Aula 9 Sequencias

Profa. Simone de Abreu

Sequencias

- É uma estrutura de armazenamento de valores onde cada item pode ser acessado por um índice
 - ☐ 1° indice sempre é 0 (zero) e o último indice TAMANHO-1
 - O acesso aos itens é pelo operador subscrito []
- Listas, tuplas e strings são exemplos de sequencias

Listas vs Vetor (Array)

- Vetor (array) em outras linguagens é uma variável homogênea unidimensional capaz de armazenar um conjunto de valores do mesmo tipo
- ☐ Lista em Python não possuem as restrições dos vetores
 - ☐ É possível armazenar dados de tipos diferentes

```
lista = [100, 'A', 'Python', 2350.89]
notas = [8.5, 9, 10, 3.5, 4.5]
```

Listas – Criação

Criando uma lista vazia



☐ Criando uma lista com valores

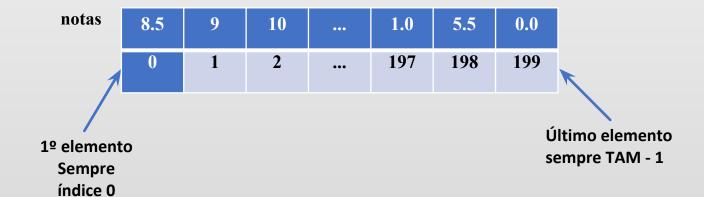
```
>>> notas = [8.5, 9, 10, 3.5, 4.5]
>>> notas
[8.5, 9, 10, 3.5, 4.5]
```

☐ Criando uma lista e inicializando com zero

```
>>> lista = [0] * 10
>>> lista
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

Listas – Índices

O índice indica a posição do dado na lista



- P1. Qual é o tamanho do vetor?
- P2. Qual é o índice da 5^a nota armazenada?
- P3. Qual é o dado do índice 198?
- P4. Qual é o 1º índice do vetor?
- P5. Qual é o último índice do vetor?
- P6. Qual é o 1º dado do vetor?

Listas – Operador subscrito []

Usa o operador subscrito [] para manipular cada dado

Cuidado ao acessar posições inválidas

```
>>> notas = [8.5, 9, 10, 3.5, 4.5]
>>> notas[1]
9
>>> notas[4]
4.5
>>> notas[5]
Traceback (most recent call last):
  File "<pyshell#5>", line 1, in <module>
    notas[5]
IndexError: list index out of range
```

Listas – Função len()

☐ Função built-in (interna) que retorna o número de itens de um objeto

```
>>> len(notas)
5
>>> len('python')
6
```

Entrada de Dados — Índice

☐ Mas com "caminhar" entre as posições da lista?

```
# criar uma lista com 5 posições
notas = [0.0] * 5

for i in range(len(notas)):
   notas[i] = float(input(f'Digite a {i+1}a nota: '))
```

índice

```
Digite a 1a nota: 8.5
Digite a 2a nota: 9
Digite a 3a nota: 10
Digite a 4a nota: 3.5
Digite a 5a nota: 4.5
```

Entrada de Dados – função append()

☐ Mas com "caminhar" entre as posições da lista?

```
# cria uma lista vazia
notas = []

for i in range(5):
   nota = float(input(f'Digite a {i+1}a nota: '))
   notas.append(nota)
```

```
Digite a 1a nota: 8.5
Digite a 2a nota: 9
Digite a 3a nota: 10
Digite a 4a nota: 3.5
Digite a 5a nota: 4.5
```

Saída de Dados

for i in range(5):
 print(notas[i])

8.5 9.0 10.0 3.5 4.5

for nota in notas:
 print(nota)

8.5 9.0 10.0 3.5 4.5

print(notas)

[8.5, 9.0, 10.0, 3.5, 4.5]

Exemplo 1

Ler 10 notas e armazena-las em uma lista.Imprimir as notas no final

