Adamson University
College of Engineering
Computer Engineering Department

Linear Algebra

Laboratory Activity No. 6

Matrices

Submitted by:
Reyes, Carl Vincent G.

Instructor:
Engr. Dylan Josh D. Lopez

November 25, 2020

I. Objectives

This laboratory activity aims to teach the student to be familiar with matrices and their relations to different equations and as well as perform basic matrix operations and be able to put it in python wisely.

II. Methods

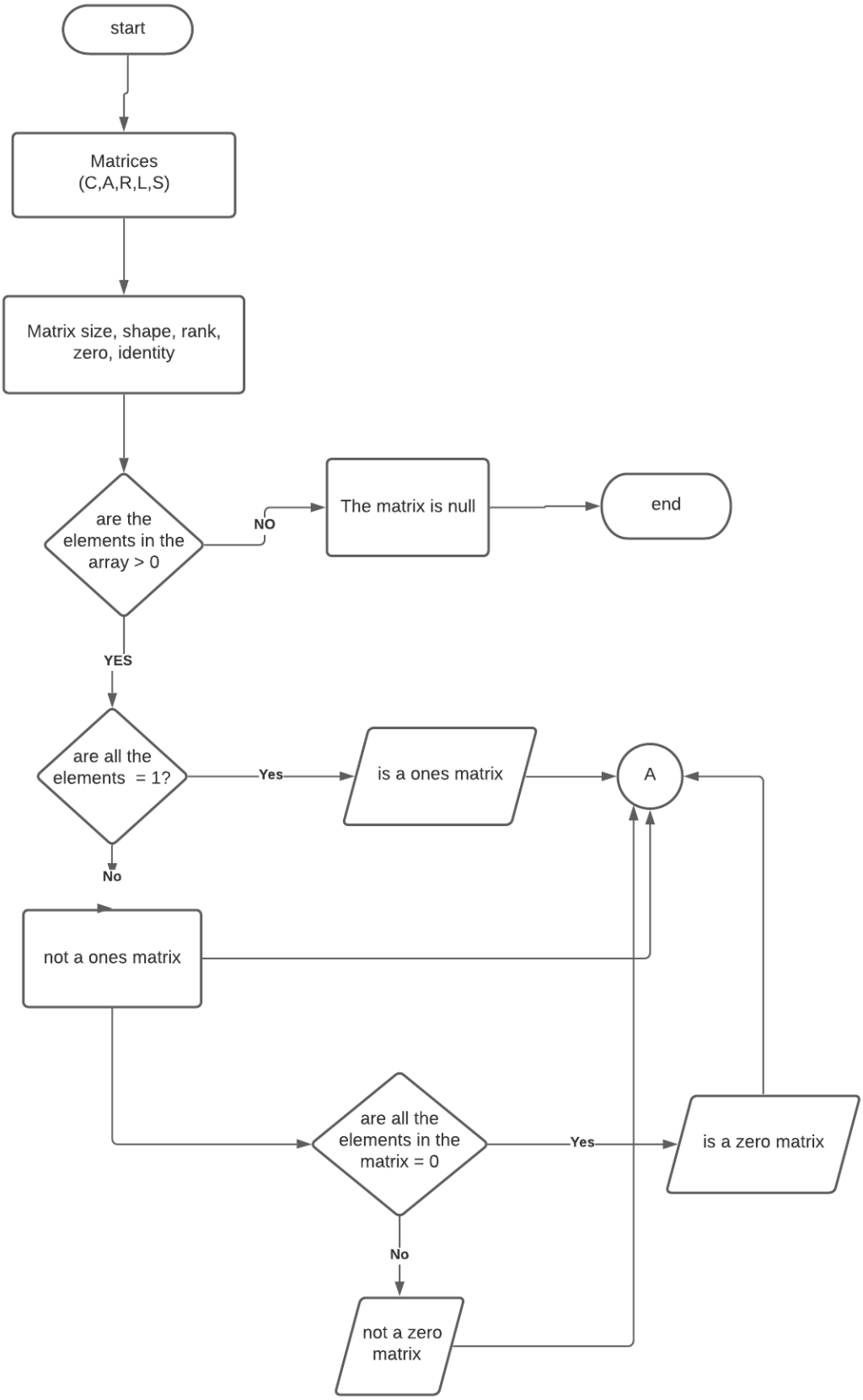

