Linear Algebra

Laboratory Activity No. 6

**Matrices**

*Submitted by: Instructor:*

Parco, Jovian Asher G. Engr. Dylan Josh D. Lopez

November 28, 2020

# Objectives

This laboratory activity aims to implement matrix operations and matrix identities using Python’s Numpy module.

# Methods

In this laboratory activity, the practices consist of creating a function that describes a matrix identity such as shape() function, size() function, ndim() function and, combinations of functions to identify different identity, also matrix operations such as numpy.sum() & numpy.diff() and element wise operation such as numpy.multiply() & numpy.divide().

# Results

The complete repository for the activity is available at Github (). The lab activity consists of 2 parts; the first part is creating a function that describes a matrix, while the second part takes two arguments (matrix/scalar) and run it through different types of matrix operation.



Figure Function for Part 1

For part 1 of the lab activity, the researcher created a function called “mat\_desc” (Figure 1); it is a simple function that takes one parameter that is a system of linear equation in a matrix form and identifies its shapes, rank, size, different identities such as upper diagonal, lower diagonal and many more. For the code in the second line, the researcher created an “if / else” condition where the matrix checks if the size of the matrix is more than one by using the size() function, if the matrix is confirmed to have more than one size of matrix it run to the next code if not the function outputs “Matrix is Null” & terminates the program, for the third line of the code is also an “if / else” where the matrix is checked if the matrix dimension is two using ndim() function and check if the shape is a square matrix using the shape() function, if the function failed to meet the argument it goes to the else statement where it only outputs its matrix shape, matrix dimension, and all the matrix identities are set to false, but if the statement is met it runs through all the arguments that describe matrix identity and form using different combinations of Numpy functions.

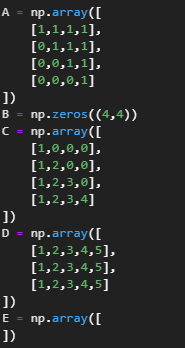


Figure Sample Matrices for part 1

Using the function that the researcher created it describes a set of matrices (Figure 2 ) that have different identities and form, which if the function takes one argument in the set of matrices (Figure 2) it outputs all the following:

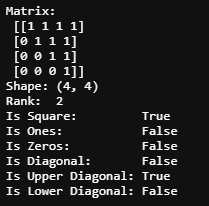


Figure Output for Part 1 Using the “A” Arguments

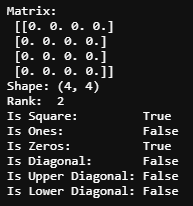


Figure Output for Part 1 Using the “B” Arguments

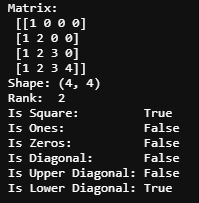


Figure Output for Part 1 Using the “C” Arguments

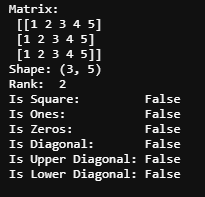


Figure Output for Part 1 Using the “D” Arguments



Figure Output for Part 1 Using the “E” Arguments

Using the sample matrices, the researcher describes all the matrices with the preview of the matrix in the top to confirm it manually.

