Disciplina BCM0505-15
Processamento da Informação

Mais sobre vetores

Profa. Carla Negri Lintzmayer

carla.negri@ufabc.edu.br

http://professor.ufabc.edu.br/~carla.negri

Centro de Matemática, Computação e Cognição
Universidade Federal do ABC

Agenda

Cópia de vetores

Strings

Outro problema básico

Pratique!

Cópia de vetores

Cópia

Com variáveis simples, operações de atribuição fazem cópias dos valores contidos nas variáveis:

1: INTEIRO: a, b
2: LEIA(a)
3: $b \leftarrow a$
4: ESCREVA(a, b)

```
1 a = int(input())
2 b = a
3 print(a, b)
```

Cópia

Com vetores, no entanto, uma atribuição **não fará** uma cópia dos valores contidos no vetor.

Elas apenas nos darão outro nome para que possamos nos referenciar ao mesmo vetor.

1: Inteiro: $A[100]$, $B[100]$, i, n
2: Leia(n)
3: **Para** $i \leftarrow 0$ **até** $n - 1$, **com** $i \leftarrow i + 1$ **faça**
4: Leia($A[i]$)
5: $B \leftarrow A$
6: Escreva($A[4]$, $B[4]$)
7: $B[4] \leftarrow 68$
8: Escreva($A[4]$, $B[4]$)

Cópia

Com vetores, no entanto, uma atribuição **não fará** uma cópia dos valores contidos no vetor.

Elas apenas nos darão outro nome para que possamos nos referenciar ao mesmo vetor.

```
1  n = int(input())
2  A = []
3  for i in range(n):
4      A.append(int(input()))
5  B = A
6  print(A[4], B[4])
7  B[4] = 68
8  print(A[4], B[4])
```

Cópia

Então, se o nosso objetivo é realmente obter uma cópia de um vetor, precisamos fazer isso **elemento a elemento**.

1: INTEIRO: $A[100]$, $B[100]$, i, n
2: LEIA(n)
3: **Para** $i \leftarrow 0$ **até** $n - 1$, **com** $i \leftarrow i + 1$ **faça**
4: LEIA($A[i]$)
5: $B[i] \leftarrow A[i]$
6: ESCREVA($A[4]$, $B[4]$)
7: $B[4] \leftarrow 68$
8: ESCREVA($A[4]$, $B[4]$)

Cópia

Então, se o nosso objetivo é realmente obter uma cópia de um vetor, precisamos fazer isso **elemento a elemento**.

```
1  n = int(input())
2  A = []
3  B = []
4  for i in range(n):
5      A.append(int(input()))
6      B.append(A[i])
7  print(A[4], B[4])
8  B[4] = 68
9  print(A[4], B[4])
```

Strings

Strings

- Strings são caracteres aglomerados.
- Em muitas linguagens de programação, são **vetores do tipo caractere**.
  - Em Python, não é exatamente uma lista, pois é **imutável**.
- Por isso, podemos acessar cada caractere de uma string.

Strings em algoritmos

Em algoritmos, vamos considerar que existe uma função StringLen, que recebe uma string e devolve a quantidade de elementos da mesma.

```
1: String: frase
2: Inteiro: n, espacos
3: Leia(frase)
4: n ← StringLen(frase)
5: espacos ← 0
6: Para i ← 0 até n − 1, com i ← i + 1 faça
7:     Se frase[i] = ' ' então
8:         espacos ← espacos + 1
9: Escreva("A frase dada tem", espacos + 1, "palavras.")
```

Strings em Python

```
1  frase = input()
2  vogais = ["a", "e", "i", "o", "u"]
3  cont_vogais = 0
4  for i in range(len(frase)):
5      j = Busca(vogais, frase[i])
6      if j != -1:
7          cont_vogais = cont_vogais + 1
8  print("A frase dada contém %d vogais." % (cont_vogais))
```

Strings em Python

```
1  frase = input()
2  vogais = ["a", "e", "i", "o", "u"]
3  cont_vogais = 0
4  for i in range(len(frase)):
5      j = Busca(vogais, frase[i])
6      if j != -1:
7          cont_vogais = cont_vogais + 1
8  print("A frase dada contém %d vogais." % (cont_vogais))
```

ERRADO!

