27 Reunião 27: 01/DEZ/2020

Figure 1: Bússola de ouro, copiado daqui

27.1 Reuniões passadas

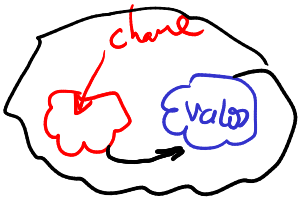


Figure 2: Desenho copiado daqui

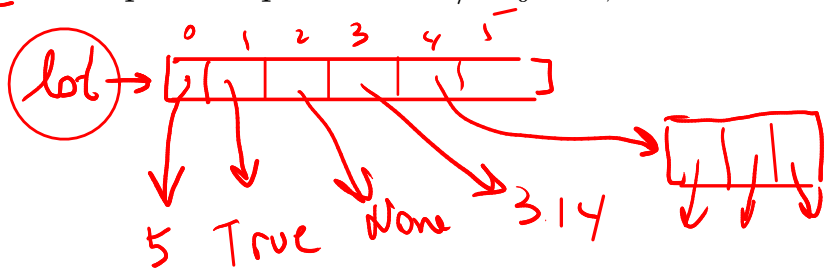
Um **dicionário** (tipo `dict`) é um conjunto de objetos ou itens cada um dotado de uma `chave` e de um `valor`.

27.2 Hoje

Começaremos a conversar sobre como representar matriz (=tabelas bidimensionais em Python). Hoje, nosso principal foco será em **percorrer matrizes** e nos habituarmos com seus índices e suas posições.

Lembrar:

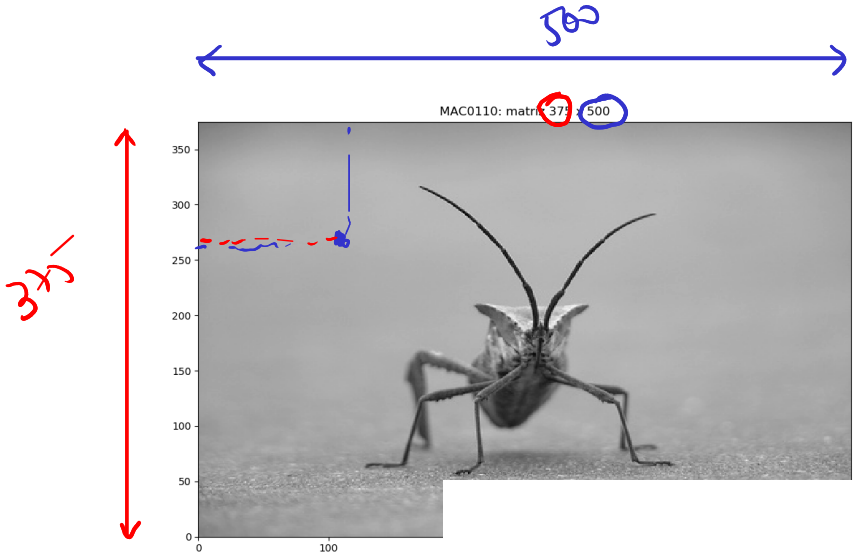
- atribuições não cria nada, apenas um apelido;
- posições de listas e de matrizes são apelidos para coisas/objetos;

