Rafael Vieira Coelho

rafaelvc2@gmail.com

Links Úteis

- **www.python.org** é a página oficial da linguagem Python. Ela contém documentação, links para outros sites e listas de discussão nas quais você pode participar.

- **http://python.org.br** é o site da comunidade Python Brasil. Ele reune grupos de usuários em todo o Brasil interessados em difundir e divulgar a linguagem de programação.

- http://www.thinkpython.com é uma página de um livro.

- http://wiki.python.org.br/IdesPython fala sobre as melhores IDEs.

Por último, se você for ao Google e buscar por “python -snake -monty”, você encontrará cerca de 750 mil resultados.

python™

Conteúdo Programático

PARTE 1

1. Sobre a Linguagem
2. Variáveis e Expressões
3. Funções
4. Condicionais
5. Funções frutíferas
6. Strings
7. Listas
8. Tuplas
9. Arquivos e exceções
10. Depuração
11. **Matrizes**
12. Busca
13. Ordenação

python™

11 - Matrizes

PARTE 1

Matrizes

Tópicos

- Concatenação de listas usando `+` e `*`
- Mais fatias de listas
- Matrizes
- Funções com matrizes

Revisão de Listas

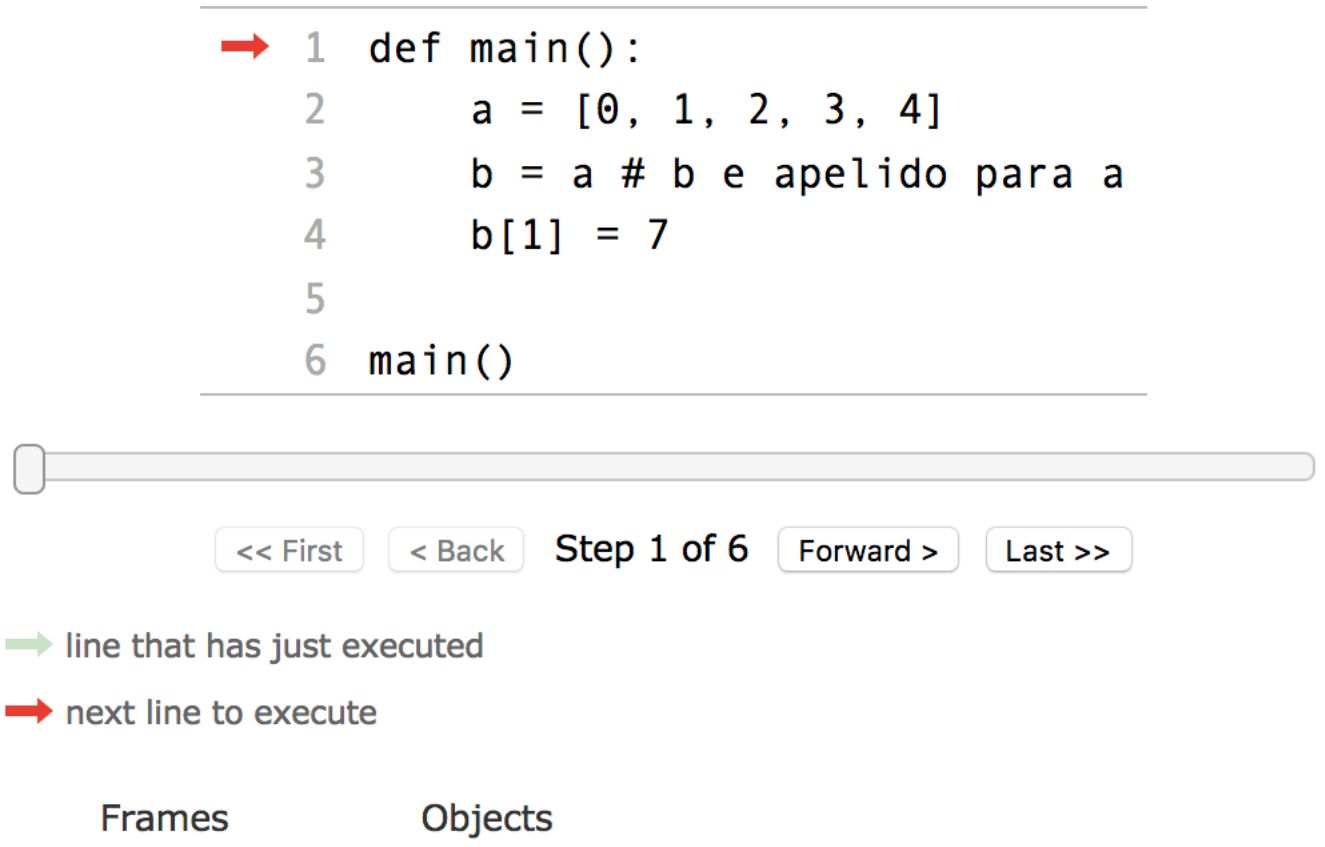
O que é impresso pelo seguinte código?

```
1  def main():
2      a = [0, 1, 2, 3, 4]
3      b = a # b e apelido para a
4      b[1] = 7
5      print("a = ", a)
6      print("b = ", b)
7
8  main()
9
10
```

```
a =  [0, 7, 2, 3, 4]
b =  [0, 7, 2, 3, 4]
```

python™

Exemplo: passo a passo



python™

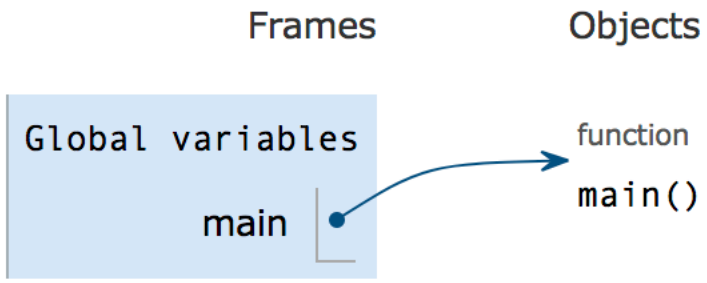
Exemplo: passo a passo

```
1  def main():
2      a = [0, 1, 2, 3, 4]
3      b = a # b e apelido para a
4      b[1] = 7
5
6  main()
```

<< First | < Back | Step 2 of 6 | Forward > | Last >>

line that has just executed

next line to execute



python™

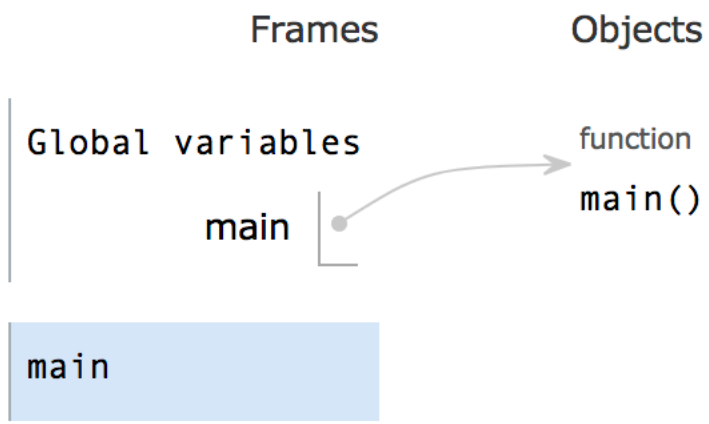
Exemplo: passo a passo

```
1  def main():
2      a = [0, 1, 2, 3, 4]
3      b = a # b e apelido para a
4      b[1] = 7
5
6  main()
```

<< First | < Back | Step 3 of 6 | Forward > | Last >>

line that has just executed

next line to execute



python™

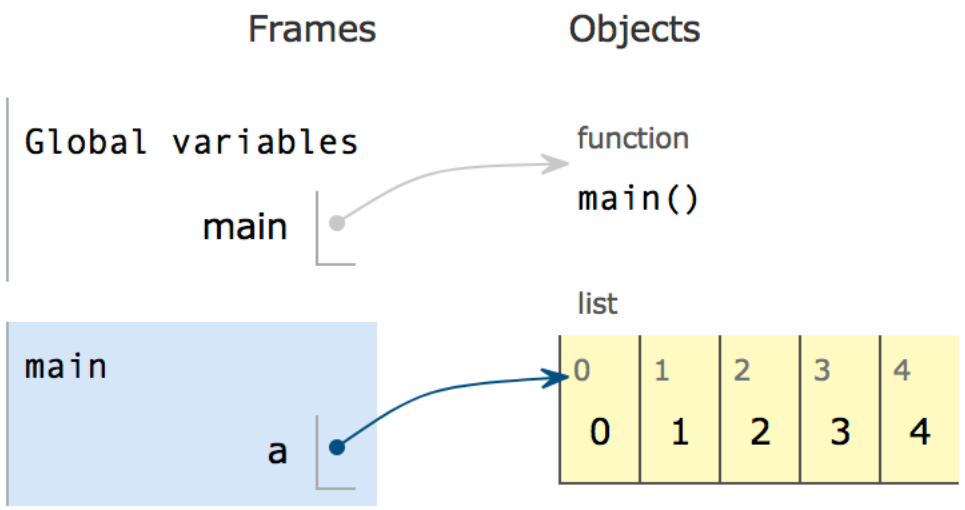
Exemplo:

```
1 def main():
2     a = [0, 1, 2, 3, 4]
3     b = a # b e apelido para a
4     b[1] = 7
5
6 main()
```

