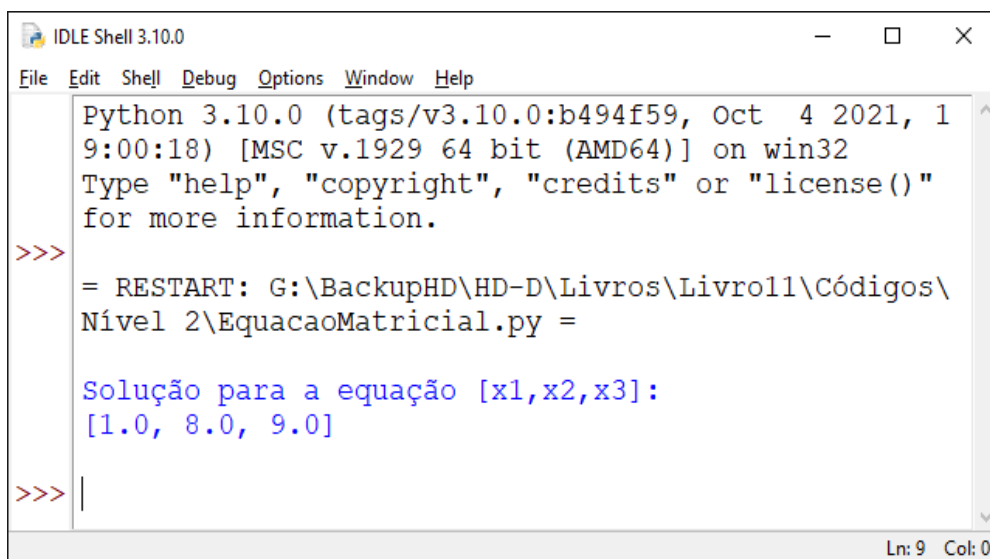


Equação Matricial

Mário Leite

...

Para a equação $3x = 15$ (do primeiro grau) a solução é muito simples: basta dividir 15 por 3, obtendo o valor **5** para a incógnita **x**. Entretanto, se for uma equação do tipo $AX = B$, onde **A** e **X** são matrizes de uma mesma dimensão e **B** um vetor de mesmo número de colunas de **A** a solução é um pouco mais elaborada, pois é uma equação matricial cujo objetivo é encontrar os elementos da matriz **X** (x_1, x_2, x_3, \dots) mesmo que, do ponto de vista lógico, a solução seria dividir o vetor **B** pela matriz **A**. Mas, como a operação de divisão não é definida para matrizes o processo não é tão simples como na solução de uma equação “normal” do primeiro grau. E embora a solução desse tipo de equação seja um pouco mais complexa, não é tão difícil assim. O programa “EquacaoMatricial” apresenta uma solução, razoavelmente simples, para resolver uma equação matricial. A **figura 1** mostra a saída do programa codificado em **Python** e a **figura 2** a comprovação dos resultados usando a ferramenta **SciLab**.



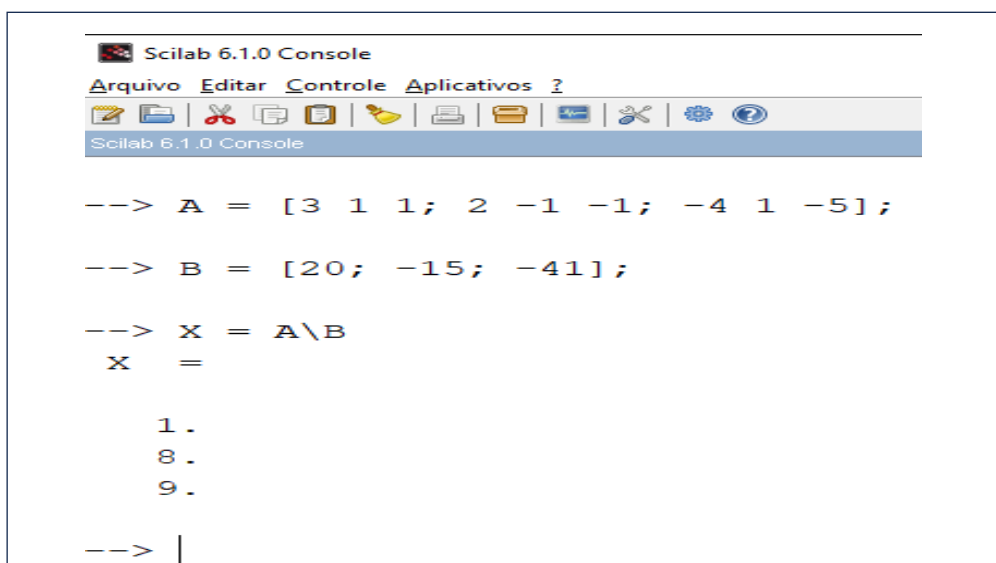
```
Python 3.10.0 (tags/v3.10.0:b494f59, Oct 4 2021, 19:00:18) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()"
for more information.

>>>
= RESTART: G:\BackupHD\HD-D\Livros\Livro11\Códigos\
Nível 2\EquacaoMatricial.py =

Solução para a equação [x1,x2,x3]:
[1.0, 8.0, 9.0]

>>> |
```

Figura 1 -Solução com Python



```
SciLab 6.1.0 Console

Arquivo Editar Controle Aplicativos ?

SciLab 6.1.0 Console

--> A = [3 1 1; 2 -1 -1; -4 1 -5];

--> B = [20; -15; -41];

--> X = A\B
X =

    1.
    8.
    9.

--> |
```

Figura 2 - Solução com SciLab

```

'''
EquacaoMatricial.py
-----

Resolve um sistema de equação matricial do tipo A*X=B (matriz A e vetor B)
através do "Método de Cramer" para calcular as incógnitas: x1, x2, x3 de um
exemplo dado.
-----

Data: 27/08/2023
Autor: Mário Leite
-----

'''
def ResolverEquacao(A, B):
    #Resolve a equação matricial
    n = len(A)
    if(len(B) != n):
        raise ValueError("Matriz A e vetor B têm que ter as mesmas dimensões!")
    detA = CalcularDeterminante(A)
    if (detA == 0):
        raise ValueError("Matriz A é singular: solução indeterminada.")
    LstX = []
    for i in range(n):
        tempA = [row[:] for row in A]
        for j in range(n):
            tempA[j][i] = B[j]
        detAi = CalcularDeterminante(tempA)
        LstX.append(detAi / detA)

    return LstX
#-----
def CalcularDeterminante(A):
    #Calcula o determinante da matriz A
    n = len(A)
    if(n == 1):
        return A[0][0]
    det = 0
    sinal = 1
    for j in range(n):
        subMat = [row[:j] + row[j+1:] for row in A[1:]]
        det += sinal * A[0][j] * CalcularDeterminante(subMat)
        sinal *= -1

    return det
#=====
#Chama a função para resolver a equação dada a matriz A e o vetor B
A = [[3, 1, 1], [2, -1, -1], [-4, 1, -5]]
B = [20, -15, -41]
LstX = ResolverEquacao(A, B)

#Mostra o resultado
print("Solução para a equação [x1,x2,x3]: ")
print(LstX)
print()
#Fim do programa "EquacaoMatricial" -----

```