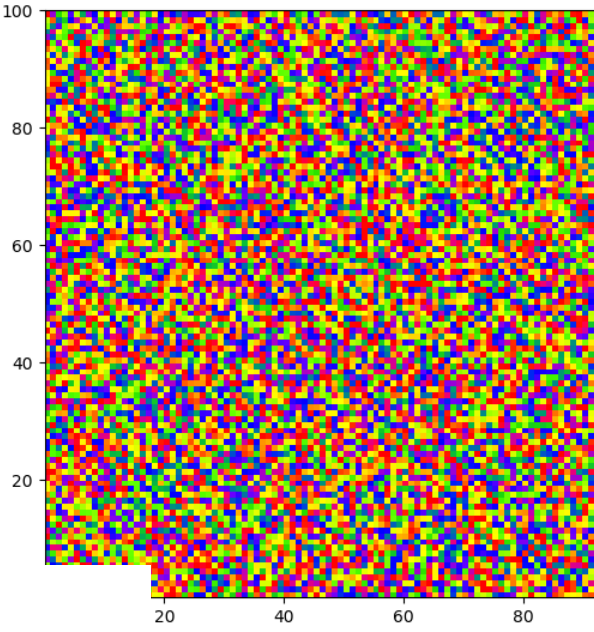


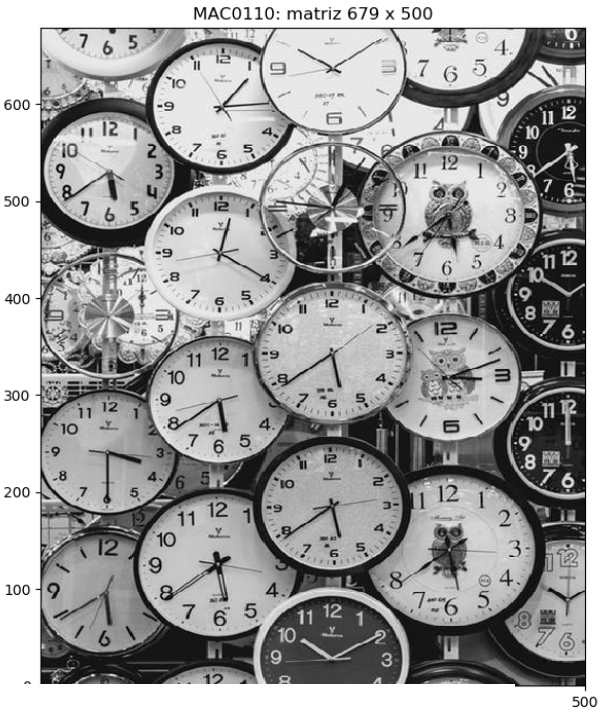
27.3 Matrizes

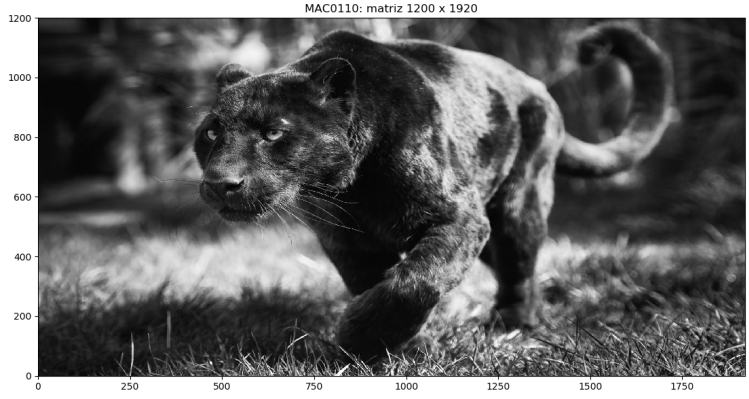
Figure 3: *Matrix*: matriz 522 x 700

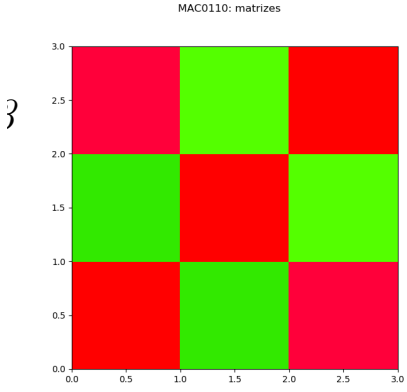
Matrizes são estruturas bidimensionais (tabelas) com `nlins` linhas por `ncols` colunas muito importantes na matemática, utilizadas por exemplo para a resolução de sistemas de equações e transformações lineares e representar imagens.