<< First | < Back | Step 4 of 6 | Forward > | Last >>

line that has just executed

next line to execute



python™

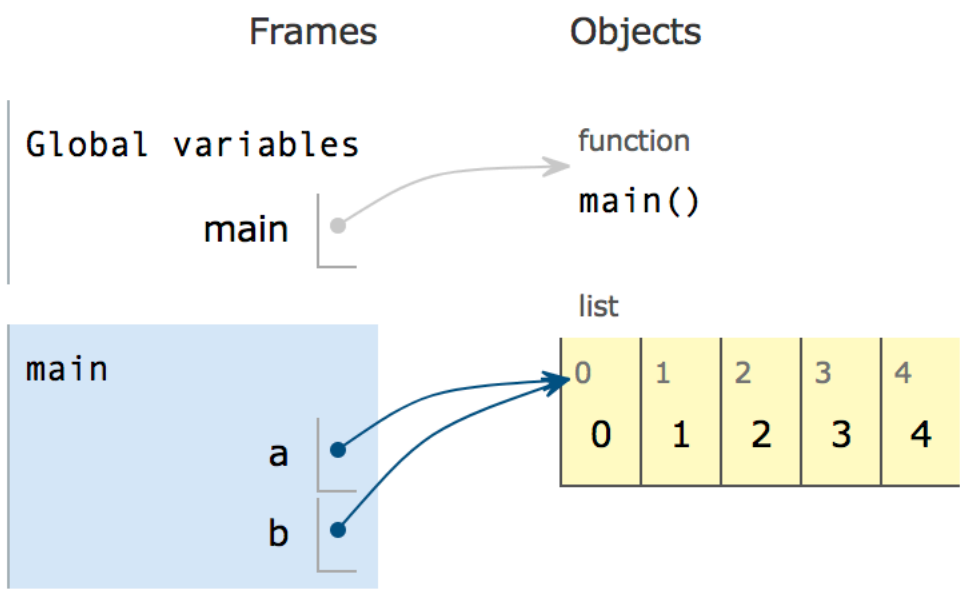
Exemplo:

```
1  def main():
2      a = [0, 1, 2, 3, 4]
3      b = a # b e apelido para a
4      b[1] = 7
5
6  main()
```

<< First | < Back | Step 5 of 6 | Forward > | Last >>

line that has just executed

next line to execute



python™

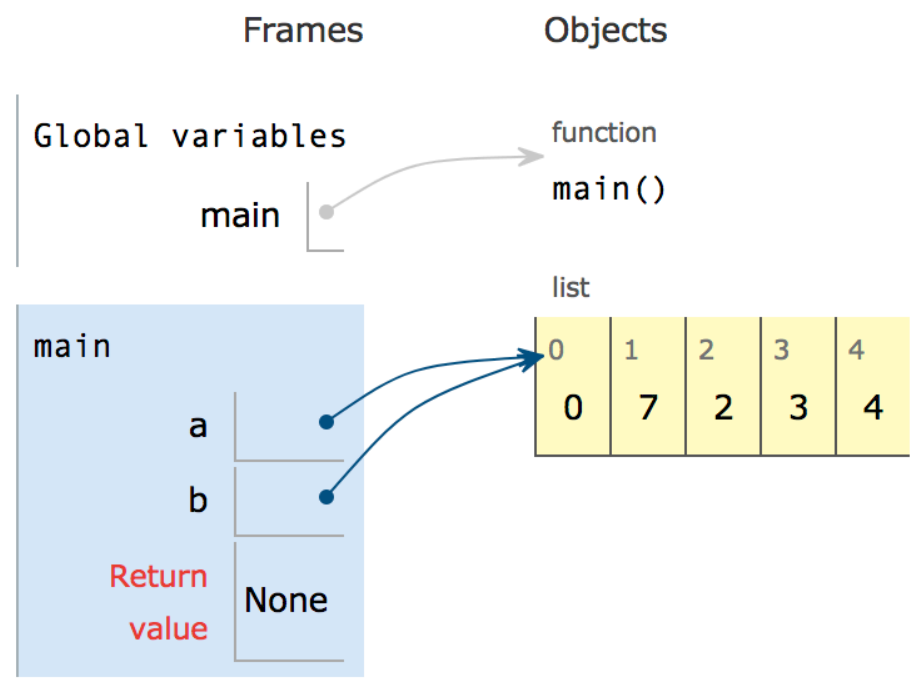
Exemplo:

```
1 def main():
2     a = [0, 1, 2, 3, 4]
3     b = a # b e apelido para a
4     b[1] = 7
5
6 main()
```

<< First | < Back | Step 6 of 6 | Forward > | Last >>

line that has just executed

next line to execute



python™

Cópia de uma Lista

No código acima `b` é uma referência ou apelido (= alias) para a mesma lista a que `a` está se referenciando.

O trecho de código a seguir cria um cópia (= clone) de `a` e `b` será uma referência a essa cópia.

```
1  def main():
2      a = [0, 1, 2, 3, 4]
3      b = a[:]
4      b[1] = 7
5      print("a = ", a)
6      print("b = ", b)
7
8  main()
9
10
```

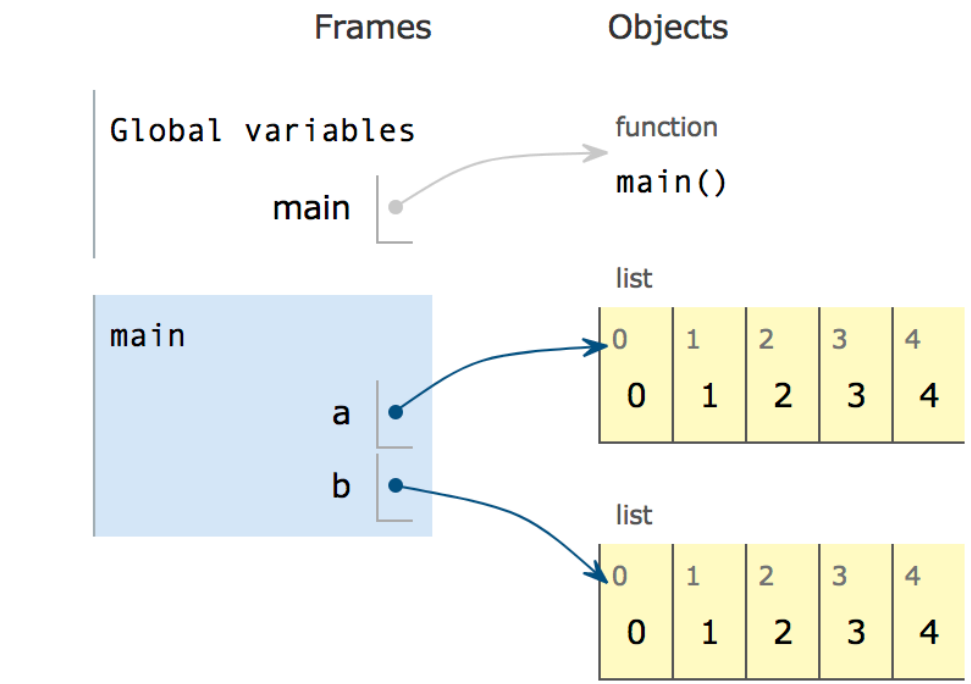
Cópia de uma Lista

```
1  def main():
2      a = [0, 1, 2, 3, 4]
3      b = a[:]
4      b[1] = 7
5
6  main()
```

<< First | < Back | Step 5 of 6 | Forward > | Last >>

line that has just executed

next line to execute



A fatia a[:] é uma cópia de a, uma fatia com todos os elementos de a. Podemos usar o : para definir qualquer fatia da lista.

