Adamson University
College of Engineering
Computer Engineering Department

Linear Algebra

Laboratory Activity No. 6

Matrices

Submitted by:
Roa, Danica Kate I.

Instructor:
Engr. Dylan Josh D. Lopez

November 28 , 2020

I. Objectives

This laboratory activity aims to Be familiar with matrices and their relation to linear equations. Perform basic matrix operations. Program and translate matrix equations and operations using Python.

II. Methods

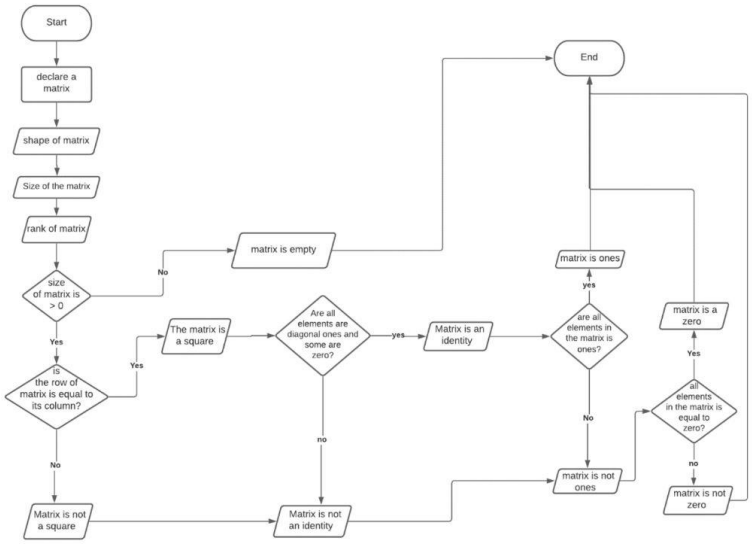

Figure 1: flowchart Activity 1

In figure 1 demonstrated the flowchart for the undertaking which to make a program that portrays a grid. The program should begin in proclaiming its framework then it should show the state of the lattice, size of the network, and rank of the grid. At that point it will head toward certain conditions that will decide whether the lattice is vacant, if not it will keep on deciding whether the framework is a square it ought to have equivalent lines and segments followed by recognizing if the network has a 1 in the inclining area and 0 in it. It will show lattice is a personality. At that point if all the components in the grid are 1 it will show the network as ones accordingly if the components of the lattice are in zero it will show the network as zero.

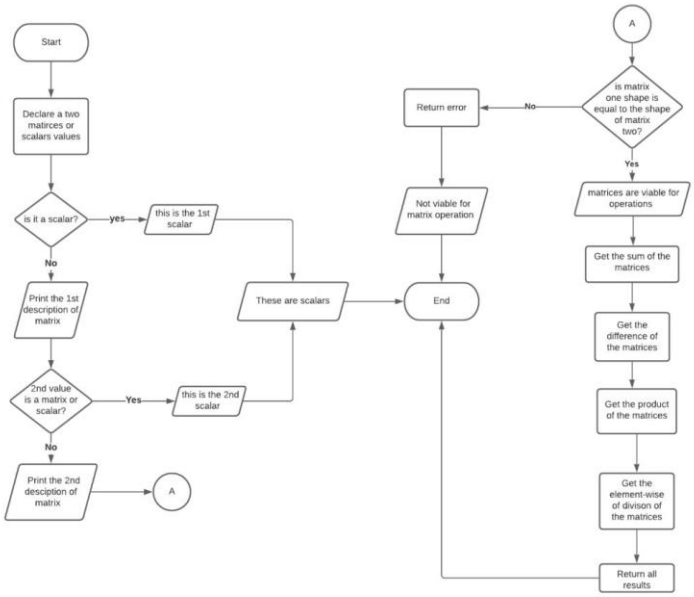

Figure 2: Flowchart Activity 1

In figure 2 demonstrated the flowchart for task 2 which to make a program that decides its boundaries, portrayal of networks, and lattice tasks. The first is to announce the estimations of two grids or scalars. At that point it will decide whether the boundaries are a scalar if not it will show the primary depiction of the lattice, for example, shape, size, and rank. At that point for the subsequent boundary on the off chance that it's a lattice it will do a similar methodology as in the main framework. At that point it will proceed through the program conditions, if the two lattices have an equivalent shape it will continue to the network activities which has and if the condition isn't met the program will re-visitation of its mistake message. The acts of this movement are to be comfortable with essential activities and with their relations, just as to interpret the grid conditions. The expectations of this movement are to make a code and capacity that will show the portrayal of the network and play out its relating lattice tasks.