MAC0110: matriz 375 x 500




MAC0110: matriz 679 x 500


MAC0110: matriz 1200 x 1920

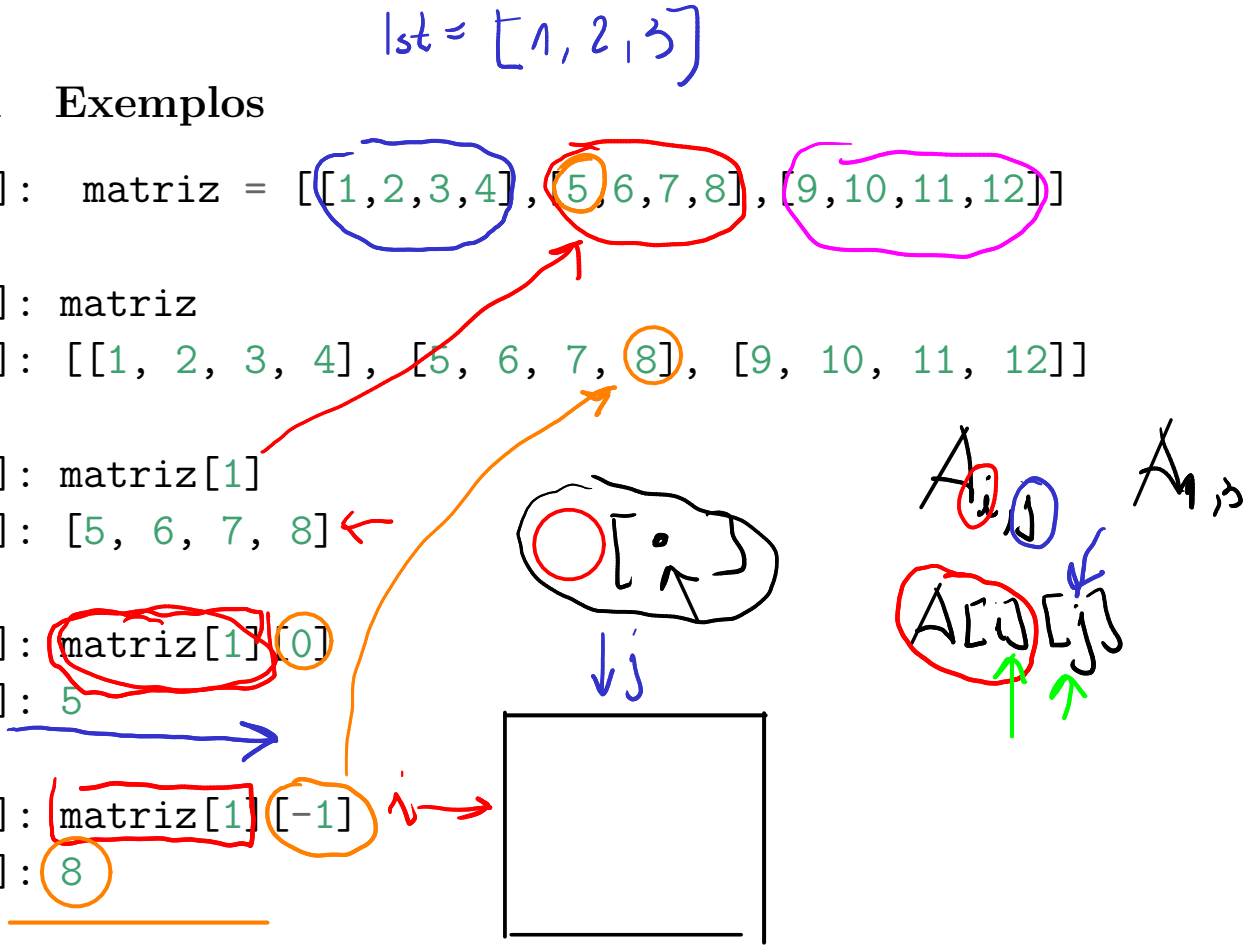

MAC0110: matrizes

27.4 Matrizes em Python

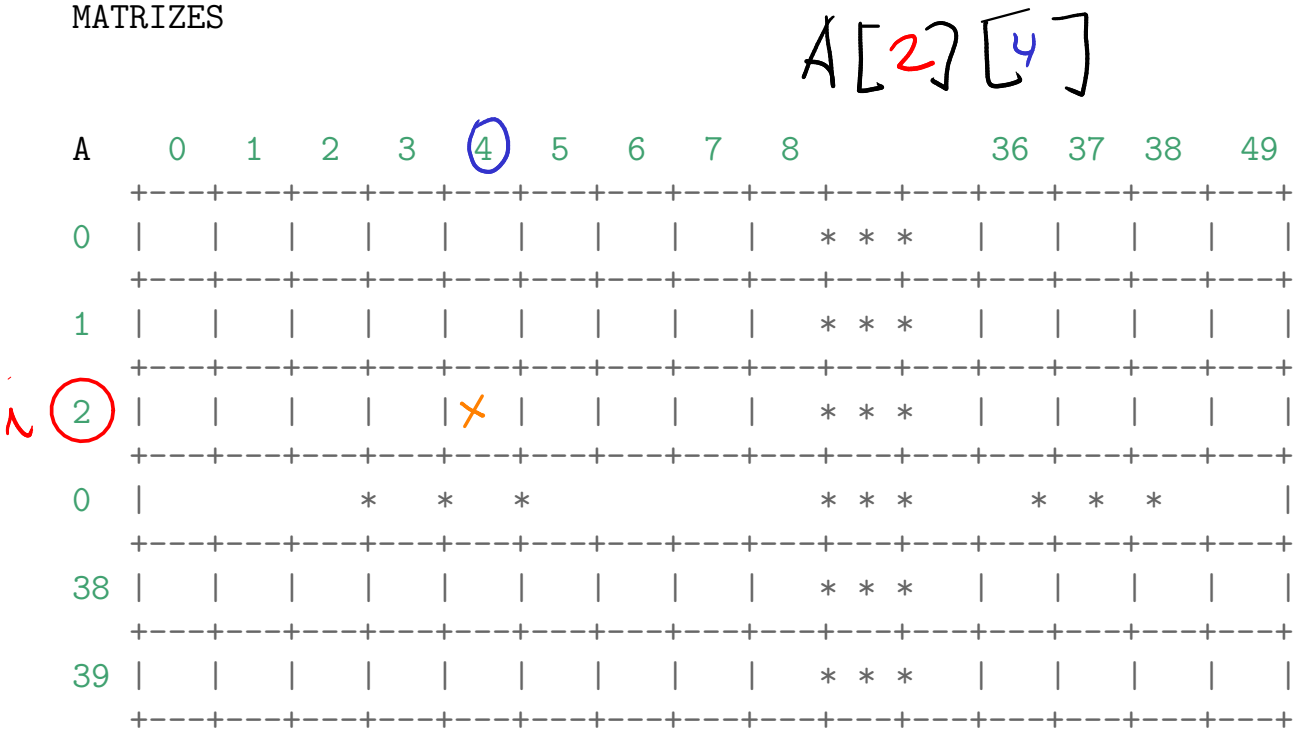
Em Python, uma matriz pode ser representada como uma lista de listas, onde elemento da lista contém uma linha da matriz, que por sua vez corresponde a uma lista com os elementos da coluna da matriz.