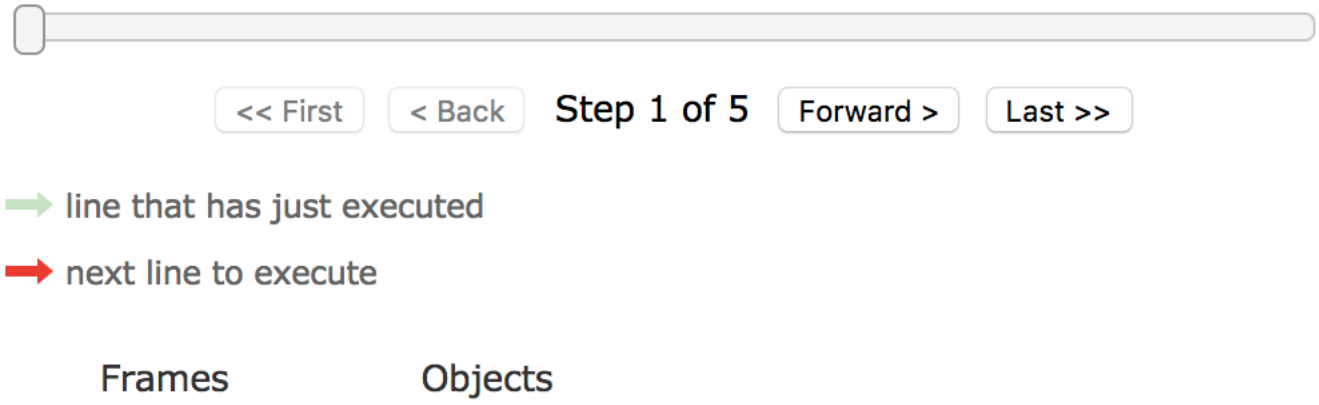
Por exemplo, a[1:3] produz a lista [1,2].

python™

Concatenação de Listas

Uma outra forma para manipular listas em Python é usando o operador de concatenação +. Veja o seguinte exemplo:

```
1  primos = []
2  primos = primos + [9]
3  primos = [2, 3] + primos
4  primos = primos[:2] + [5, 7] + primos[2:]
5  primos += [11]
```



python™

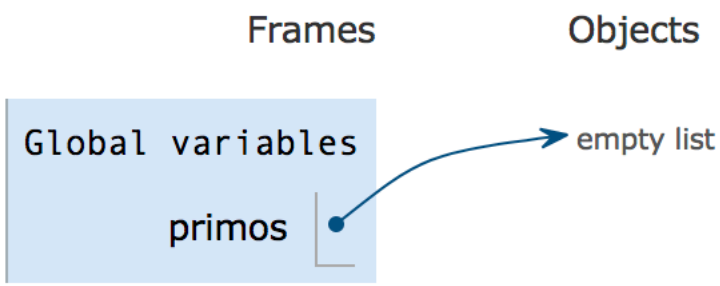
Concatenação de Listas

```
1  primos = []
2  primos = primos + [9]
3  primos = [2, 3] + primos
4  primos = primos[:2] + [5, 7] + primos[2:]
5  primos += [11]
```

<< First | < Back | Step 2 of 5 | Forward > | Last >>

line that has just executed

next line to execute



python™

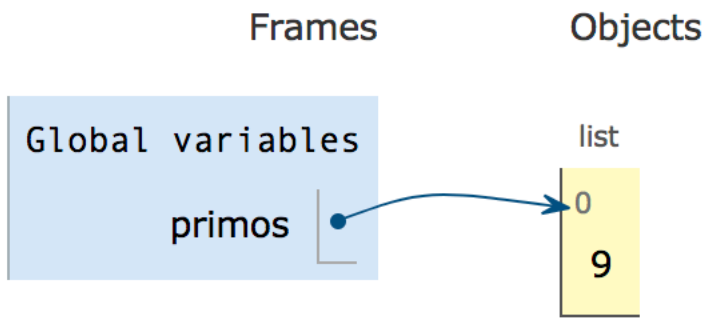
Concatenação de Listas

```
1  primos = []
2  primos = primos + [9]
3  primos = [2, 3] + primos
4  primos = primos[:2] + [5, 7] + primos[2:]
5  primos += [11]
```

<< First | < Back | Step 3 of 5 | Forward > | Last >>

line that has just executed

next line to execute



python™

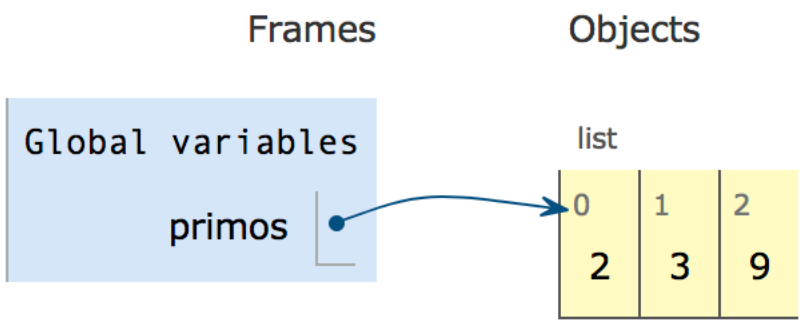
Concatenação de Listas

```
1  primos = []
2  primos = primos + [9]
3  primos = [2, 3] + primos
4  primos = primos[:2] + [5, 7] + primos[2:]
5  primos += [11]
```

<< First | < Back | Step 4 of 5 | Forward > | Last >>

line that has just executed

next line to execute



python™

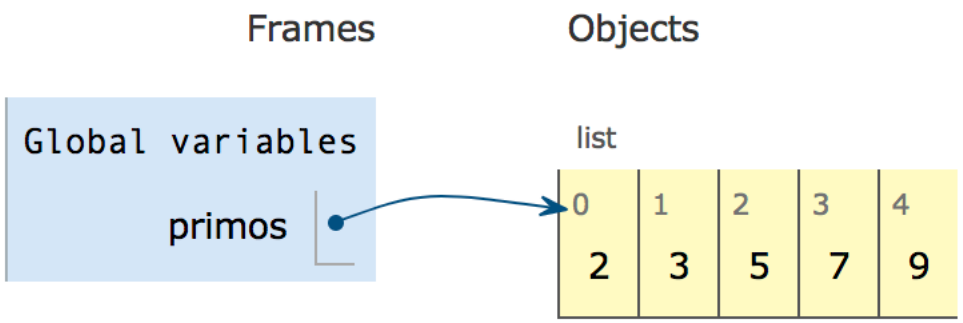
Concatenação de Listas

```
1  primos = []
2  primos = primos + [9]
3  primos = [2, 3] + primos
4  primos = primos[:2] + [5, 7] + primos[2:]
5  primos += [11]
```

<< First | < Back | Step 5 of 5 | Forward > | Last >>

line that has just executed

next line to execute



python™

Concatenação de Listas

Observe que a concatenação pode ser utilizada (junto com fatias) para inserir e/ou remover elementos em qualquer posição da lista. Porém, diferentemente do método append, a concatenação cria uma nova lista contendo o resultado.

Múltiplas concatenações de listas podem ser realizadas pelo operador *. Assim, para se criar uma lista com 5 zeros podemos escrever:

```
1  # concatenando 5 zeros usando +
2  cinco_zeros = [0] + [0] + [0] + [0] + [0]
3
4  # concatenando 5 zeros usando *
5  zeros = [0] * 5
6  print("cinco_zeros", cinco_zeros)
7  print("zeros: ", zeros)
```

<< First | < Back | Step 1 of 4 | Forward > | Last >>

python™

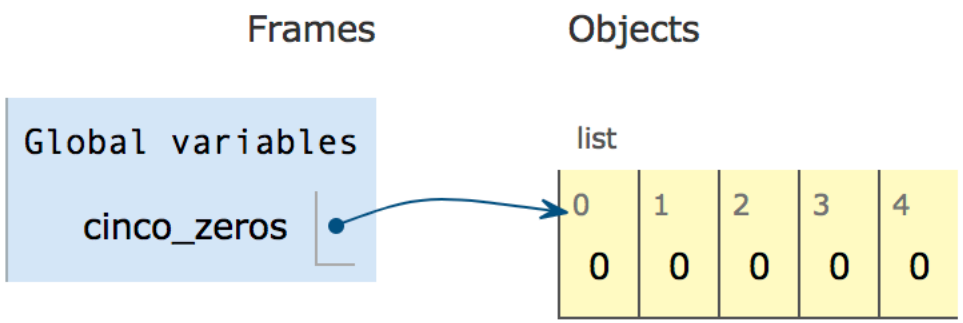
Concatenação de Listas

```
1  # concatenando 5 zeros usando +
2  cinco_zeros = [0] + [0] + [0] + [0] + [0]
3
4  # concatenando 5 zeros usando *
5  zeros = [0] * 5
6  print("cinco_zeros", cinco_zeros)
7  print("zeros: ", zeros)
```

<< First | < Back | Step 2 of 4 | Forward > | Last >>

line that has just executed

next line to execute



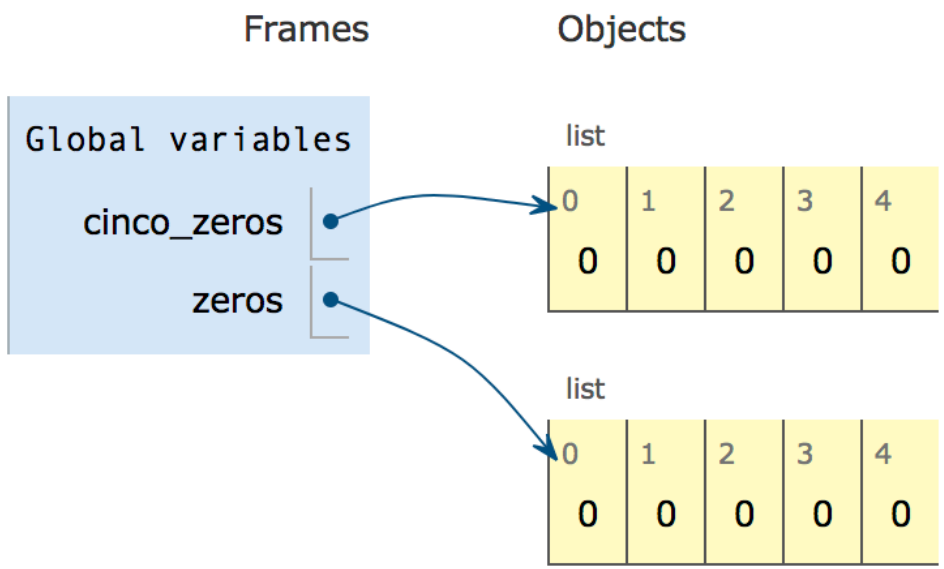
python™

```
1  # concatenando 5 zeros usando +
2  cinco_zeros = [0] + [0] + [0] + [0] + [0]
3
4  # concatenando 5 zeros usando *
5  zeros = [0] * 5
6  print("cinco_zeros", cinco_zeros)
7  print("zeros: ", zeros)
```