III. Results

```
def mat_desc(matrix):
   print(f'{matrix}\n\nShape:\t{matrix.shape}\nSize:\t{np.product(matrix.shape)}\nRank:\t{matrix.ndim}\n')
   if matrix.size > 0:
    is_square = True if matrix.shape[0] == matrix.shape[1] else False
    is_identity = True if matrix.shape[0] == matrix.shape[1] and np.allclose(matrix, np.identity(matrix.shape[0])) else False
    is_ones = True if np.all((matrix == 1)) else False
    is_zero = True if np.all((matrix == 0)) else False
    print(
        f'is it a square?: {is_square}\n'
        f'is it identity?: {is_identity}\n'
        f'Is is a ones: {is_ones}\n'
        f'Is it a zero: {is_zero}\n'
        )
   else:
     print("Matrix is Empty")
```

Figure 3: Code for Activity 1

```
A=np.array([
[1,2,3],
[4,5,6],
[7,8,9]])

B=np.array([
[2,4,6],
[8,10,12],
[14,16,18]])

C=np.array([
[3,6,9],
[12,15,16],
[18,20,22]])

D=np.array([
[4,8,12],
[16,20,24],
[28,32,36]])

E=np.array([
[5,10,15],
[20,25,30],
[35,40,45]])
```

Figure 4: Matrices

```
print("First Matrix:")
mat_desc(A)
```

```
First Matrix:
[[1 2 3]
 [4 5 6]
 [7 8 9]]

Shape:  (3, 3)
Size:   9
Rank:   2

is it a square?: True
is it identity?: False
Is is a ones: False
Is it a zero: False
```

Figure 5: Sample Declaration task 1

```
print("Second Matrix:")
mat_desc(B)
```

```
Second Matrix:
[[ 2  4  6]
 [ 8 10 12]
 [14 16 18]]

Shape:  (3, 3)
Size:   9
Rank:   2

is it a square?: True
is it identity?: False
Is is a ones: False
Is it a zero: False
```

Figure 6 : Sample Declaration task 1

```
print("Third Matrix:")
mat_desc(C)
```

```
Third Matrix:
[[ 3  6  9]
 [12 15 16]
 [18 20 22]]

Shape:  (3, 3)
Size:   9
Rank:   2

is it a square?: True
is it identity?: False
Is is a ones: False
Is it a zero: False
```

Figure 7: Sample Declaration task 1

```
print("Fourth Matrix:")
mat_desc(D)
```

```
Fourth Matrix:
[[ 4  8 12]
 [16 20 24]
 [28 32 36]]

Shape:  (3, 3)
Size:   9
Rank:   2

is it a square?: True
is it identity?: False
Is is a ones: False
Is it a zero: False
```

Figure 8: Sample Declaration task 1

```
print("Fifth Matrix:")
mat_desc(E)
```

```
Fifth Matrix:
[[ 5 10 15]
 [20 25 30]
 [35 40 45]]

Shape:  (3, 3)
Size:   9
Rank:   2

is it a square?: True
is it identity?: False
Is is a ones: False
Is it a zero: False
```

Figure 9: Sample Declaration task 1

In Figure 3 to Figure 9 shown as the code snippet for task 1, the requirement is to display the description of the matrix and create 5 samples of the matrix. To describe the matrices, mat_operations() is used.

```
def mat_operations(A,B):
    alpha=10**-10
    if(A.ndim & B.ndim)==0:
        print(f'scalar:\n{matrix1},\n{B}\n')
        print("This is scallar sample")
    else:
        print(f'Matrix:\n{A},\n{B}')
        if(A.shape==B.shape):
            sum=A+B
            difference=A-B
            product=A*B
            quotient=A//B+alpha
            print("This is matrix example")
            return sum,difference,product,quotient
        else:
            return
```

Figure 10: Code for Activity 2

```
print('First Pair')
mat_operations(A,B)
```

```
First Pair
Matrix:
[[1 2 3]
 [4 5 6]
 [7 8 9]],
[[ 2  4  6]
 [ 8 10 12]
 [14 16 18]]
This is matrix example

(array([[ 3,  6,  9],
        [12, 15, 18],
        [21, 24, 27]]),
 array([[-1, -2, -3],
        [-4, -5, -6],
        [-7, -8, -9]]),
 array([[  2,   8,  18],
        [ 32,  50,  72],
        [ 98, 128, 162]]),
 array([[1.e-10, 1.e-10, 1.e-10],
        [1.e-10, 1.e-10, 1.e-10],
        [1.e-10, 1.e-10, 1.e-10]]))
```