27.4.1 Exemplos

```
In [1]:  matriz = [[1,2,3,4],[5,6,7,8],[9,10,11,12]]

In [2]: matriz
Out[2]: [[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]]

In [3]: matriz[1]
Out[3]: [5, 6, 7, 8]

In [4]: matriz[1][0]
Out[4]: 5

In [5]: matriz[1][-1]
Out[5]: 8

In [6]: matriz[1][5]
---------------------------------------------------------------------------
IndexError                                Traceback (most recent
<ipython-input-6-fa3662a2b8eb> in <module>
----> 1 matriz[1][5]

IndexError: list ind
```

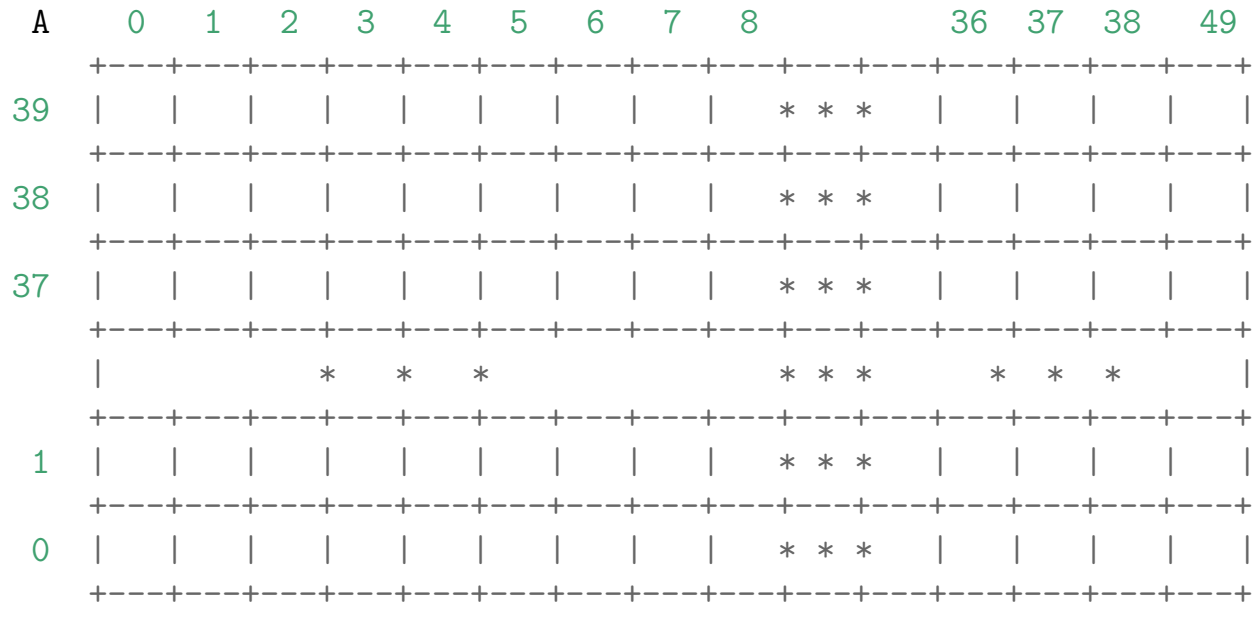


MATRIZES



nlins = 40
ncols = 50

ALGUMAS IMAGENS



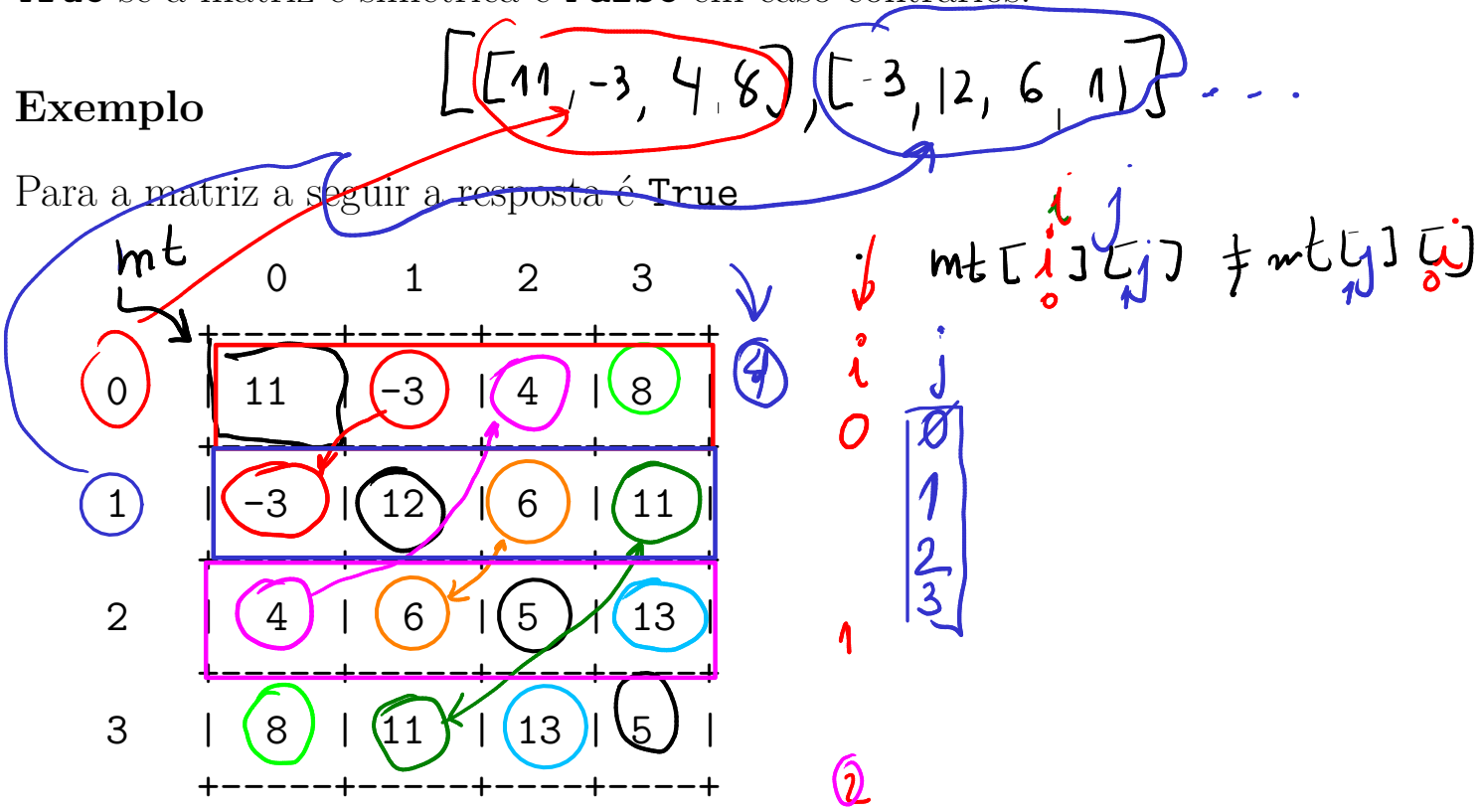
nlins = 40
ncols = 50

27.5 Exercício: matrizes simétricas

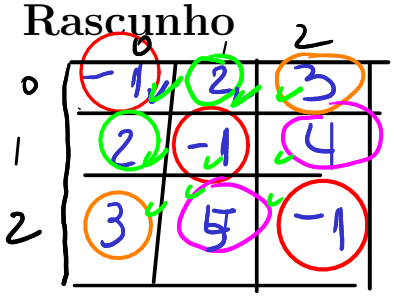
Escreva uma função `simetrica()` que **recebe** um matriz *quadrada* e **retorna** `True` se a matriz é simétrica e `False` em caso contrários.

