Thiago', 'Zuleide']
```

```
idades = [10, 12, 3, 4, 2, 76, 22]  
idades.sort()  
print(idades)  
[2, 3, 4, 10, 12, 22, 76]
```

# Ordenando e Invertendo Listas

```
nomes = ['Maria', 'Ana', 'Cássio', 'Thiago', 'Zuleide', 'Elen']  
nomes.reverse()  
print(nomes)  
['Elen', 'Zuleide', 'Thiago', 'Cássio', 'Ana', 'Maria']
```

```
idades = [10, 12, 3, 4, 2, 76, 22]  
idades.reverse()  
print(idades)  
[22, 76, 2, 4, 3, 12, 10]
```

# Número de Ocorrências na Listas

- A função **count()** retorna o número de ocorrências de determinado objeto, passado como parâmetro, em uma lista.

```
animais_domesticos = ['cachorro', 'peixe', 'gato', 'coelho', 'hamster', 'gato']  
print(f"Há {animais_domesticos.count('gato')} gatos")  
print(f"Há {animais_domesticos.count('ovelhas')} ovelhas")
```

```
Há 2 gatos  
Há 0 ovelhas
```



# Copiando Listas

- Qual será o valor de **animais**?

```
animais_domesticos = ['cachorro', 'gato', 'coelho']  
animais = animais_domesticos  
animais_domesticos.append('dinossauro')  
  
print(animais)
```

```
['cachorro', 'gato', 'coelho', 'dinossauro']
```

# Copiando Listas

- Com `lista_2 = lista_1` não é criado uma nova cópia.
- É criado uma nova referência (`lista_2`) para o mesmo objeto de lista que `lista_1` já referencia.
- Ambas as variáveis apontam para o mesmo local na memória.
- Ao modificar a lista através de uma das variáveis, a mudança será refletida em ambas.

```
lista_1 = [1, 2, 3]
lista_2 = lista_1

lista_2.append(4)

print(lista_1)  # Saída: [1, 2, 3, 4]
print(lista_2)  # Saída: [1, 2, 3, 4]
```

# Copiando Listas

- Qual será o valor de **animais**?

```
animais_domesticos = ['cachorro', 'gato', 'coelho']  
animais = animais_domesticos[:]  
animais_domesticos.append('dinossauro')  
  
print(animais)
```

```
['cachorro', 'gato', 'coelho']
```

# Percorrendo Listas

```
notas = [6.0, 8.5, 4.5, 10.0, 9.5]
```

```
for i in notas:  
    print(i)
```

```
for i in range(len(notas)):  
    print(notas[i])
```



6.0

8.5

4.5

10.0

9.5

# Listas

- Voltando para o problema inicial:
  - Ler as notas de 5 alunos, calcular e informar a média;
  - Imprimir as notas que são superiores a média da turma.

# Listas

```
notas = []

for i in range(5):
    notas.append(float(input("Digite a nota do estudante: ")))

media = sum(notas) / 5
print(f"A média é: {media}")

for i in range(5):
    if notas[i] > media:
        print(f"A nota {notas[i]} foi maior que a média")
```

```
Digite a nota do aluno: 7
Digite a nota do aluno: 8
Digite a nota do aluno: 4
Digite a nota do aluno: 3
Digite a nota do aluno: 9.5
A média é: 6.3
```

```
A nota 7.0 foi maior que a média
A nota 8.0 foi maior que a média
A nota 9.5 foi maior que a média
```

# Listas

```
notas = []

for i in range(5):
    notas.append(float(input("Digite a nota do estudante: ")))

media = sum(notas) / 5
print(f"A média é: {media}")

for nota in notas:
    if nota > media:
        print(f"A nota {nota} foi maior que a média")
```

```
Digite a nota do aluno: 7
Digite a nota do aluno: 8
Digite a nota do aluno: 4
Digite a nota do aluno: 3
Digite a nota do aluno: 9.5
```

A média é: 6.3

A nota 7.0 foi maior que a média

A nota 8.0 foi maior que a média

A nota 9.5 foi maior que a média

# Listas

- Exercício
  - Preencher uma lista com 10 números do tipo inteiro, usando `input()`;
  - Somar todos os valores.



# Listas

- Exercício
  - Preencher uma lista com 10 números inteiros, usando input;
  - Mostrar os valores armazenados na lista;
  - Informar o menor elemento da lista usando o **min()**;
  - Informar o maior elemento da lista e seu índice sem o uso de **funções**;
  - Informar quantos valores ímpares existem na lista.

# Listas

- Exercício
  - Faça duas listas para armazenar as idades e alturas de 10 alunos, usando input;
  - O programa deverá determinar quantos alunos com mais de 13 anos possuem altura inferior à média das alturas dos alunos.

# Listas

- Exercício

- Faça um programa que faça 5 perguntas para uma pessoa sobre um crime. As perguntas são:
  - 1. "Telefonou para a vítima?"
  - 2. "Esteve no local do crime?"
  - 3. "Mora perto da vítima?"
  - 4. "Devia para a vítima?"
  - 5. "Já trabalhou com a vítima?"
- O programa deve no final emitir uma classificação sobre a participação da pessoa no crime. Se a pessoa responder positivamente a 2 questões ela deve ser classificada como "Suspeita", entre 3 e 4 como "Cúmplice" e 5 como "Assassino". Caso contrário, ele será classificado como "Inocente".

# Listas

- Lista de Exercício 05 disponível.