Lista de Exercícios – Matrizes
Prof: Yuri Frota

MATRIZES

1) Faça um programa que recebe como parâmetro uma matriz A (nxn) de números reais. Essa função deve informar

(a) a soma dos elementos da coluna “n”

(b) a soma dos elementos da diagonal principal da matriz e

(c) a soma dos elementos acima da diagonal principal

(d) o número de células da matriz que têm valor menor que a média dos valores das células da matriz,

R:

```
import random


A = []


n = random.randint(1,10) # numero de linhas e colunas de A


for i in range(n): # cria matriz nxn aleatoria
  l = []
  for j in range(n): # cria linha aleatoria
    l.append(random.randint(1,10))
  A.append(l) # adiciona linha a matriz


soma_col_n = 0
soma_diag = 0
for i in range(n): # calcula soma da n-ésima coluna
  soma_col_n += A[n-1][i]
  soma_diag += A[i][i]
```

```
soma_acima_diag = 0
soma_mat = 0
for i in range(n):
  for j in range(n):
    if j > i:
      soma_acima_diag += A[i][j]
    soma_mat += A[i][j]

media = soma_mat/(n*n) # calcula a media
n_abaixo_media = 0

for i in range(n):
  for j in range(n):
    if A[i][j] < media:
      n_abaixo_media += 1

for i in range(n):
  print(A[i])
print(soma_col_n)
print(soma_diag)
print(soma_acima_diag)
print(media, n_abaixo_media)
```

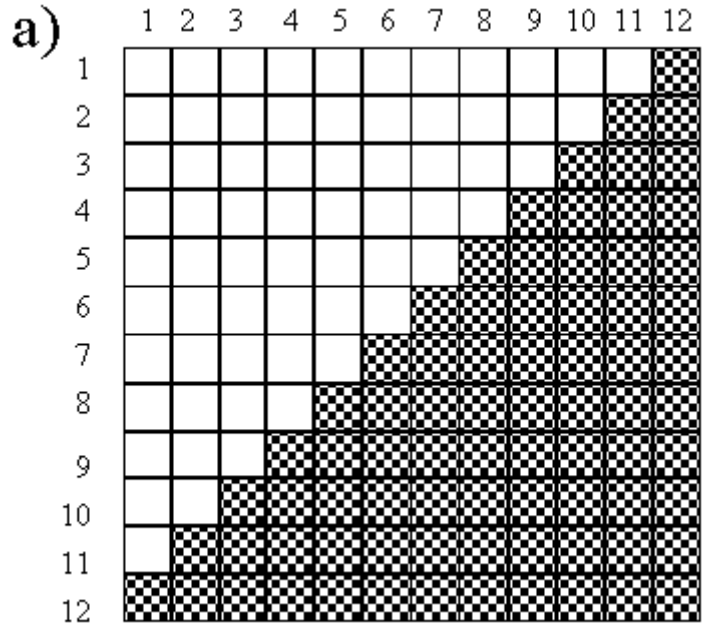
2) Faça um programa que receba uma matriz A (nx m) e diga se esta matriz é simétrica

R:

```
A = [] # matriz nxm

n = int(input('Digite o número de linhas da matriz: '))
m = int(input('Digite o número de colunas da matriz: '))

print("Digite os números da matriz: ")
for i in range(n):
  l = []
  for j in range(n):
    l.append(int(input()))
  A.append(l)

print(A)

simetrica = True

n = len(A)
m = len(A[0])

if n != m:
  simetrica = False # A só é simetrica se for quadrada
else:
  # analisa o triângulo superior da matriz
  for i in range(n):
    for j in range(i+1, n):
      if A[i][j] != A[j][i]:
        simetrica = False
        break
```

```
    if simetrica == False:
      break


if simetrica:
  print('A é uma matriz simétrica')
else:
  print('A não é uma matriz simétrica')
```

3) Faça um programa que receba uma matriz A (nxn) e calcule e escreva o menor elemento e a sua posição (índices) de sua área hachurada. A Figura a) fornece um exemplo de uma matriz A (12x12)