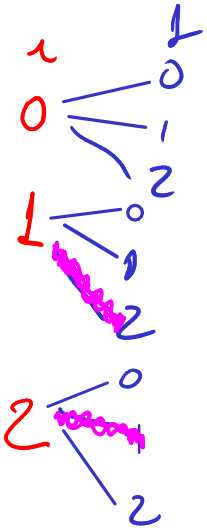
Exemplo

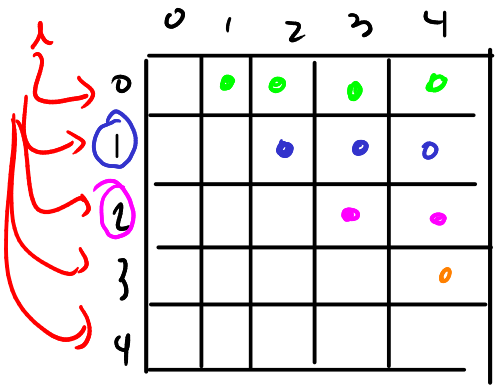
Para a matriz a seguir a resposta é `True`

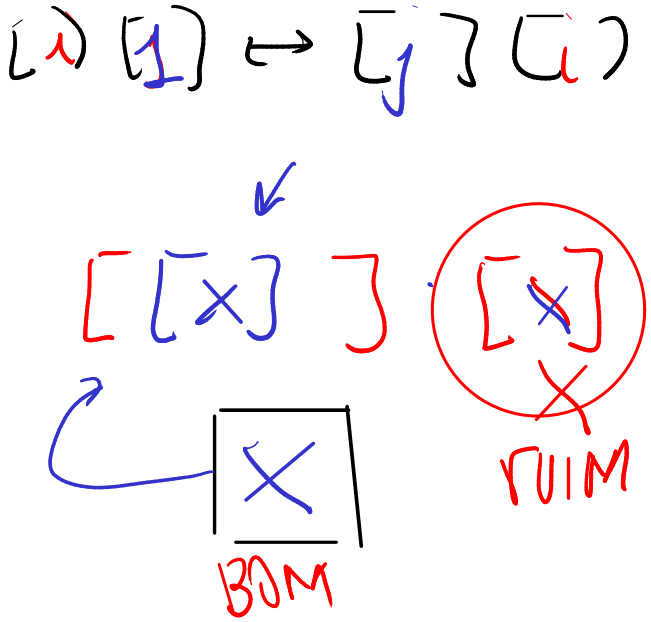


Rascunho


0
1
2
0
1
2
-1
2
3
2
-1
4
3
5
-1


i
j
0
1
2
0
1
2
0
1
2
0
1
2


i
0
1
2
3
4
0
1
2
3
4


[i] [j] ↔ [j] [i]
[[x]]
[x]
RUIM
x
BOM

Solução

```
def simetrica(mt):
    '''(matriz) -> bool

    RECEBE uma matriz quadrada representada por lista de
           listas (list[list]).
    RETORNA True se a matriz for simétrica, em caso
           contrário retorna False.

    Pré-condição: a função supões que a matriz e quadrada

    In [10]: a = [[1,2,3],[2,1,4],[3,4,1]]
    In [11]: a
    Out[11]: [[1, 2, 3], [2, 1, 4], [3, 4, 1]]
    In [12]: simetrica(a)
    Out[12]: True
    '''
    n = len(mt)
    e_simetrica = True

    i = 0
    while i < n and e_simetrica:
        j = 0
        while j < i and e_simetrica:
            if mt[i][j] != mt[j][i]:
                e_simetrica = False
                print("Posições mt[{i}][{j}] = {mt[i][j]} != {mt[
            j = j + 1
        i = i + 1

    return e_simetrica
```

27.6 Exercício: linhas com `val`

Escreva uma função `linhas_val()` que **recebe** um matriz `mt` e um valor `val` e **retorna** o número de linhas em que todos os valores são iguais a `val`

Exemplo

Para a matriz e seguir e `val = 11` a resposta é `2` e para `val = 5` a resposta é `0`.