Strings em Python

```
1  frase = input()
2  vogais = ["a", "A", "e", "E", "i", "I", "o", "O", "u", "U"]
3  cont_vogais = 0
4  for i in range(len(frase)):
5      j = Busca(vogais, frase[i])
6      if j != -1:
7          cont_vogais = cont_vogais + 1
8  print("A frase dada contém %d vogais." % (cont_vogais))
```

Versão bem Pythoniana

```
1  frase = input()
2  vogais = ["a", "A", "e", "E", "i", "I", "o", "O", "u", "U"]
3  cont_vogais = 0
4  for letra in frase:
5      if letra in vogais:
6          cont_vogais = cont_vogais + 1
7  print("A frase dada contém %d vogais." % (cont_vogais))
```

Outro problema básico

O problema da ordenação

Dada uma coleção de elementos com uma relação de ordem entre si, ordenar esses elementos.

O problema da ordenação

Dada uma coleção de elementos com uma relação de ordem entre si, ordenar esses elementos.

$3, 7, 1, 9, 2 \quad \rightarrow \quad 1, 2, 3, 7, 9$

O problema da ordenação

Dada uma coleção de elementos com uma relação de ordem entre si, ordenar esses elementos.

3, 7, 1, 9, 2 $\quad \rightarrow \quad$ 1, 2, 3, 7, 9

"manga", "abacate", "laranja", "banana", "abacaxi" $\quad \rightarrow$
"abacate", "abacaxi", "banana", "laranja", "manga"

O problema da ordenação

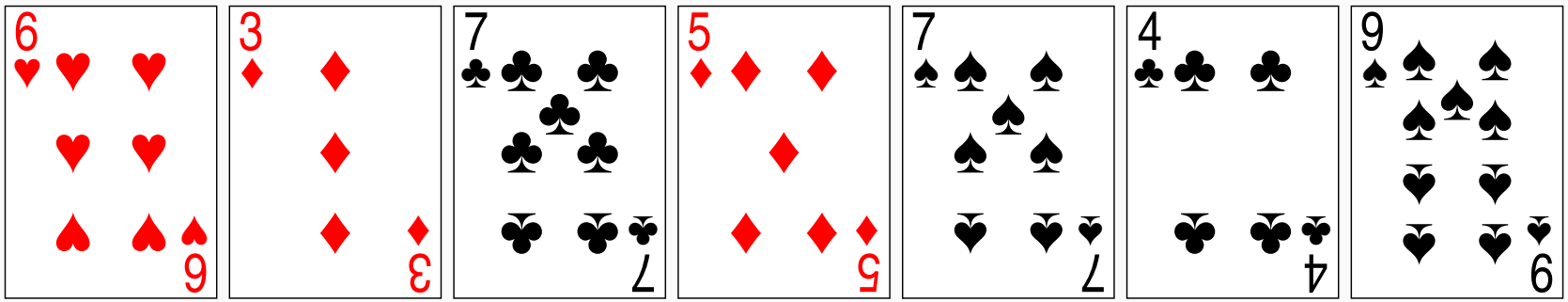
- É um dos mais básicos, com várias aplicações diretas ou indiretas.
  - Criar *rankings*
  - Definir preferências em atendimentos por prioridades
  - Criar listas
  - Otimizar processos de busca
  - Manter estruturas em bancos de dados
  - ...

Ordenação

Existem **muitos** algoritmos de ordenação:

- Bogosort
- Bubble Sort
- Heapsort
- Insertion Sort
- Merge Sort
- Quicksort
- Selection Sort
- Shell Sort
- https://www.youtube.com/watch?v=ZZuD6iUe3Pc

Ordenação



Selection Sort

- Ache o menor elemento do vetor e troque-o com o elemento da posição 0.