Figure 1: flowchart for task 1 part 1

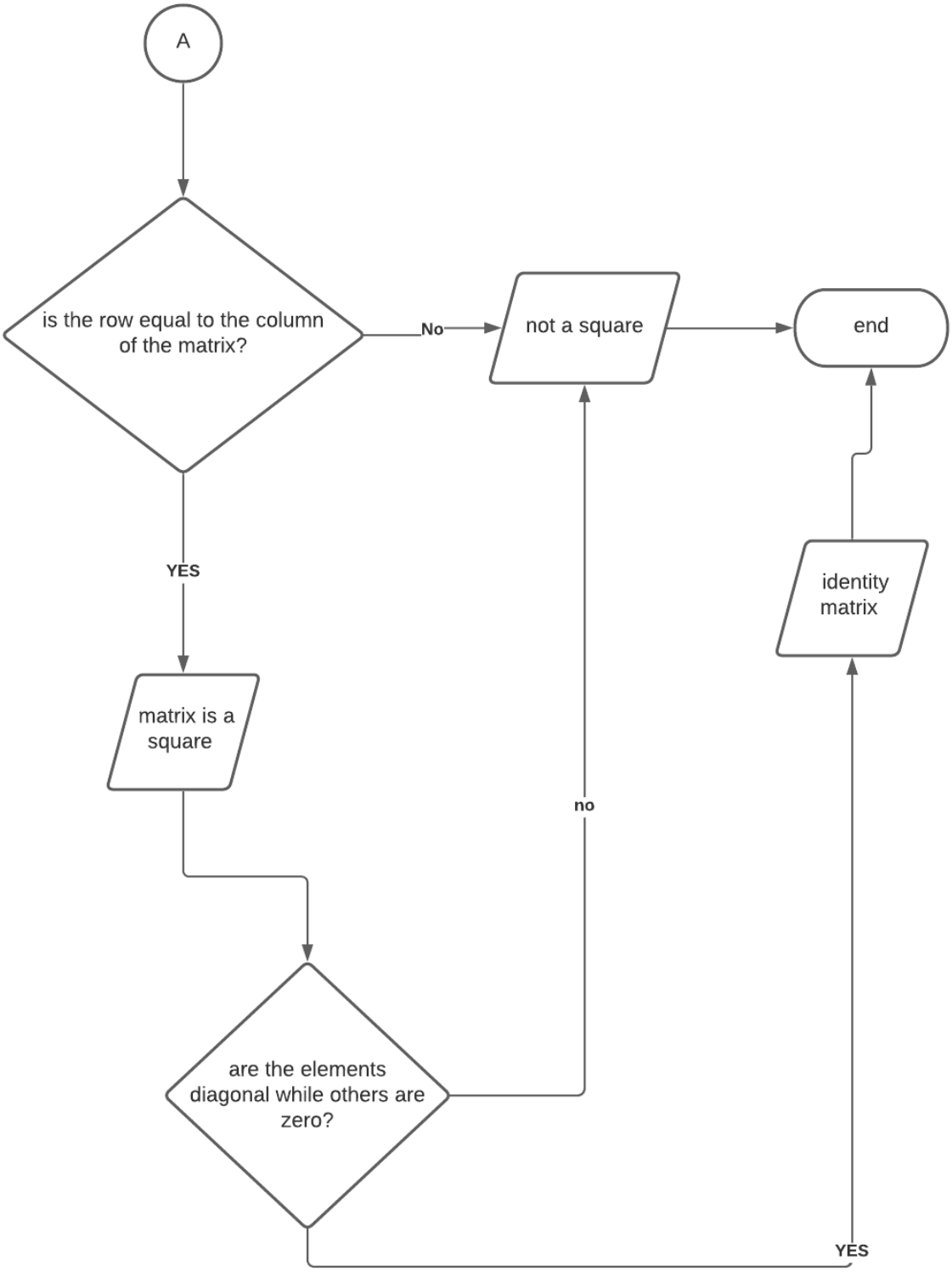

Figure 2: flowchart task 1 part 2

As seen from the figure 1 and 2 above, the flowchart describes how the code will work and what will be the first steps it will do until it reaches its final output. First, the elements inside the matrices defined, and the codes for their sizes, shapes, ranks, is It zero, is it ones, and is it identity codes takes place. How it works is also in the flowchart which can