	0	1	2	3	4
0	11	11	11	11	11
1	-3	12	6	11	34
2	11	11	11	11	11
3	8	11	13	5	7

Solução

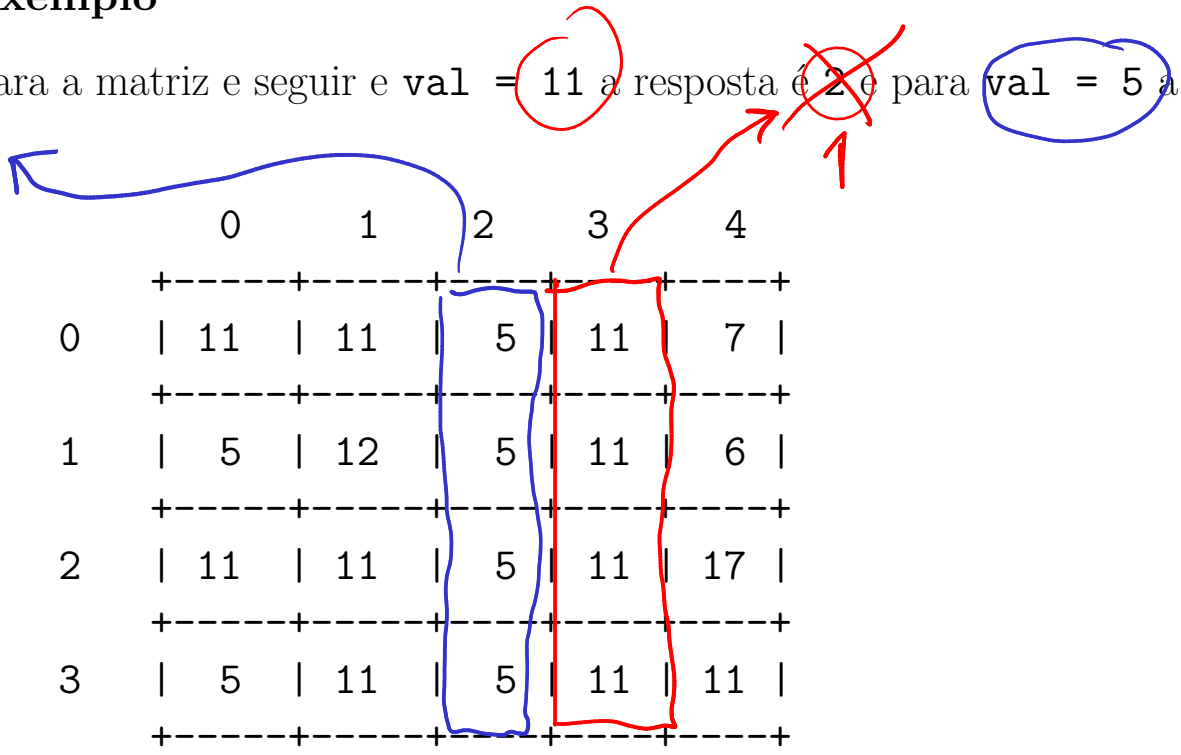
```
#-----------------------------------------------------------
def linhas_val(mt, val):
    '''(matriz, obj) -> int

    RECEBE uma matriz `mat` representada por lista de
           listas (list[list]) e um valor `val`.
    RETORNA o número de linhas da m`mt` em que todos os valores
           são iguais a `val`.

    In [10]: a = [[1,1,1],[2,1,4],[1,1,1]]
    In [11]: a
    [[1, 1, 1], [2, 1, 4], [1, 1, 1]]
    In [12]: linhas_val(a, 1)
    Out[12]: 2
    '''
    cont = 0
    nlins = len(mt)
    ncols = len(mt[0])

    for i in range(nlins):
        iguais = True
        for j in range(ncols):
            if mt[i][j] != val:
                iguais = False
        if iguais:
            cont += 1

    return cont
```

27.7 Exercício: colunas com `val`

Escreva uma função `colunas_val()` que **recebe** um matriz `mt` e um valor `val` e **retorna** o número de colunas em que todos os valores são iguais a `val`

Exemplo

Para a matriz e seguir e `val = 11` a resposta é ~~2~~ 1 e para `val = 5` a resposta é 1.