<< First | < Back | Step 3 of 4 | Forward > | Last >>

line that has just executed

next line to execute



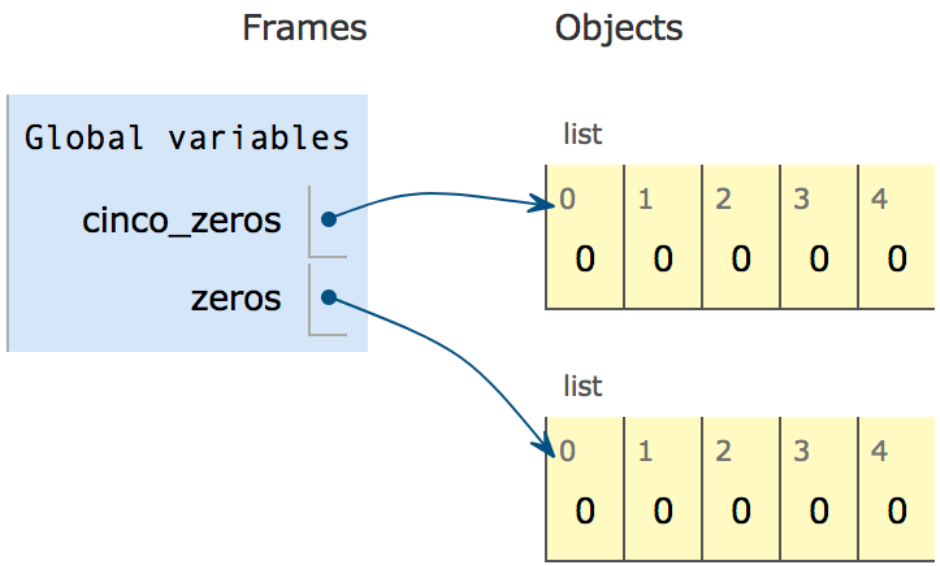
python™

```
1  # concatenando 5 zeros usando +
2  cinco_zeros = [0] + [0] + [0] + [0] + [0]
3
4  # concatenando 5 zeros usando *
5  zeros = [0] * 5
6  print("cinco_zeros", cinco_zeros)
7  print("zeros: ", zeros)
```

<< First | < Back | Step 4 of 4 | Forward > | Last >>

line that has just executed

next line to execute



python™

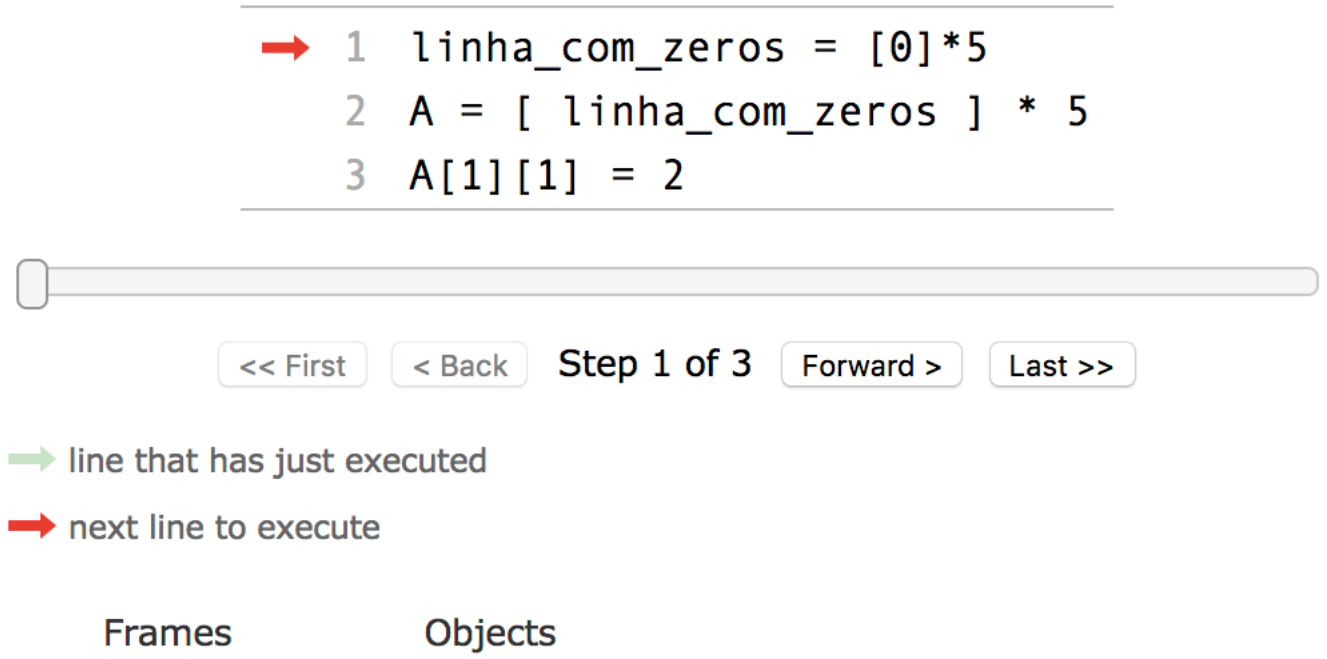
Criação de Matrizes

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

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

Criação de Matrizes

Qual o problema do seguinte pedaço de código para criação de uma matriz A com 5 linha e 5 colunas com o valor 2 na posição [1][1] e zero nas demais posições?



python™

Criação de Matrizes

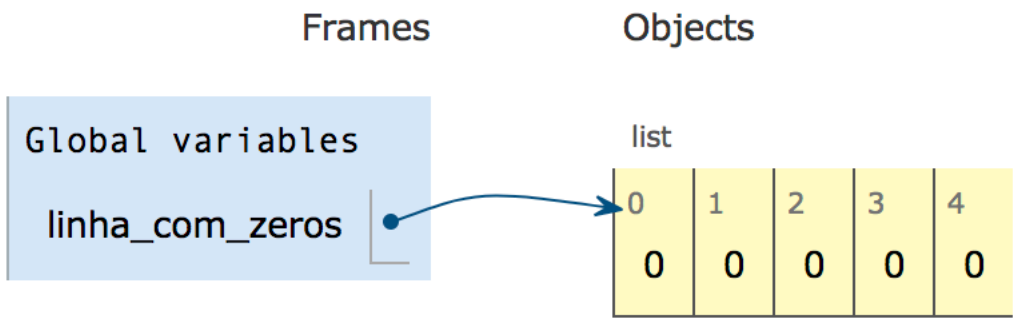
Qual o problema do seguinte pedaço de código para criação de uma matriz A com 5 linha e 5 colunas com o valor 2 na posição [1][1] e zero nas demais posições?

```
1  linha_com_zeros = [0]*5
2  A = [ linha_com_zeros ] * 5
3  A[1][1] = 2
```

<< First | < Back | Step 2 of 3 | Forward > | Last >>

line that has just executed

next line to execute



python™

Criação de Matrizes

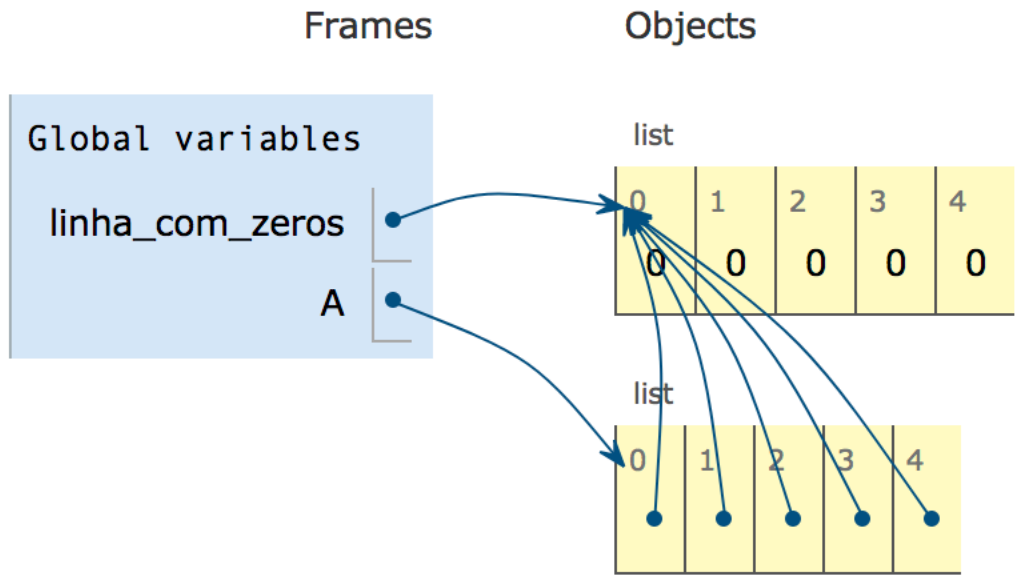
Qual o problema do seguinte pedaço de código para criação de uma matriz A com 5 linha e 5 colunas?

```
1  linha_com_zeros = [0]*5
2  A = [ linha_com_zeros ] * 5
3  A[1][1] = 2
```