be seen above, for example is “are the elements in the matrix > 0?” is inside the decision making shape, and if yes it continues to the next question but if it is not, the result will appear and it will print “The matrix is null” then will terminate or end the loop.

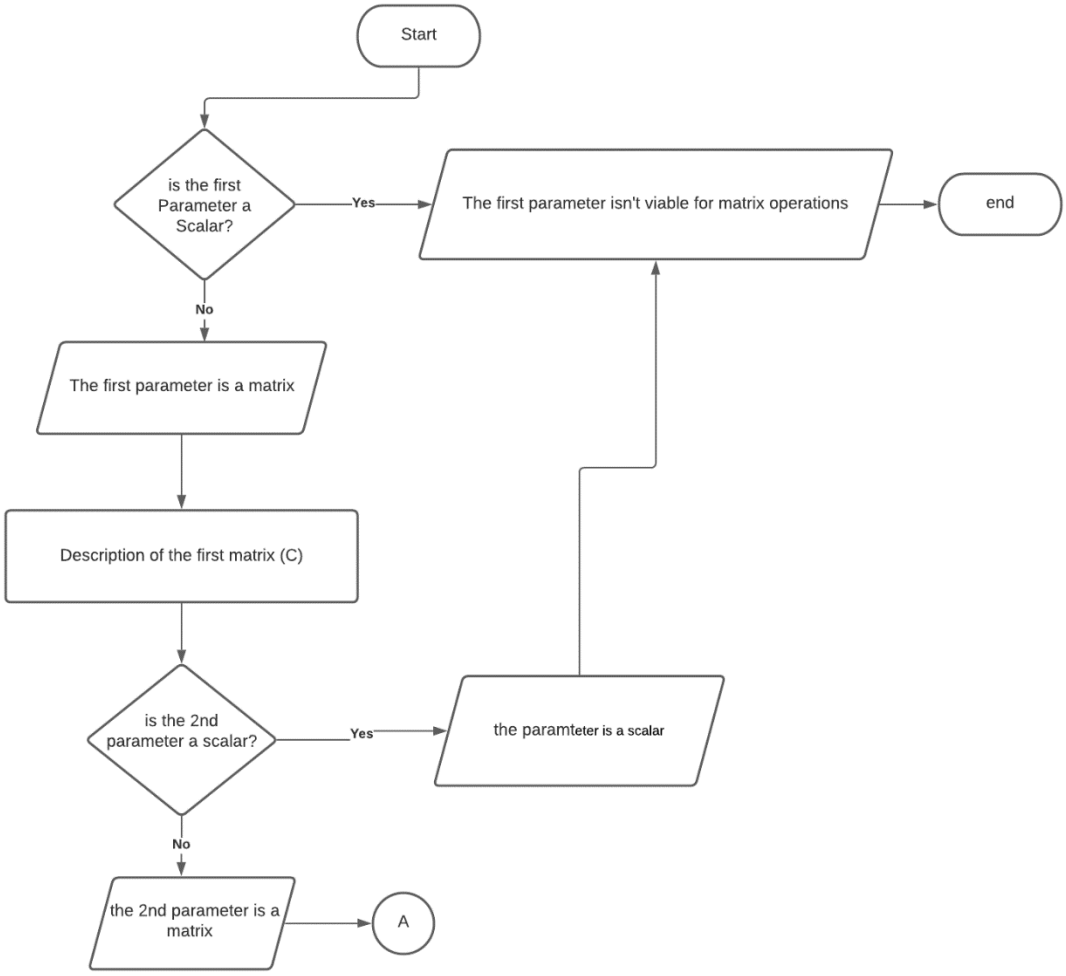

Figure 3: task 2 flowchart part 1

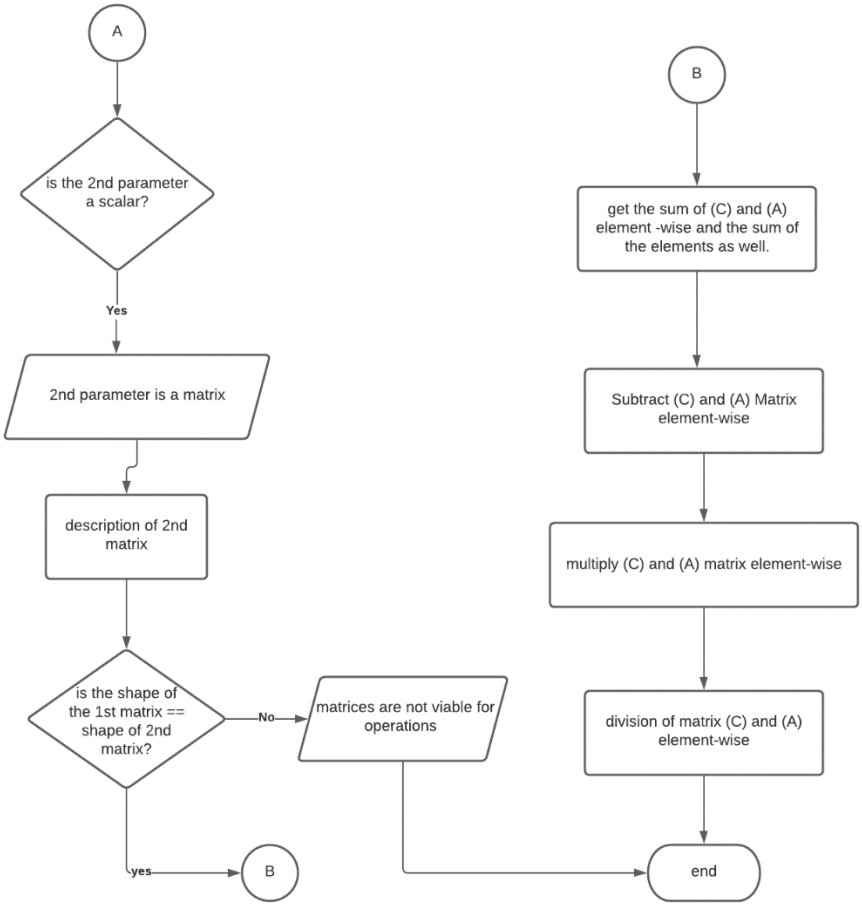

Figure 4:task 2 part 2

As seen from the figure above the difference of the 2nd task to the 1st task is that it has a pair, and it asks first if the parameters of the matrix are scalar and if not it is not viable for operation and if yes then is prints the description of the matrix. Then after that it asks the 2nd matrix if its parameters are scalar then the process repeats, then both matrices will be multiplied, divided, difference and the sum element-wise talking will be printed out. Then the loop will be terminated.

```
In [14]: def mat_operations(matrix):
             if matrix.size > 0:
                 is_ones = True if np.all((matrix == 1)) else False
                 is_square = True if matrix.shape[0] == matrix.shape[1] else False ##checks if the matrix is a square
                 is_zero = True if np.all((matrix == 0)) else False ##checks if the matrix is equals to 0
                 is_identity = True if matrix.shape[0] == matrix.shape[1] and np.allclose(matrix, np.eye(matrix.shape[0])) else False
                 print(f'Matrix:\n{matrix}\n\nShape:\t{matrix.shape}\nRank:\t{matrix.ndim}\nIs Square: {is_square}\n')
                 print (f' is the matrix a ones matrix? {is_ones}\n')
                 print (f' is the matrix a zero matrix? {is_zero} \n')
                 print (f' is the matrix an identity matrix? {is_identity} \n' )
             else:
                 print('Matrix is Null')## Function area
```