Selection Sort

- Ache o menor elemento do vetor e troque-o com o elemento da posição 0.
- Ache o segundo menor elemento do vetor e troque-o com o elemento da posição 1.

Selection Sort

- Ache o menor elemento do vetor e troque-o com o elemento da posição 0.
- Ache o segundo menor elemento do vetor e troque-o com o elemento da posição 1.
- Ache o terceiro menor elemento do vetor e troque-o com o elemento da posição 2.

Selection Sort

- Ache o menor elemento do vetor e troque-o com o elemento da posição 0.
- Ache o segundo menor elemento do vetor e troque-o com o elemento da posição 1.
- Ache o terceiro menor elemento do vetor e troque-o com o elemento da posição 2.
- ...

Selection Sort

- Ache o menor elemento do vetor e troque-o com o elemento da posição 0.
- Ache o segundo menor elemento do vetor e troque-o com o elemento da posição 1.
- Ache o terceiro menor elemento do vetor e troque-o com o elemento da posição 2.
- ...
- Ache o penúltimo menor elemento do vetor e troque-o com o elemento da posição $n - 1$.

Selection Sort

- Ache o menor elemento do vetor e troque-o com o elemento da posição 0.
- Ache o segundo menor elemento do vetor e troque-o com o elemento da posição 1.
- Ache o terceiro menor elemento do vetor e troque-o com o elemento da posição 2.
- ...
- Ache o penúltimo menor elemento do vetor e troque-o com o elemento da posição $n - 1$.
- Ache o último menor elemento do vetor e troque-o com o elemento da posição n.

Selection Sort

- Ache o menor elemento do vetor a partir da posição 0 e troque-o com o elemento da posição 0.

Selection Sort

- Ache o menor elemento do vetor a partir da posição 0 e troque-o com o elemento da posição 0.
- Ache o menor elemento do vetor a partir da posição 1 e troque-o com o elemento da posição 1.

Selection Sort

- Ache o menor elemento do vetor a partir da posição 0 e troque-o com o elemento da posição 0.
- Ache o menor elemento do vetor a partir da posição 1 e troque-o com o elemento da posição 1.
- Ache o menor elemento do vetor a partir da posição 2 e troque-o com o elemento da posição 2.

Selection Sort

- Ache o menor elemento do vetor a partir da posição 0 e troque-o com o elemento da posição 0.
- Ache o menor elemento do vetor a partir da posição 1 e troque-o com o elemento da posição 1.
- Ache o menor elemento do vetor a partir da posição 2 e troque-o com o elemento da posição 2.
- ...

Selection Sort

- Ache o menor elemento do vetor a partir da posição 0 e troque-o com o elemento da posição 0.
- Ache o menor elemento do vetor a partir da posição 1 e troque-o com o elemento da posição 1.
- Ache o menor elemento do vetor a partir da posição 2 e troque-o com o elemento da posição 2.
- ...
- Ache o menor elemento do vetor a partir da posição $n-2$ e troque-o com o elemento da posição $n-2$.

Selection Sort

```
1  def indice_menor(vet: [int], ini: int) -> int:
2      imin = ini
3      for i in range(ini+1, len(vet)):
4          if vet[i] < vet[imin]
5              imin = i
6      return imin
```

Selection Sort

```
1  def selection_sort(vet: [int]) -> None:
2      for i in range(len(vet)-1):
3          imin = indice_menor(vet, i)
4          aux = vet[i]
5          vet[i] = vet[imin]
6          vet[imin] = aux
```

Pratique!

Exercício 1

Faça um algoritmo/programa que determine se uma string é palíndromo.

Exercício 2

Faça um algoritmo/programa que determine se um vetor é permutação dos 10 primeiros números inteiros.

Exercício 3

Faça um algoritmo/programa que recebe um vetor ordenado e um elemento e faz a inserção do elemento no vetor.