<< First | < Back | Step 3 of 3 | Forward > | Last >>

line that has just executed

next line to execute



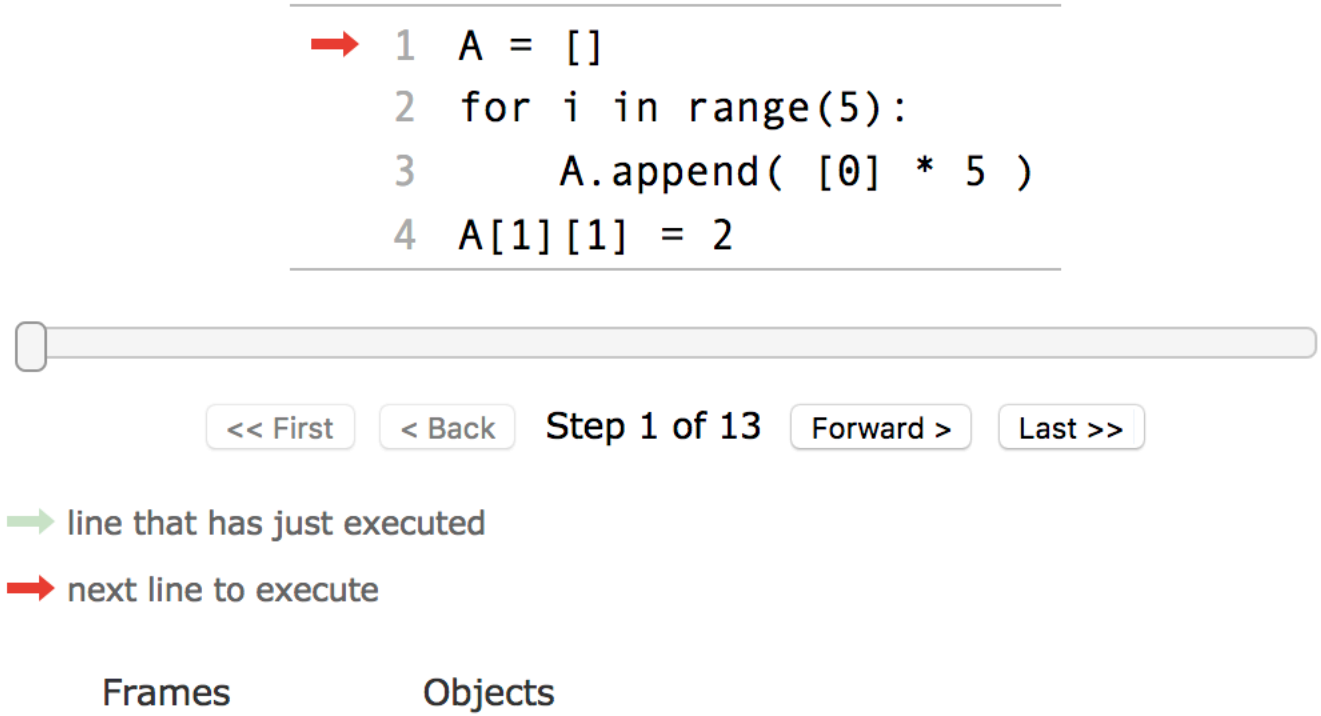
A variável linha_com_zeros contém uma referência à lista [0, 0, 0, 0, 0].

Na tentativa de cria uma matriz A, essa mesma referência é copiada 5 vezes.

python™

Criação de Matrizes

Para criarmos uma matriz é necessário criarmos 5 linhas diferentes como por exemplo:



python™

Criação de Matrizes

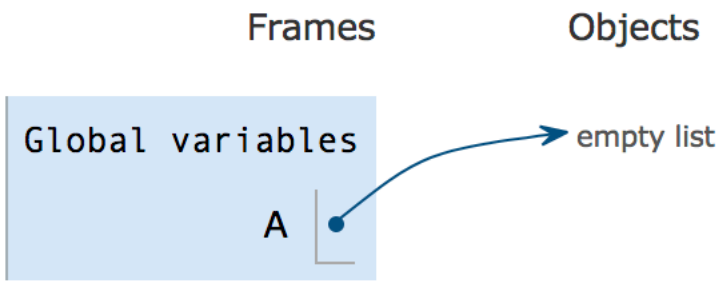
Para criarmos uma matriz é necessário criarmos 5 linhas diferentes como por exemplo:

```
1  A = []
2  for i in range(5):
3      A.append( [0] * 5 )
4  A[1][1] = 2
```

<< First | < Back | Step 2 of 13 | Forward > | Last >>

line that has just executed

next line to execute



python™

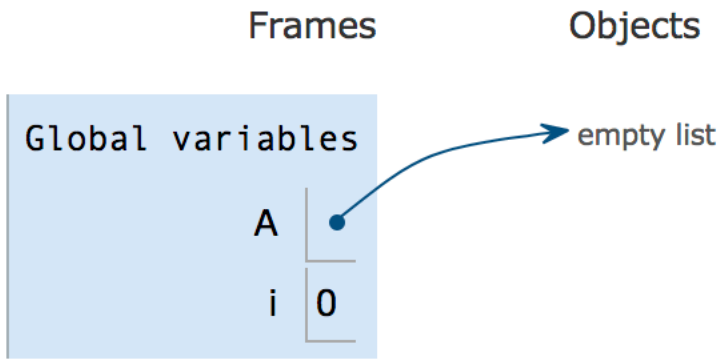
Criação de Matrizes

```
1  A = []
2  for i in range(5):
3      A.append( [0] * 5 )
4  A[1][1] = 2
```

<< First | < Back | Step 3 of 13 | Forward > | Last >>

line that has just executed

next line to execute



python™

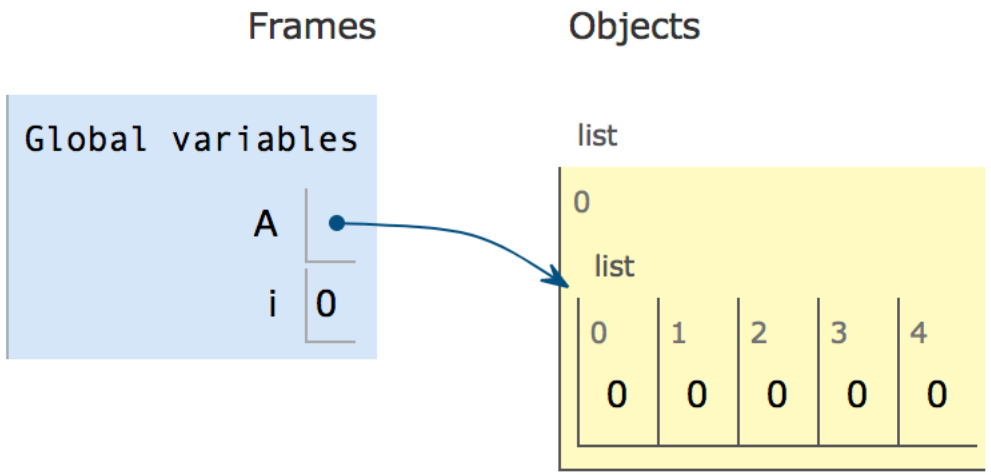
Criação de Matrizes

```
1  A = []
2  for i in range(5):
3      A.append( [0] * 5 )
4  A[1][1] = 2
```

<< First | < Back | Step 4 of 13 | Forward > | Last >>

line that has just executed

next line to execute



python™

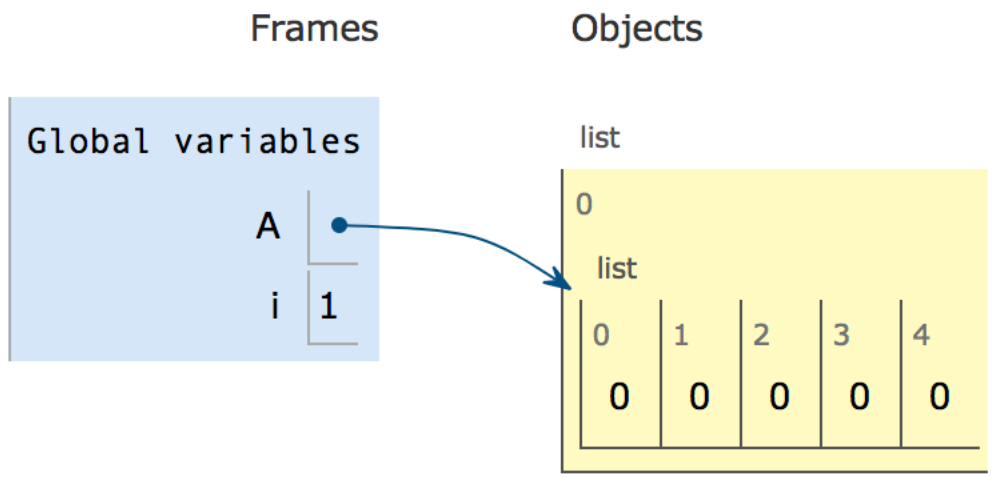
Criação de Matrizes

```
1 A = []
2 for i in range(5):
3     A.append( [0] * 5 )
4 A[1][1] = 2
```