```
🍦 exemplo1.py 🗡
aula9 > exemplo1.py > ...
   1 # cria uma lista vazia
      notas = []
      # laço para leitura de 10 notas
      for i in range(10):
   6
           nota = float(input(f'Digite a {i+1}a nota: '))
           notas.append(nota) # armazena a nota lida na lista
   8
      # laço para imprimir as notas da lista
       for nota in notas:
  10
           print(nota)
  11
```

Exemplo 2

☐ Encontrar o maior elemento em uma lista de 20 inteiros

```
maior.py X
aula9 > 👘 maior.py > ...
      import random
      numeros = []
   4
      # laço para armezenar os dados na lista
       for i in range(20):
           num = random.randint(0, 301) # gera o número
   8
           print(f'{num}', end = ' ')
   9
           numeros.append(num) # armazena na lista
  10
      # atribui o lo número da lista a variável maior
  12
      maior = numeros[0]
  13
      # encontra o maior
       for num in numeros:
  16
           if num > maior:
  17
               maior = num
  18
      print(f'\n\n0 maior número é: {maior}')
```



KEEP CALM AND **VAMOS** PRATICAR

Pense, Pareie, Compartilhe Em DUPLA

1. Leia 10 valores inteiros e armazene-os em uma Lista. Em seguida, imprima todos os elementos.

2. Leia 15 palavras e armazene-as em uma Lista. Em seguida, mostre todas as palavras da última para primeira.

3. Leia 8 elementos em uma Lista de inteiros A. Construa uma outra Lista B, de mesma dimensão de A, com seus elementos sendo a **multiplicação** dos elementos de A por 3. Mostre os elementos de B.

$$b[0] = a[0] * 3;$$

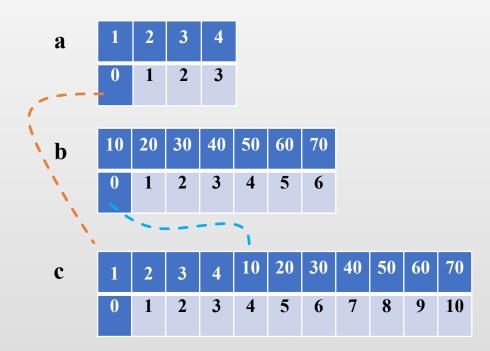
4. Leia duas listas A e B inteiros, de mesma dimensão. Construa a lista C, de mesma dimensão, cujo elementos de C é a subtração dos elementos de A por B. Mostre a lista C.

$$c[0] = a[0] - b[0]$$

5. Leia duas lista reais A (de tamanho 4) e B (de tamanho 7). Construa uma 3° lista que seja a junção de A e B (colocar os elementos de A seguidos dos elementos de B).

Faça a solução manipulando índice a índice.

Depois pesquise a função extend()



6. Sortear 1000 números inteiros e armazenar em uma lista. Percorrer o lista e encontrar o menor elemento.

7. Criar uma lista para armazenar 10.000 números inteiros gerados de forma aleatória (entre 1 e 10).

Após gerar a lista, calcular a quantidade de vezes que cada número foi sorteada.

O número 1 aparece 5 vezes

O número 2 aparece 2 vezes

. . . .

O número 10 aparece 3 vezes

8. Faça um programa que preencha por leitura uma lista de 8 elementos.

Ler um número e buscar esse número na lista. Ao final da busca, imprimir uma das duas mensagens: "Número X encontrado na lista" ou "Número X NÃO encontrado na lista"

Faça a solução manipulando os índices da lista!

Desafio de Programação _

Desafio – Litros por Km

- Construir um programa que leia em uma lista os modelos de cinco carros (exemplo: FUSCA, GOL, VECTRA, etc).
- Leia outra lista com o consumo desses carros, isto é, quantos quilômetros cada um desses carros faz com 1 litro de combustível. Calcule e mostre:
 - O modelo do carro mais econômico.
 - Quantos litros de combustível cada um dos carros cadastrados consome para percorrer uma distância de 1.000 quilômetros.