Figure 5: codes used in task 1

As seen in figure 3, these are the codes used in order to call the function mat_operations which checkes all the elements of the matrix if they are greater than 0, if it is

equals to 1, is it a square, and is it an identity matrix, this is just the first part in completing the task. The codes above also checks the size, rank and the shape of the matrix.

```
]: C = np.array([
       [2,1,3],
       [1,2,3],
       [4,5,9]])

   A = np.array([
       [5,6,7,8],
       [1,2,3,4],
       [1,1,1,1],
       [0,1,0,1]])

   R = np.array([
       [3,2,0],
       [2,0,0],
       [1,0,1],
       [0,0,0]])
   L = np.array([
       [3,1,2],
       [2,1,4],
       [1,4,3],
       [1,5,0]])
   S = np.array([
       [0,1,0,1,0],
       [1,1,1,1,1],
       [2,3,2,3,2],
       [4,3,2,1,0]

   ])


       ## Matrix declarations
```

Figure 6: declaration of matrices

Figure 4 shows the matrix that the programmer used, with these values, the function mat_operation will be able to call them and use them as determinants for the code to work. As observed, the matrix is not that simple, because according to the instructions the shape of the matrix must not be lower than (3,3), which is met by the programmer.

```
[25]: def mat_operations(C,A):
          alpha = 10**-10
          if np.isscalar(C):
              print("The first parameter is a scalar")
          else:
              print(f'1st matrix:\n {C}\n'
              f'The shape of the 1st matrix is: {C.shape}\n'
              f'The size of the 1st matrix is: {np.product(C.shape)}\n'
              f'The rank of the 1st matrix is: {C.ndim}')
              if C.size > 0:
                  is_ones = True if np.all((C == 1)) else False ##checks if the elements are 1
                  is_zero = True if np.all((C == 0)) else False ## checks if the elements are 0
                  is_square = True if C.shape[0] == C.shape[1] else False ##checks if the elements are square
                  is_identity = True if C.shape[0] == C.shape[1] and np.allclose(C, np.eye(C.shape[0])) else False
                  ## checks if the matrix is a square, identity, ones or zero/s
                  print(f'Is the matrix a ones matrix?: {is_ones}\n'
                  f'Is the matrix a zero matrix?: {is_zero}\n'
                  f'Is the matrix a square?: {is_square}\n'
                  f'Is the matrix an identity matrix?: {is_identity}\n\n'
                  )
              else:
                  print("The Matrix is Null or Empty")

          if np.isscalar(A):
              print("The first parameter is a scalar")
          else:
              print(f'1st matrix:\n {A}\n'
              f'The shape of the 1st matrix is: {A.shape}\n'
              f'The size of the 1st matrix is: {np.product(A.shape)}\n'
              f'The rank of the 1st matrix is: {A.ndim}')
              if A.size > 0:
                  is_ones = True if np.all((A == 1)) else False ##checks if the elements are 1
                  is_zero = True if np.all((A == 0)) else False ## checks if the elements are 0
                  is_square = True if A.shape[0] == A.shape[1] else False ##checks if the elements are square
```

Figure 7:Codes for task 2 part 1

```
            is_zero = True if np.all((A == 0)) else False ## checks if the elements are 0
            is_square = True if A.shape[0] == A.shape[1] else False ##checks if the elements are square
            is_identity = True if A.shape[0] == A.shape[1] and np.allclose(A, np.eye(A.shape[0])) else False
            ## checks if the matrix is a square, identity, ones or zero/s
            print(f'Is the matrix a ones matrix?: {is_ones}\n'
            f'Is the matrix a zero matrix?: {is_zero}\n'
            f'Is the matrix a square?: {is_square}\n'
            f'Is the matrix an identity matrix?: {is_identity}\n\n'
            )
        else:
            print("The Matrix is Null or Empty")

    if np.isscalar(C) == False and np.isscalar(A) == False:
        if (C.shape == A.shape):
            print(f'The sum of the matrices is: \n{np.sum(np.add(C,A))}\n'
                f'\nThe difference of the matrices is: \n{np.diff(np.subtract(C,A))}\n'
                f'\nThe element-wise multiplication of the matrices is: \n{np.multiply(C,A)}\n'
                f'\nThe element-wise division of the matrices is: \n{np.divide(C,A+alpha)}\n')
        else:
            print("The matrices aren't viable for operation\n")
    else:
        print("The parameters aren't viable for matrix operation\n")
```