<< First | < Back | Step 5 of 13 | Forward > | Last >>

line that has just executed

next line to execute



python™

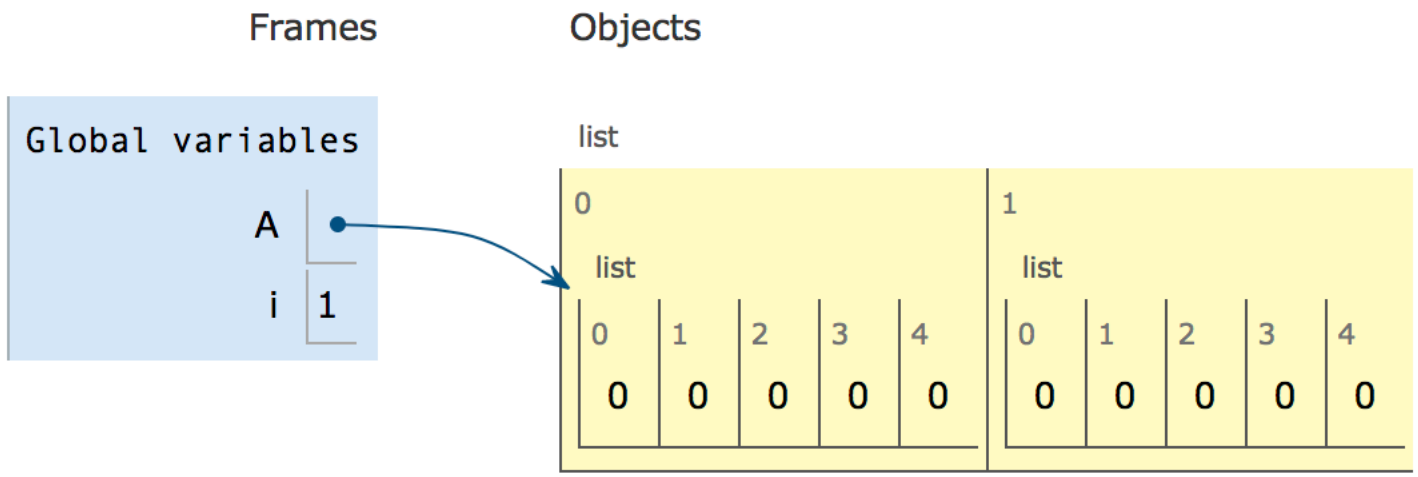
Criação de Matrizes

```
1 A = []
2 for i in range(5):
3     A.append( [0] * 5 )
4 A[1][1] = 2
```

<< First | < Back | Step 6 of 13 | Forward > | Last >>

line that has just executed

next line to execute



python™

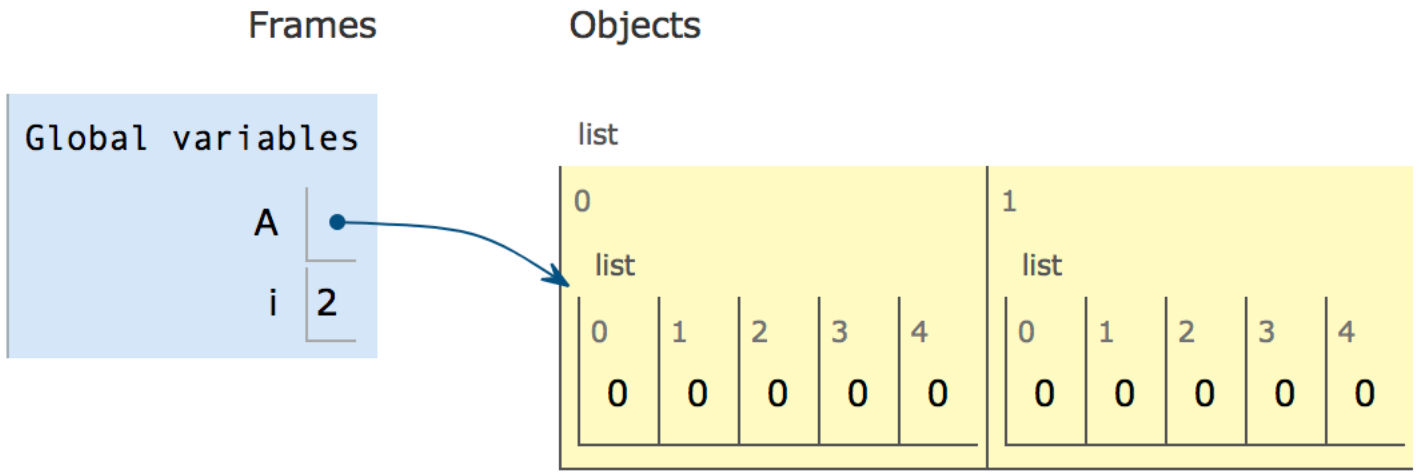
Criação de Matrizes

```
1  A = []
2  for i in range(5):
3      A.append( [0] * 5 )
4  A[1][1] = 2
```

<< First | < Back | Step 7 of 13 | Forward > | Last >>

line that has just executed

next line to execute



python™

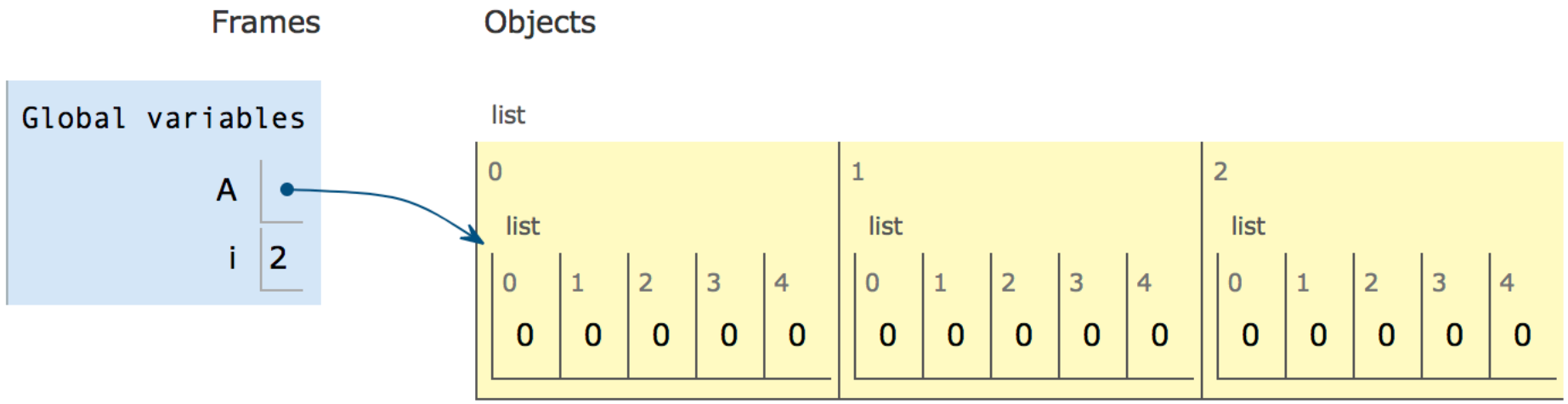
Criação de Matrizes

```
1  A = []
2  for i in range(5):
3      A.append( [0] * 5 )
4  A[1][1] = 2
```

<< First | < Back | Step 8 of 13 | Forward > | Last >>

line that has just executed

next line to execute



python™

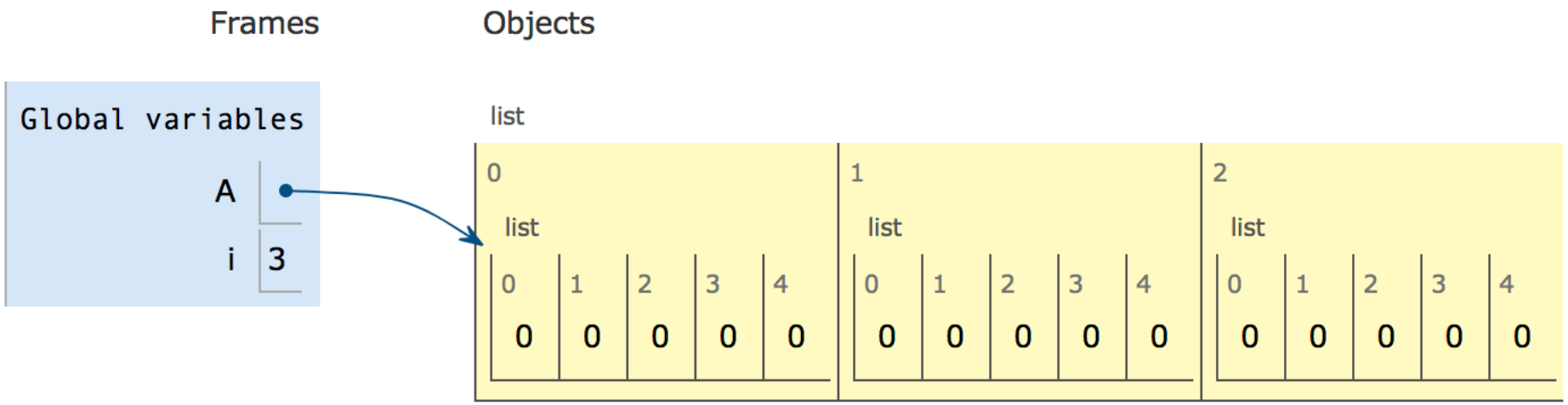
Criação de Matrizes

```
1  A = []
2  for i in range(5):
3      A.append( [0] * 5 )
4  A[1][1] = 2
```