R:

```
import math
import random


A = [] # matriz nxn
```

```
n = random.randint(1,10) # numero de linhas e colunas de A

for i in range(n): # cria matriz nxn aleatoria
  l = []
  for j in range(n): # cria linha aleatoria
    l.append(random.randint(1,10))
  A.append(l) # adiciona linha a matriz

print(A)

n = len(A)

min_i, min_j = -1, -1 # indices do menor elemento
min_val = math.inf

# analisa elementos da diagonal secundária para baixo
for i in range(n):
  for j in range(n):
    # verifica se pertence ao conjunto hachurado
    if i + j >= n-1:
      if A[i][j] < min_val:
        min_val = A[i][j]
        min_i = i
        min_j = j

print(min_val, "(i=" + str(min_i), "j=" + str(min_j) + ")")
```

4) Faça um programa que receba uma matriz A (nx m) e divida cada um dos “n” elementos de cada uma das “m” colunas pelo maior elemento em módulo daquela coluna. Imprima a matriz modificada

R:

```
import random


A = [] # matriz nxm


n = random.randint(1,10) # numero de linhas de A

m = random.randint(1,10) # numero de colunas de A


for i in range(n): # cria matriz nxn aleatoria

  l = []

  for j in range(m): # cria linha aleatoria

    l.append(random.randint(-10,10))

  A.append(l) # adiciona linha a matriz




n = len(A)

m = len(A[0])


B = []

for i in range(n): # cria matriz B

  l = []

  for j in range(m):

    l.append(0)

  B.append(l)


for j in range(m):
```

```
  maior_mod = 0
  for i in range(n):
    if abs(A[i][j]) > maior_mod:
      maior_mod = abs(A[i][j])
  for i in range(n):
    B[i][j] = round( A[i][j]/maior_mod, 2) # precisao de 2 casas

for i in range(n):
  print(A[i])

print('\n')

for i in range(n):
  print(B[i])
```

5) Faça um programa que receba duas matrizes A (nx n) e B (nxn) e

(a) imprime as matrizes A e B,

(b) imprima a soma das matrizes (A+B),

(c) imprime a diferença das matrizes (A-B)

(d) imprima a multiplicação das matrizes (AxB)

(e) imprima a transposta de A

R:

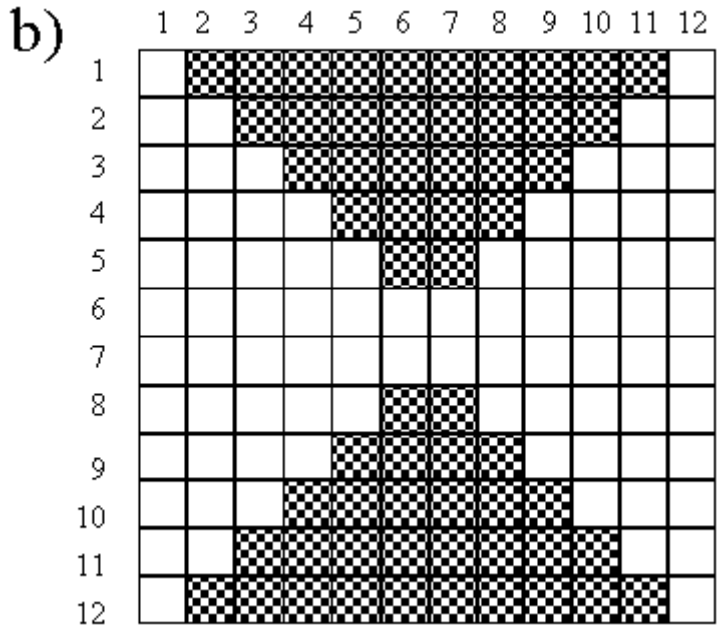
```
import random

A = [] # matriz nxn
B = [] # matriz nxn
```

```
n = random.randint(1,10) # numero de linhas e colunas de A

for i in range(n): # cria matriz nxn aleatoria
  l1 = []
  l2 = []
  for j in range(n): # cria linha aleatoria
    l1.append(random.randint(1,10))
    l2.append(random.randint(1,10))
  A.append(l1) # adiciona linha em A
  B.append(l2) # adiciona linha em B

n = len(A)

print("A:")
for i in range(n):
  l = "[ "
  for j in range(n):
    l = l + str(A[i][j]) + " "
  l = l + "]"
  print(l) # imprime linha da matriz

print('\nB:')
for i in range(n):
  l = "[ "
  for j in range(n):
    l = l + str(B[i][j]) + " "
  l = l + "]"
  print(l) # imprime linha da matriz
```

```
print('\nA + B:')
for i in range(n):
  l = "[ "
  for j in range(n):
    l = l + str(A[i][j] + B[i][j]) + " "
  l = l + "]"
  print(l) # imprime linha da matriz

print('\nA - B:')
for i in range(n):
  l = "[ "
  for j in range(n):
    l = l + str(A[i][j] - B[i][j]) + " "
  l = l + "]"
  print(l) # imprime linha da matriz

print('\nA x B:')
for i in range(n):
  l = "["
  # produto interno da linha i de A
  # com cada coluna j de B
  for j in range(n):
    soma = 0
    for k in range(n):
      soma += A[i][k] * B[k][j]
    l = l + str(soma) + " "
  l = l + "]"
```

```
  print(l)


print('\nA^t:')
for i in range(n):
  l = "[ "
  for j in range(n):
    l = l + str(A[j][i]) + " "
  l = l + "]"
  print(l)
```

