Fundamentos de Algoritmos
Memória e passagem de parâmetros

Marco A L Barbosa malbarbo.pro.br

Lembre-se de seguir o processo de projeto de funções e de usar o `mypy` e o `doctest` na etapa de verificação.

Começando

1) O que significa dizer que as variáveis em Python são referências?
2) O que são apelidos?
3) Para que usamos apelidos?
4) De que maneira a especificação das funções que alteram os seus argumentos é diferente das funções que não alteram os argumentos?
5) Quais são os passos para implementar funções que alteram os seus argumentos?
6) O que é exibido na tela após a execução do código abaixo (confira a sua resposta usando o Python Tutor).

```
x = 60
lst = [x, x, x]
x = 30
lst[0] = 20
print(lst)
```

7) O que é exibido na tela após a execução do código abaixo (confira a sua resposta usando o Python Tutor).

```
x = [60]
lst = [x, x, x]
x.append(30)
lst[0] = [20]
print(lst)
```

8) O que é exibido na tela após a execução do código abaixo (confira a sua resposta usando o Python Tutor).

```
def troca(a: int, b: int):
    t = a
    a = b
    b = t

x = 10
y = 20
troca(x, y)
print(x, y)
```

9) O que é exibido na tela após a execução do código abaixo (confira a sua resposta usando o Python Tutor).

```python
def troca(lst: list[int], a: int, b: int):
    t = lst[a]
    lst[a] = lst[b]
    lst[b] = t

lst = [4, 8, 5, 6, 3]
troca(lst, 3, 1)
print(lst)
```

Praticando

10) Projete uma função que modifique uma lista de números somando um valor `n` a cada um dos seus elementos.

11) Projete uma função que modifique uma lista de strings deixando todas com o mesmo tamanho. Adiciona espaços em branco ao final das strings se necessário.

12) Projete uma função que receba duas listas como parâmetro e modifique a primeira lista adicionando todos os elementos da segunda lista no final da primeira.

13) Projete uma função que receba como parâmetros uma lista, um índice `i` e um valor `v`, e modifique a lista inserindo o valor `v` no índice `i`. Dica: veja o exemplo `insere_ordenado`.

14) Projete uma função que receba como parâmetros uma lista e um índice `i` e modifique a lista removendo o elemento do índice `i`.

```
>>> lst = [7, 1, 8, 9]
>>> remove_indice(lst, 2)
>>> lst
[7, 1, 9]
# Escreva mais exemplos!
```

Dica: mova o elemento do índice `i` até o final e depois use `list.pop` para removê-lo. A função `list.pop` funciona da seguinte forma

```
>>> lst = [3, 9, 1, 2]
>>> lst.pop()
2
>>> lst
[3, 9, 1]
```

15) Projete uma função que modifique uma lista removendo todos os elementos que estão em índices pares.

16) Projete uma função que modifique uma lista de strings removendo todas as strings vazias.

17) Projete uma função que modifique uma lista colocando os valores menores ou iguais a zero antes dos valores maiores que zero.

18) Ordenação por seleção é um algoritmo para ordenar uma lista de valores. A ideia do algoritmo é selecionar um valor mínimo da lista a partir da posição 0 e colocá-lo na posição 0, depois encontrar um valor mínimo da lista a partir da posição 1 e colocá-lo na posição 1, depois encontrar um valor mínimo da lista a partir da posição 2 … e assim por diante. Por exemplo, vamos considerar a lista `[8, 5, 4, 1, 2]`.

    O valor mínimo a partir da posição 0 é 1 (que está no índice 3), colocando 1 na posição 0, obtemos `[1, 5, 4, 8, 2]`.

    O valor mínimo a partir da posição 1 é 2 (que está no índice 4), colocando 2 na posição 1, obtemos `[1, 2, 4, 8, 5]`.

    O valor mínimo a partir da posição 2 é 4 (que está no índice 2), colocando 4 na posição 2, obtemos `[1, 2, 4, 8, 5]`.

O valor mínimo a partir da posição 3 é 5 (que está no índice 4), colocando 5 na posição 3, obtemos `[1, 2, 4, 5, 8]`.

Baseado nesta descrição, projete uma função que faça a ordenação dos valores usando o algoritmo de ordenação por seleção.