<< First | < Back | Step 9 of 13 | Forward > | Last >>

line that has just executed

next line to execute



python™

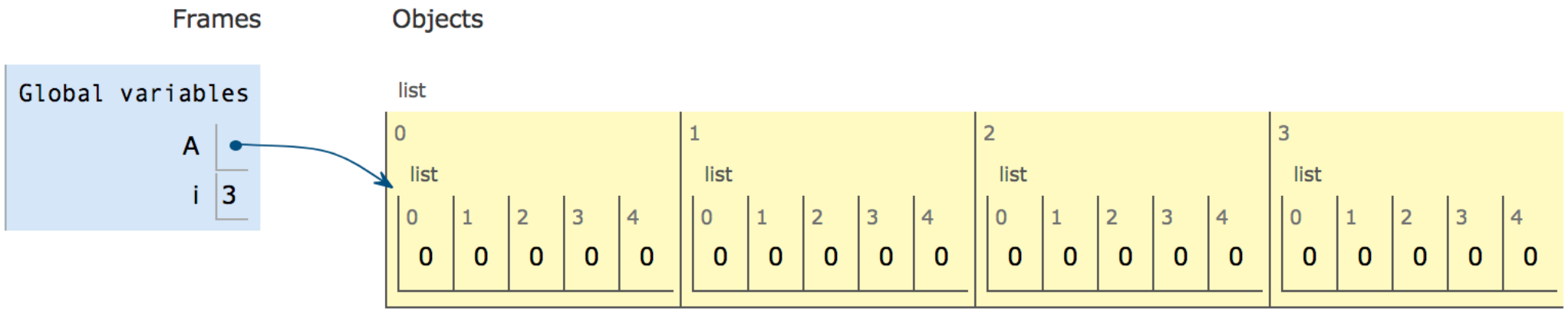
Criação de Matrizes

```
1 A = []
2 for i in range(5):
3     A.append( [0] * 5 )
4 A[1][1] = 2
```

<< First | < Back | Step 10 of 13 | Forward > | Last >>

line that has just executed

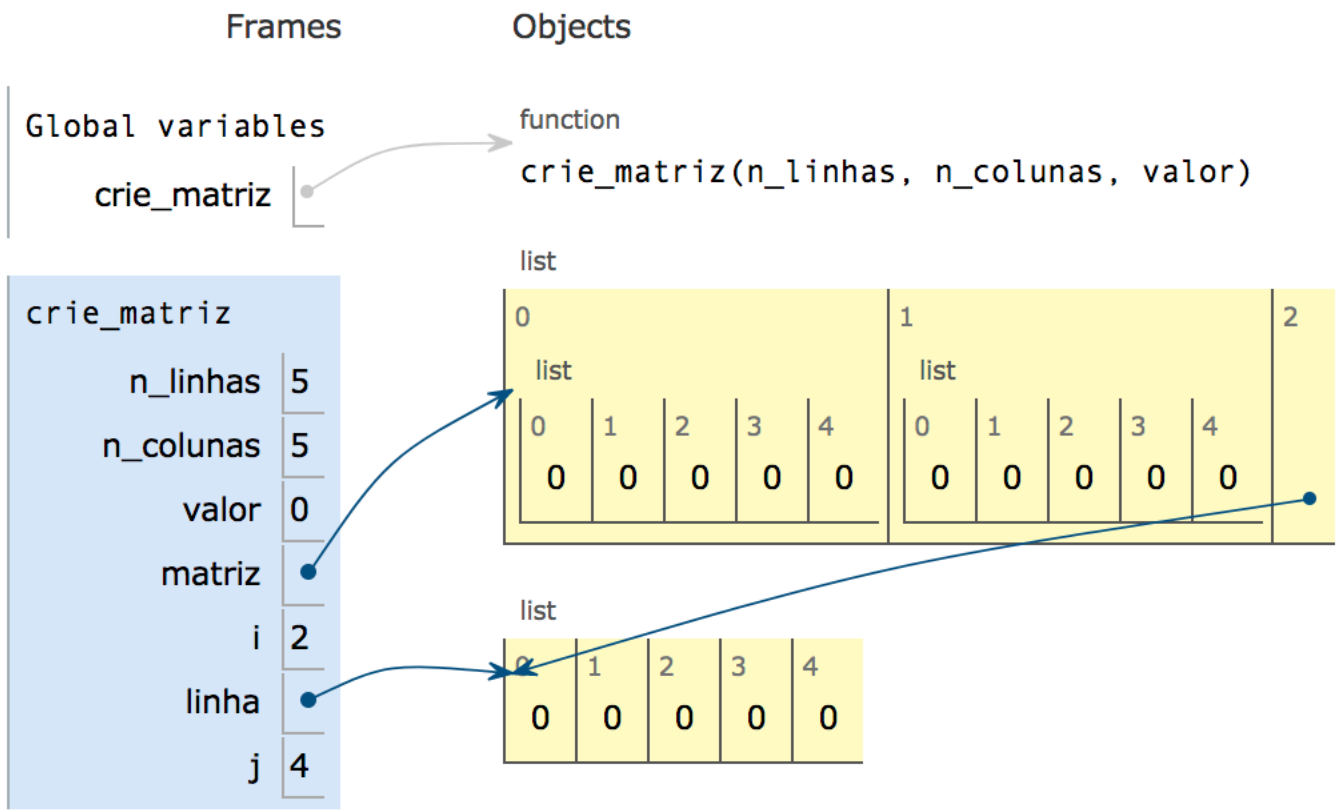
next line to execute



python™

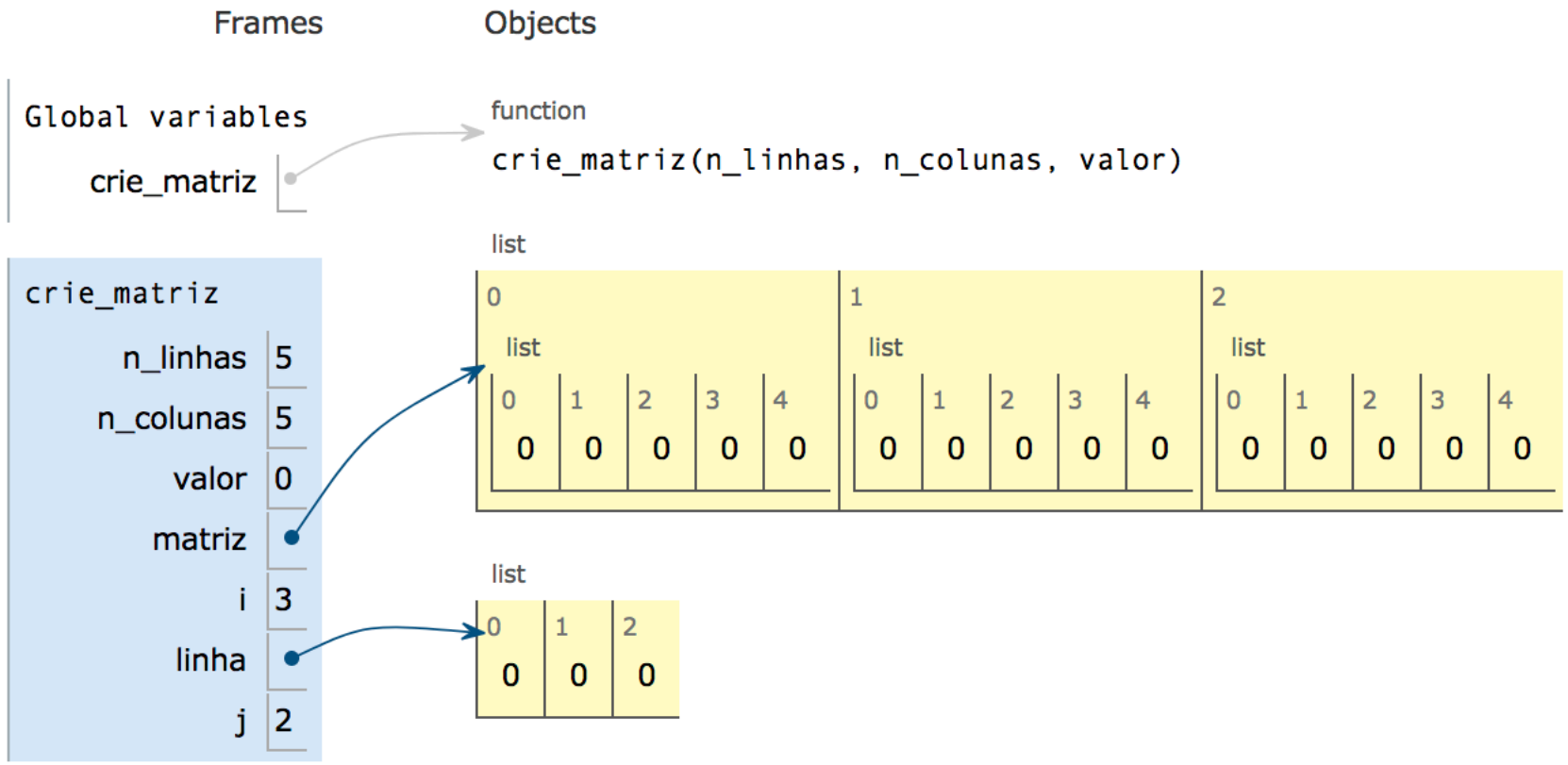
Criação de Matrizes com Funções

```
def crie_matriz(n_linhas, n_colunas, valor):
    # lista vazia
    matriz = []
    for i in range(n_linhas):
        # cria a linha i
        linha = [] # lista vazia
        for j in range(n_colunas):
            linha.append(valor)
        # coloque linha na matriz
        matriz.append(linha)
    return matriz


#----------------------
A = crie_matriz(5,5,0)
A[1][1] = 2
```