	0	1	2	3	4
0	11	11	5	11	7
1	5	12	5	11	6
2	11	11	5	11	17
3	5	11	5	11	11

Solução

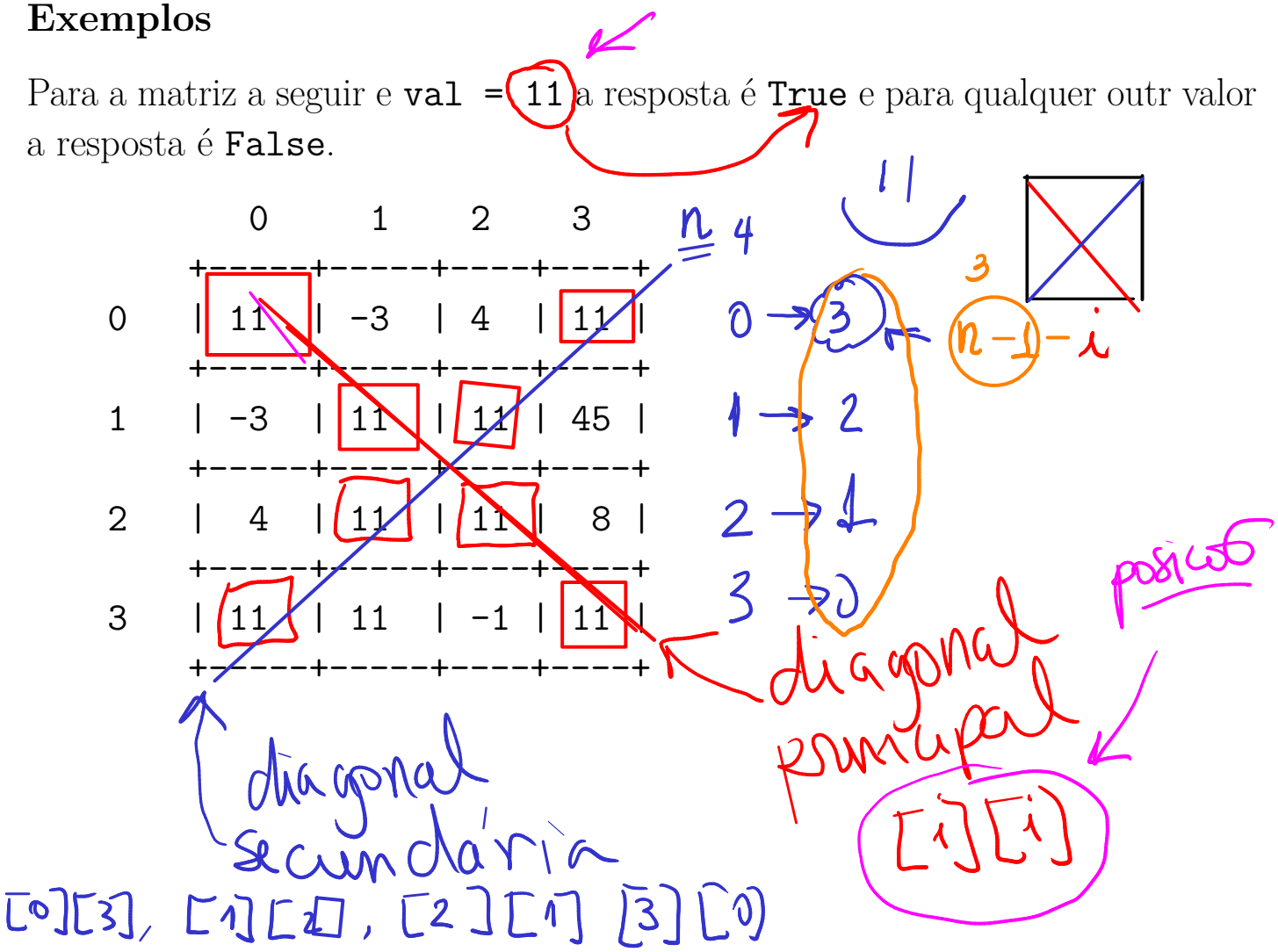
```
def colunas_val(mt, val):
    '''(matriz, obj) -> int

    RECEBE uma matriz `mat` representada por lista de
           listas (list[list]) e um valor `val`.
    RETORNA o número de colunas de `mt` em que todos os valores
           são iguais a `val`.

    In [10]: a = [[1,1,1],[2,1,4],[1,1,1]]
    In [11]: a
    [[1, 1, 1], [2, 1, 4], [1, 1, 1]]
    In [12]: colunas_val(a, 1)
    Out[12]: 1
    '''
    cont = 0
    nlins = len(mt)
    ncols = len(mt[0])

    for j in range(ncols):
        iguais = True
        for i in range(nlins):
            if mt[i][j] != val:
                iguais = False
        if iguais:
            cont += 1

    return cont
```

27.8 Exercício: diagonais com `val`

Escreva uma função `diagonais_val()` que **recebe** um matriz *quadrada* `mt` e um valor `val` e **retorna** `True` se a matriz é a diagonal principal e secundária de `mt` tem todos seus valores iguais a `val`. Em caso contrário a função **retorna** `False`.

Exemplos

Para a matriz a seguir e `val = 11` a resposta é `True` e para qualquer outr valor a resposta é `False`.



Solução

```
def diagonais_val(mt, val):
    '''(matriz, obj) -> int

    RECEBE uma matriz quadrada `mt` representada por lista de
           listas (list[list]) e um valor `val`.
    RETORNA True se todos os valores da diagonal principal e sec
           de `mt` são iguais a `val`. Em caso contrário a funçã
           False.

    In [10]: a = [[1,1,1],[2,1,4],[1,1,1]]
    In [11]: a
    [[1, 1, 1], [2, 1, 4], [1, 1, 1]]
    In [12]: diagonais_val(a, 1)
    Out[12]: True
    '''
    iguais = True
    nlins = len(mt)

    for i in range(nlins):
        if mt[i][i] != val or mt[nlins-1-i][i] != val:
            iguais = False

    return iguais
```