PYTHON: Vetores (Pesquisar e Excluir)

Jean Eduardo Glazar

Programação I

Curso de Sistemas de Informação
Ifes Campus Colatina

INSTITUTO FEDERAL
Espírito Santo

Remover um elemento

- Utiliza-se a função **del()**, passando a posição (índice) do vetor que se deseja remover.
- Os índices são reorganizados.

```
del( meuVetor[<índice>] )
```

Remover um elemento

```
meuVetor = [20, "Junho", 2023]
del ( meuVetor[1] )
```

ANTES

meu_vetor

0	20
1	“Junho
2	”2023

DEPOIS

meu_vetor

0	20
1	2023

Pesquisar em um vetor

- Percorrer do primeiro ao último, comparando cada posição com o que pretendemos pesquisar.
- Quando encontrar, pare de percorrer o vetor.
- Exemplo: admita a função `lerNomes(vetNomes)` que armazena em um vetor vários nomes.
- A função `pesquisarNome(vetor,pesq)` recebe dois parâmetros, o vetor e a informação que se pretende pesquisar, e retorna a posição do vetor onde está o elemento.
- Se não encontrar, retorna a posição -1, que é uma posição inválida.

Pesquisar em um vetor

```
### PROGRAMA PRINCIPAL ###
def main():
    vNomes = []     # Cria o vetor vazio
    lerNomes(vNomes)
    pesq = input("Nome para pesquisar: ")
    pos = pesquisarNome (vNomes, pesq)
    if pos >= 0:
      print("Nome encontrado: %s" %vNomes[pos] )
    else:
      print("Nome não encontrado.")

main()
```

Pesquisar em um vetor

```
def pesquisarNome (vetor, pesq):
    i = 0
    while i < len(vetor) :
        if vetor[i] == pesq:
            return i;
        else:
            i = i + 1
    return -1     # Se chegar aqui é porque
                  # não encontrou no vetor
```

- OBS.: Cuidado na comparação de strings. O Python é *case sensitive*.

Pesquisar e remover do vetor

```
### PROGRAMA PRINCIPAL ###
def main():
    vNomes = []     # Cria o vetor vazio
    lerNomes(vNomes)
    pesq = input("Nome para pesquisar: ")
    pos = pesquisarNome (vNomes, pesq)
    if pos >= 0:
        del ( vNomes[pos] )
    else:
        print("Nome não encontrado.")

main()
```

Comando FOR

- O comando **for** é apropriado para percorrer vetores.
- Não usa o contador para percorrer os índices. Faz isso automaticamente, pegando um elemento a cada iteração.

```
for e in meuVetor:
```

Variável que corresponde a cada elemento do vetor

Variável vetor

Exemplo: Comando FOR

- Percorre e imprime todos os elementos do vetor

EQUIVALENTES

Com for

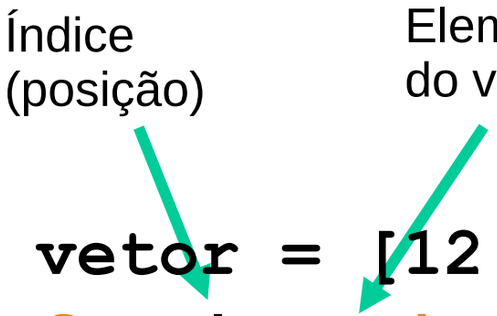
```
vetor = [12, 50, 130]
for e in vetor :
    print( e )
```

Com while

```
vetor = [12, 50, 130]
i = 0
while i < len(vetor):
    e = vetor[i]
    print( e )
    i = i + 1
```

Comando FOR

- Com o comando **for** simples, não é possível pegar o índice, apenas o elemento da lista.
- Para pegar o índice, devemos usar a função **enumerate**.




```
vetor = [12, 50, 130]
for i, e in enumerate(vetor):
    print("Posição: %d. Elemento: %d" %(i,e))
```

Pesquisar com FOR

```
def pesquisarNome (vetor, pesq):
    for i, e in enumerate(vetor):
        if e == pesq:
            return i;

    return -1     # Se chegar aqui é porque
                  # não encontrou no vetor
```

Comando RANGE

- A função **range** gera um intervalo de números. Exemplos:
  - **range(10)** → gera números de 0 a 9
  - **range(5,8)** → gera os números 5, 6 e 7
  - OBS.: O último número não entra no intervalo
  - **range(0,20,2)** → gera os números de 0 a 19 saltando de 2 em 2. Ou seja, gera somente os números pares.
- Podemos utilizar o **for** para percorrer o intervalo gerado pelo **range**.

Exemplo: Comando RANGE

- Percorre e imprime de 0 a 19.

EQUIVALENTES

Com for

```
for i in range(20):
    print( i )
```

Com while

```
i = 0
while i < 20:
    print( i )
    i = i + 1
```

Exemplo: Comando RANGE

- Imprime de 5 em 5 até 100

EQUIVALENTES

Com for

```
for i in range(5,101,5):
    print( i )
```

Com while

```
i = 5
while i < 101:
    print( i )
    i = i + 5
```