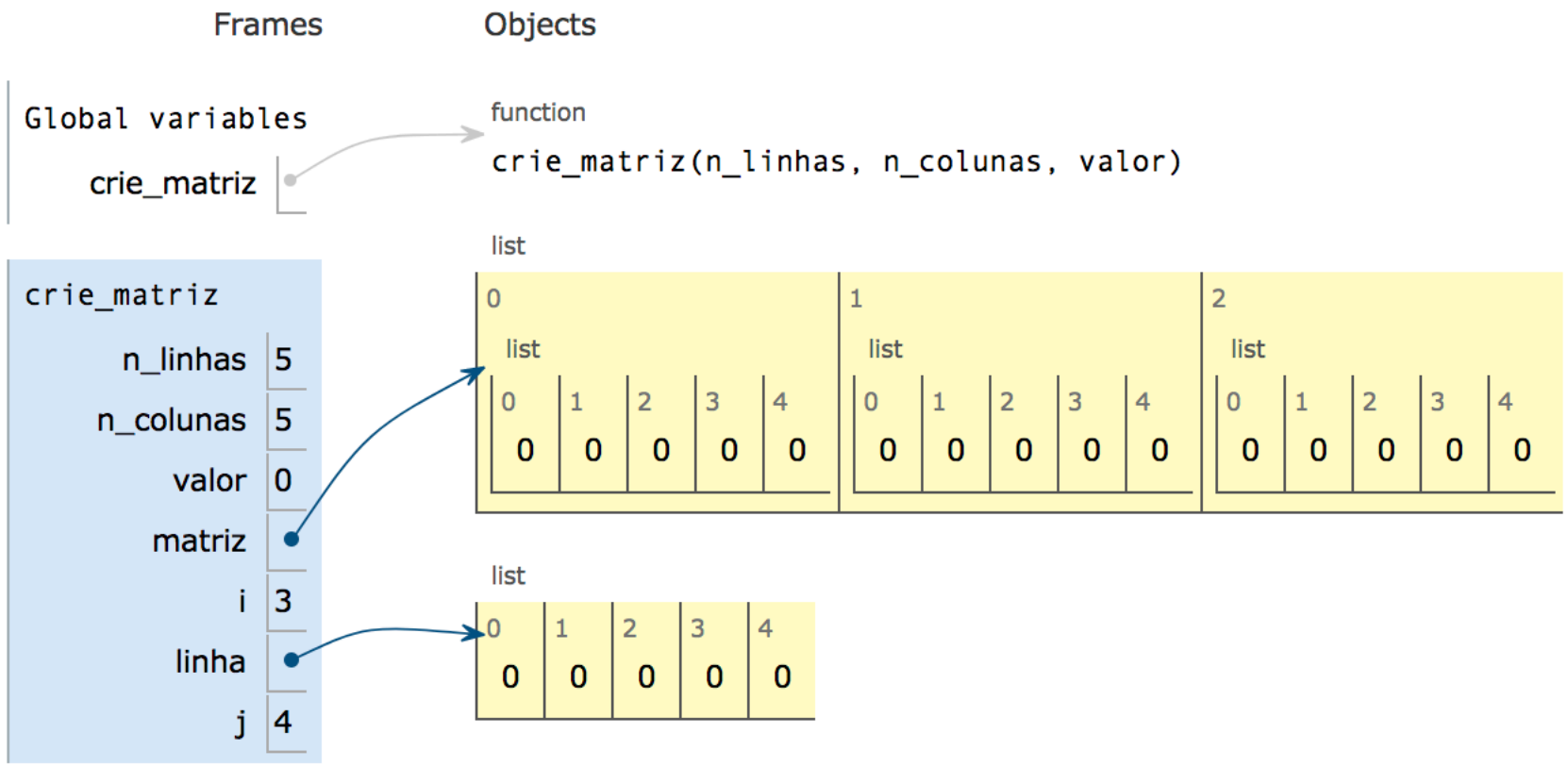
Criação de Matrizes com Funções

```
def crie_matriz(n_linhas, n_colunas, valor):
    # lista vazia
    matriz = []
    for i in range(n_linhas):
        # cria a linha i
        linha = [] # lista vazia
        for j in range(n_colunas):
            linha.append(valor)
        # coloque linha na matr
        matriz.append(linha)
    return matriz


#----------------------
A = crie_matriz(5,5,0)
A[1][1] = 2
```



Criação de Matrizes com Funções

```
def crie_matriz(n_linhas, n_colunas, valor):
    # lista vazia
    matriz = []
    for i in range(n_linhas):
        # cria a linha i
        linha = [] # lista vazia
        for j in range(n_colunas):
            linha.append(valor)
        # coloque linha na matr
        matriz.append(linha)
    return matriz


#----------------------
A = crie_matriz(5,5,0)
A[1][1] = 2
```



Tarefas

1)

```
1  #-----------------------------------------------------
2  def leia_matriz():
3      '''(None) -> matriz (lista de listas)
4
5      Funcao que le do teclado o numero n_linhas de linhas
6      e o numero n_colunas de colunas e os elementos de
7      uma matriz de inteiros dimensao n_linha x n_colunas.
8
9      A funcao cria e retorna a matriz lida.
10     '''
11     print("Vixe! Ainda nao fiz a funcao!")
12
13 #-----------------------------------------------------
14 # teste
15 a_mat = leia_matriz()
16 print(a_mat)
17
18
```

```
>>> a = leia_matriz()
Digite o número de linhas: 3
Digite o número de colunas: 4
matriz = []
linha 0 = []
Digite o elemento (0,0): 1
linha 0 = [1]
Digite o elemento (0,1): 2
linha 0 = [1, 2]
Digite o elemento (0,2): 3
linha 0 = [1, 2, 3]
Digite o elemento (0,3): 4
linha 0 = [1, 2, 3, 4]
matriz = [[1, 2, 3, 4]]
linha 1 = []
Digite o elemento (1,0): 5
NetBeans 8.2
ento (1,1): 6
linha 1 = [5, 6]
Digite o elemento (1,2): 7
linha 1 = [5, 6, 7]
Digite o elemento (1,3): 8
linha 1 = [5, 6, 7, 8]
matriz = [[1, 2, 3, 4], [5, 6, 7, 8]]
linha 2 = []
Digite o elemento (2,0): 9
linha 2 = [9]
Digite o elemento (2,1): 10
linha 2 = [9, 10]
Digite o elemento (2,2): 11
linha 2 = [9, 10, 11]
Digite o elemento (2,3): 12
linha 2 = [9, 10, 11, 12]
matriz = [[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]]
>>> a
```

Tarefas

2)

```
#-----------------------------------------------
def imprima_matriz(matriz):
    '''(matriz) -> None

    Recebe e imprime uma matriz de inteiros.

    >>> a = [[1,2,3],[2,1,4],[3,4,1]]
    >>> a
    [[1, 2, 3], [2, 1, 4], [3, 4, 1]]
    >>> imprima_matriz(a)
    Matriz: 3 x 3
          1     2     3
          2     1     4
          3     4     1

    '''
    print("Vixe! Ainda nao fiz a funcao!")

#-----------------------------------------------
# testes
a = [[1,2,3],[2,1,4],[3,4,1]]
imprima_matriz(a)
```

Tarefas

3)

```
def imprima_matriz(matriz):
    '''(matriz) -> None
    Recebe e imprime uma matriz de inteiros.
    >>> a = [[1,2,3],[2,1,4],[3,4,1]]
    >>> a
    [[1, 2, 3], [2, 1, 4], [3, 4, 1]]
    >>> imprima_matriz(a)
    Matriz: 3 x 3
        1    2    3
        2    1    4
        3    4    1'''
    print("Vixe! Ainda nao fiz a funcao!")

a = [[1,2,3],[2,1,4],[3,4,1]]
imprima_matriz(a)
```