Figure 11: Sample Declaration task 2

```
print('Second Pair')
mat_operations(B,D)
```

```
Second Pair
Matrix:
[[ 2  4  6]
 [ 8 10 12]
 [14 16 18]],
[[ 4  8 12]
 [16 20 24]
 [28 32 36]]
This is matrix example

(array([[ 6, 12, 18],
        [24, 30, 36],
        [42, 48, 54]]),
 array([[ -2,  -4,  -6],
        [ -8, -10, -12],
        [-14, -16, -18]]),
 array([[  8,  32,  72],
        [128, 200, 288],
        [392, 512, 648]]),
 array([[1.e-10, 1.e-10, 1.e-10],
        [1.e-10, 1.e-10, 1.e-10],
        [1.e-10, 1.e-10, 1.e-10]]))
```

Figure 12: Sample Declaration task 2

```
print('Third Pair')
mat_operations(C,B)
```

```
Third Pair
Matrix:
[[ 3  6  9]
 [12 15 16]
 [18 20 22]],
[[ 2  4  6]
 [ 8 10 12]
 [14 16 18]]
This is matrix example

(array([[ 5, 10, 15],
        [20, 25, 28],
        [32, 36, 40]]),
 array([[1, 2, 3],
        [4, 5, 4],
        [4, 4, 4]]),
 array([[  6,  24,  54],
        [ 96, 150, 192],
        [252, 320, 396]]),
 array([[1., 1., 1.],
        [1., 1., 1.],
        [1., 1., 1.]]))
```

Figure 13: Sample Declaration task 2

```
print('Fourth Pair')
mat_operations(D,E)
```

```
Fourth Pair
Matrix:
[[ 4  8 12]
 [16 20 24]
 [28 32 36]],
[[ 5 10 15]
 [20 25 30]
 [35 40 45]]
This is matrix example

(array([[ 9, 18, 27],
        [36, 45, 54],
        [63, 72, 81]]),
 array([[-1, -2, -3],
        [-4, -5, -6],
        [-7, -8, -9]]),
 array([[  20,   80,  180],
        [ 320,  500,  720],
        [ 980, 1280, 1620]]),
 array([[1.e-10, 1.e-10, 1.e-10],
        [1.e-10, 1.e-10, 1.e-10],
        [1.e-10, 1.e-10, 1.e-10]]))
```

Figure 14: Sample Declaration task 2

```
print('Fifth Pair')
mat_operations(E,A)
```

```
Fifth Pair
Matrix:
[[ 5 10 15]
 [20 25 30]
 [35 40 45]],
[[1 2 3]
 [4 5 6]
 [7 8 9]]
This is matrix example

(array([[ 6, 12, 18],
        [24, 30, 36],
        [42, 48, 54]]),
 array([[ 4,  8, 12],
        [16, 20, 24],
        [28, 32, 36]]),
 array([[  5,  20,  45],
        [ 80, 125, 180],
        [245, 320, 405]]),
 array([[5., 5., 5.],
        [5., 5., 5.],
        [5., 5., 5.]]))
```

Figure 15: Sample Declaration task 2

In figure 10 to 15 appeared as the code piece for task 2. The necessity in this assignment is to make a capacity mat_operations which has two grids or scalars esteems and its five model sets which isn't lower by (3,3) shape. Likewise, I utilized a joke code or a short code to rehearse a decent documentation. At that point the condition is to show every grid if the boundary 10 is scalar it will print the scalar qualities. From that point forward, it will decide whether the estimations of the frameworks are suitable for the tasks, for example, expansion, deduction, component shrewd increase, and to find a particular solution for the component shrewd of divison alpha is utilized to show a little worth and to forestall getting endless worth. In the event that the condition isn't met it should re-visitation of its blunder message

IV. Conclusion

In this lab action presented an idea about how the essential activities of frameworks work and it was appeared in the outcomes that networks can be decide as per their shape, size and rank. Additionally it tends to be decide whether the network is vacant, square, a personality, ones, or then again zero grid. Followed by its fundamental activities, for example, getting its aggregate, distinction, component shrewd of product,and component savvy of the remainder utilizing distinctive NumPy's present functions.The use of this action is to show various depictions of the lattices and its fundamental tasks to accomplish it I utilized NumPy's presents capacities which is simpler to comprehend. Likewise, it very well may be utilized to deal with different direct conditions and to a framework straight conditions

References

[1] D.J.D. Lopez. “Adamson University Computer Engineering Department Honor Code,” AdU-CpE Departmental Policies, 2020.