Computação 1, 2021.1
Lista 6

Data limite para entrega: 31/08 às 23:59

Submeta suas soluções colocando os arquivos correspondentes na sua pasta do Google Drive*

Lembre-se de escolher bons nomes para suas funções e variáveis, de documentar seu código com *docstrings* (documentação de função) e comentários onde for apropriado, e de fazer testes para suas funções (na documentação usando o módulo `doctest` ou em funções dedicadas).

Questão 1. Transforme os seguintes programas em *list comprehensions*

a.

```
L1 = []
for i in range(10):
    L1.append(i**2+3)
```

b.

```
L2 = [5,4,3,2,1]*3
L3 = []
for i in L2[2:8]:
    if 2**i < 99:
        L3.append(i**2+3)
```

c.

```
L1 = []
for i in range(10):
    L2 = []
    for j in range(i):
        L2.append(i**j)
    L1.append(L2)
```

d.

```
L3 = []
for i in range(10):
    for j in range(i):
        L3.append(i**j)
```

*Link recebido por email em 19/7/2021 — o nome é parecido com `<seu nome> - Computação 1 - Submissões e Feedback`.

Questão 2 (Matrizes)**.** Podemos representar uma matriz real M com ℓ linhas e c colunas,

$$M = \begin{pmatrix} m_{0,0} & m_{0,1} & m_{0,2} & \cdots & m_{0,c-1} \\ m_{1,0} & m_{1,1} & m_{1,2} & \cdots & m_{1,c-1} \\ m_{2,0} & m_{2,1} & m_{2,2} & \cdots & m_{2,c-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ m_{\ell-1,0} & m_{\ell-1,1} & m_{\ell-1,2} & \cdots & m_{\ell-1,c-1} \end{pmatrix},$$

em `python` como uma lista `L` de comprimento ℓ, onde cada elemento é por sua vez uma lista de comprimento c, de forma que dados $0 \leq i < \ell$ e $0 \leq j < c$ tenhamos

$$\texttt{L}[i][j] = \tilde{m}_{i,j},$$

sendo $\tilde{m}_{i,j}$ o `float` correspondente ao real $m_{i,j}$.

Em outras palavras,

$$\begin{aligned} \texttt{L} = [&[\tilde{m}_{0,0},\ \tilde{m}_{0,1},\ \tilde{m}_{0,2},\ \ldots,\ \tilde{m}_{0,c-1}], \\ &[\tilde{m}_{1,0},\ \tilde{m}_{1,1},\ \tilde{m}_{1,2},\ \ldots,\ \tilde{m}_{1,c-1}], \\ &[\tilde{m}_{2,0},\ \tilde{m}_{2,1},\ \tilde{m}_{2,2},\ \ldots,\ \tilde{m}_{2,c-1}], \\ &\qquad\qquad\vdots\qquad\qquad\qquad\quad, \\ &[\tilde{m}_{\ell-1,0},\ \tilde{m}_{\ell-1,1},\ \tilde{m}_{\ell-1,2},\ \ldots,\ \tilde{m}_{\ell-1,c-1}]]. \end{aligned}$$

(Essa forma de representar uma matriz privilegia as linhas em detrimento das colunas; poderíamos muito bem ter escolhido uma forma que privilegiasse as colunas em detrimento das linhas.)

a. Faça uma função que receba uma lista de listas, e retorne `True` se for uma representação de matrizes como acima, e `False` caso contrário.

b. Faça uma função que receba uma matriz na forma descrita acima e retorne a quantidade de linhas da matriz.

c. Faça uma função que receba uma matriz na forma descrita acima e retorne a quantidade de colunas da matriz.

d. Faça uma função que receba uma matriz na forma descrita acima e um número inteiro i e retorne a i-ésima linha da matriz dada, caso exista, ou algum tipo de erro informativo em caso contrário[1].

e. Faça uma função que receba uma matriz na forma descrita acima e um número inteiro j e retorne a j-ésima coluna da matriz dada, caso exista, ou algum tipo de erro informativo em caso contrário.

f. Faça uma função que receba uma matriz na forma descrita acima e retorne sua transposta.

g. No mesmo espírito, podemos representar um vetor real

$$v = (v_0, v_1, v_2, \ldots, v_{n-1})$$

[1] No caso de erro, "retornado" deve ser entendido em sentido amplo, não necessariamente usando um `return`. O mesmo comentário vale para as questões seguintes.

em `python` usando a lista

$$[\tilde{v}_0,\ \tilde{v}_1,\ \tilde{v}_2,\ \ldots,\ \tilde{v}_{n-1}],$$

onde novamente $\tilde{v}_i$ é o `float` que representa o real v_i.

Dados dois vetores reais de mesmo comprimento $u = (u_0, \ldots, u_{n-1})$ e $v = (v_0, \ldots, v_{n-1})$, seu *produto escalar* é o número real

$$\sum_{i=0}^{n-1} (u_i \cdot v_i).$$

Faça uma função que receba como entrada dois vetores (representados como acima) e retorne seu produto escalar, caso exista, ou algum tipo de erro informativo em caso contrário.

h. O *produto* de duas matrizes reais M e N existe quando (e apenas quando) o número de colunas de M é igual ao número de linhas de N. Neste caso, o produto é a matriz real cujo número de linhas coincide com o de M e cujo número de colunas coincide com o de N, e tal que o real que ocupa a i-ésima linha e j-ésima coluna é o produto escalar da i-ésima linha de M com a j-ésima coluna de N (vistos como vetores). Implemente essa operação em `python`, usando as representações descritas acima; caso o produto das matrizes dadas não exista, novamente, algum tipo de erro informativo deve ser retornado.

Desafios (opcionais)

Questão 3. Transforme os seguintes códigos para construir as listas usando *list comprehensions*.

a.

```
L5 = []
L6 = []
for i in range(10):
    for j in range(10):
        L6.append(i**j)
    L1.append(L6)
```

b.

```
L4 = []
for i in range(10):
    s = sum(L4)
    L4.append(i**2+s)
```