6) Faça um programa que receba uma matriz A (nxn) e calcule e escreva a média dos elementos da sua área hachurada. A Figura b) fornece um exemplo de uma matriz A (12x12)



R:

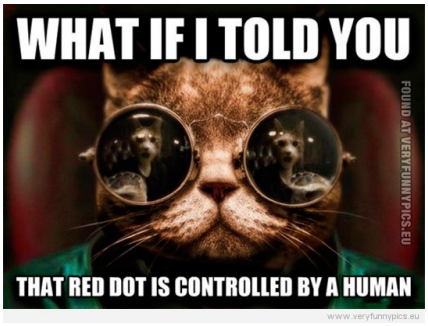
Um elemento está hachurado se está acima da diagonal principal **e** da diagonal secundária (J > i e i+j ≤ n-2, para índices de 0 até n-1 na matriz), **ou** se o elemento está abaixo da diagonal secundária **e** da diagonal principal (i+j > n-1 e j < i, para índices de 0 até n-1 na matriz). Dessa forma, segue o código:

```
import random

A = [] # matriz nxn

n = random.randint(1,10) # numero de linhas e colunas de A

for i in range(n): # cria matriz nxn aleatoria
  l = []
  for j in range(n): # cria linha aleatoria
    l.append(random.randint(1,10))
  A.append(l) # adiciona linha em A

for i in range(n):
  print(A[i])

n = len(A)

soma = 0
contador = 0

# analisa elementos da diagonal secundária para baixo
for i in range(n):
  for j in range(n):
    # condição para ser hachurado
    if (j > i and i+j <= n-2) or (i+j > n-1 and j < i):
      soma += A[i][j]
      contador += 1
```

```
print("Média: ", soma/contador)
```

7) Faça um programa que receba uma matriz A (nxm) e ordene os elementos de cada linha da matriz. Imprimir o resultado

R:




```
import random


A = [] # matriz nxm


n = random.randint(1,10) # numero de linhas de A

m = random.randint(1,10) # numero de colunas de A


for i in range(n): # cria matriz nxn aleatoria

  l = []

  for j in range(m): # cria linha aleatoria

    l.append(random.randint(0,10))

  A.append(l) # adiciona linha a matriz


n = len(A)

m = len(A[0])


for i in range(n):


  print(A[i]) # imprime linha atual


  # ordenação por inserção da linha

  for j in range(1,m):
```

```
    k = j

    while A[i][k] < A[i][k-1] and k > 0:
      aux = A[i][k]
      A[i][k] = A[i][k-1] # troca
      A[i][k-1] = aux
      k = k-1

print(A[i]) # imprime linha ordenada
print('\n')
```