Figure 8: codes used for task 2 part 2

Based on the figure 5 and 6, these are the codes used by the programmer in order to make the function for the second task. First thing to noticed is that there is an alpha after the if statement, this will be used later for the division, then it first asks if the matrix is a scalar, then after that, similar to the 1st task, it asks for the size, shape, rank, is the matrix an identity, is it a square, is it one and is It a zero matrix. In addition, the task now have addition, subtraction, multiplication and division, which was also in the scope of the laboratory 6. Then lastly, it prints them.

III. Results

```
In [12]: mat_operations(C)## Test Areas

         Matrix:
         [[2 1 3]
          [1 2 3]
          [4 5 9]]

         Shape:  (3, 3)
         Rank:   2
         Is Square: True

          is the matrix a ones matrix? False

          is the matrix a zero matrix? False

          is the matrix an identity matrix? False
```

Figure 9: Result from task 1

```
In [13]: mat_operations(A)

         Matrix:
         [[5 6 7 8]
          [1 2 3 4]
          [1 1 1 1]
          [0 1 0 1]]

         Shape:  (4, 4)
         Rank:   2
         Is Square: True

          is the matrix a ones matrix? False

          is the matrix a zero matrix? False

          is the matrix an identity matrix? False
```

Figure 10: Result from task 1 part 2

Figure 7 shows the result from the code that the programmer created, there were no errors present and the output that was shown is accurate and is aligned to what the instructions said. There are no problems when it comes to running the code meaning that the programmer used the right codes and condition in making the said program/task.

```
matrices pair 1:

1st matrix:
 [[2 1 3]
 [1 2 3]
 [4 5 9]]
The shape of the 1st matrix is: (3, 3)
The size of the 1st matrix is: 9
The rank of the 1st matrix is: 2
Is the matrix a ones matrix?: False
Is the matrix a zero matrix?: False
Is the matrix a square?: True
Is the matrix an identity matrix?: False
```

Figure 11: Result for task 2 part 1

```
1st matrix:
 [[5 6 7 8]
 [1 2 3 4]
 [1 1 1 1]
 [0 1 0 1]]
The shape of the 1st matrix is: (4, 4)
The size of the 1st matrix is: 16
The rank of the 1st matrix is: 2
Is the matrix a ones matrix?: False
Is the matrix a zero matrix?: False
Is the matrix a square?: True
Is the matrix an identity matrix?: False


The matrices aren't viable for operation
```

Figure 12: Result for task 2 part 2

```
The difference of the matrices is:
[[ 1 -3]
 [-1 -3]
 [-4  2]
 [-4  5]]

The element-wise multiplication of the matrices is:
[[9 2 0]
 [4 0 0]
 [1 0 3]
 [0 0 0]]

The element-wise division of the matrices is:
[[1.         2.         0.        ]
 [1.         0.         0.        ]
 [1.         0.         0.33333333]
 [0.         0.         0.        ]]
```

Figure 13:Result for task 2 part 3

As seen from the figures above, there were no errors again, and the programmer was able to do the task that was given upon the instruction, the difference between this task and the 1st task is that there are pairings in this and it asks where the matrix is a scalar. Then the instructions also included vector operations which is not done in the first task.

IV. Conclusion

The laboratory activity helped the student to be able to identify some real-time applications and be able to manage to do the tasks given to them. This activity also shows that matrices can be solved in python with the use of operations that we are familiar with during our earlier years of studies. This also gives the student more knowledge about NumPy and expand their learnings by introducing them to new functions and successfully be able to apply it.

When it comes to solving some agricultural problem, I think this activity will be able to to give an organized matter for the farmers when it comes to planting, because if you were to imagine the matrix as a plane with some crops on them, being able to identify and record each and every one of them will probably lessen their time on going in them one by one. They could use It as a marker or specifying a location in their fields.

References

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

GITHUB REPO:

https://github.com/ReyesCarl/LA-LAB6