

Processando a Informação: um livro prático de programação independente de linguagem

Rogério Perino de Oliveira Neves

Francisco de Assis Zampirolli

EDUFABC

editora.ufabc.edu.br

Notas de Aulas inspiradas no livro

Utilizando a(s) Linguagem(ns) de Programação:

C

Exemplos adaptados para Correção Automática no Moodle+VPL

Francisco de Assis Zampirolli

3 de setembro de 2022

Sumário

0.1	Processando a Informação: Cap. 5: Vetores - Prática 2	2
0.1.1	Exercícios	2
0.2	Guia de formatação com f-string	3
0.2.1	Alinhamento	3

0.1 Processando a Informação: Cap. 5: Vetores - Prática 2



Este caderno (Notebook) é parte complementar *online* do livro **Processando a Informação: um livro prático de programação independente de linguagem**, que deve ser consultado no caso de dúvidas sobre os temas apresentados.

Este conteúdo pode ser copiado e alterado livremente e foi inspirado nesse livro.

0.1.1 Exercícios

1. Criar um vetor de entrada com n posições com valores inteiros positivos e como saída criar um outro vetor também com n posições, onde a cada posição i seja atribuído a cálculo do mínimo do seu vizinho de $v1$ à esquerda $i-1$, do próprio elemento i e do seu vizinho à direita $i+1$.

```
[ ]: # escreva o seu código
```

2. Criar um vetor com n posições com valores inteiros positivos e, como saída, criar um outro vetor também com n posições, onde em cada posição i seja atribuído a cálculo dos mínimos dos seus vizinhos de $v1$ à esquerda $i-2$ e $i-1$, do próprio elemento i e dos seus vizinhos à direita $i+1$ e $i+2$. Generalize este código para os m vizinhos à esquerda e à direita.

```
[ ]: # escreva o seu código
```

-
3. O MMC (Mínimo Múltiplo Comum) de dois ou mais números inteiros é o menor múltiplo inteiro positivo comum a todos eles. Fazer uma função chamada MMC que recebe um vetor de números inteiros e retorna o MMC de todos. Veja um exemplo abaixo para calcular o MMC de 12 e 15:

a	b	/
12	15	2
6	15	2
3	15	3
1	5	5
1	1	60

$$MMC = 60 = 2 * 2 * 3 * 5$$

```
[ ]: # escreva o seu código
```

4. Criar um vetor de inteiros com n elementos. Inverter este vetor sem usar vetor auxiliar.

```
[ ]: # escreva o seu código
```

5. Criar dois vetores de inteiros com n elementos cada. Calcular o produto escalar entre eles.

```
[ ]: # escreva o seu código
```

0.2 Guia de formatação com f-string

[Ref](#)

0.2.1 Alinhamento

< à esquerda

> à direita

= zeros à esquerda

^ centralizar

```
[37]: x = 4.5
```

```
[55]: print(f'This will print out the variable x: {x:12}')
      print(f'This will print out the variable x: {x:>12}')
      print(f'This will print out the variable x: {x:>14.3f}')
```

```
print(f'This will print out the variable x: {x:=014.3f}')
```

```
print(f'\n{"="*50}')
```

```
print(f'{"My List":~50s}')
```

```
print(f'{"="*50}')
```

```
[62]: table = ['Sjoerd', 'Jack', 'Dcab']
      for name in table:
          print(f'{name:>20}')
```

```
print()
```

```
table2 = [4127, 4098, 7678]
for num in table2:
    print(f'{num:10}')
```

```
[10]: print(f'Number\tSquare\tCube')
```

```
for x in range(1, 11):
    print(f'{x:2d} \t{x*x:3d} \t{x*x*x:4d}')
```

```
[25]: print(f'Number\tSquare\t\tCube')
```

```
for x in range(1, 11):
    x = float(x)
    print(f'{x:5.2f}\t{x*x:6.2f}\t{x*x*x:12.2f}')
```

```
[31]: APPLES = .50
      BREAD = 1.50
      CHEESE = 2.25
      numApples = 3
      numBread = 4
      numCheese = 2
      prcApples = 3 * APPLES
      prcBread = 4 * BREAD
      prcCheese = 2 * CHEESE
      strApples = 'Apples'
      strBread = 'Bread'
      strCheese = 'Cheese'
```

```
total = prcBread + prcBread + prcApples
print(f'{"My Grocery List":~31s}')
```

```
print(f'{"="*31}')
```

```
print(f'{strApples}\t{numApples:10d}\tR${prcApples:>5.2f}')
```

```
print(f'{strBread}\t{numBread:10d}\tR${prcBread:>5.2f}')
```

```
print(f'{strCheese}\t{numCheese:10d}\tR${prcCheese:>5.2f}')
```

```
print(f'{"Total":>19s}\tR${total:>5.2f}')
```

```
[33]: number = 1000000
      print(f'The number, 1000000, formatted with a comma{number:,.2f}')
```

```
print(f'The number, 1000000, formatted with a comma and right-aligned,␣  
↳in a width of 15 {number:>15,.2f}')
```