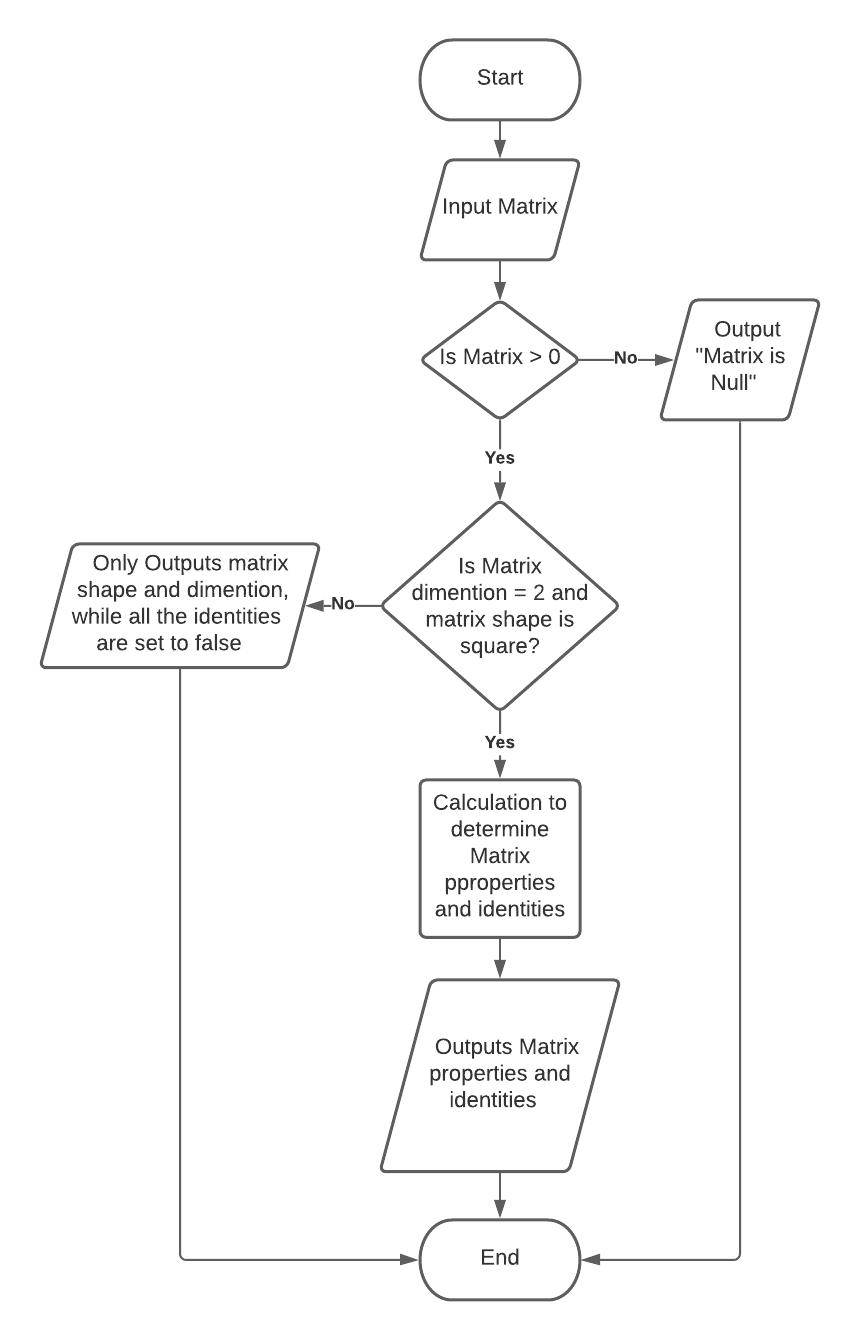


Figure Flowchart for Part1

The researcher created a simple flowchart to describe or summarize the function created.

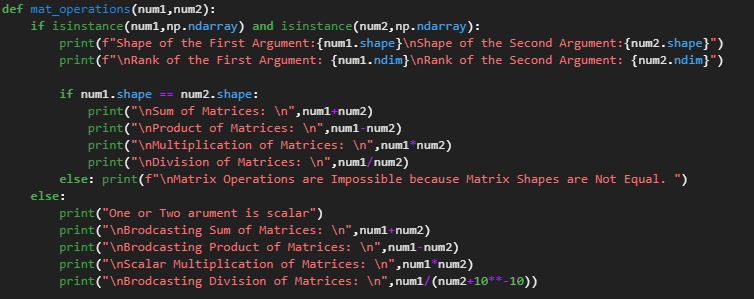


Figure 9 Function for Part 2

For part 2 of the lab activity, the researcher created another function (Figure 8) this time; it takes two parameters that can be either matrices or scalars and will output matrix operation such as the sum of the matrix, difference of matrix, element-wise multiplication of the matrix, and element-wise division of the matrix. The second line of the researcher created an “if / else” statement where it uses the isinstance() function that checks if the type of the inputted argument is the same array type or matrix and will run a simple function describes the matrix. Simultaneously, if it is not the same, it will output “One or Two arguments is scalar” but still run all the operation as a broadcasting operation. If we continue, there is one last “if / else” statement, where it checks the shape of the matrix if it is equal using the shape()function; if not, it will output “Matrix Operations are Impossible because Matrix Shapes are Not Equal.” However, if it is equal, it will run all the matrix operations that are mention above.

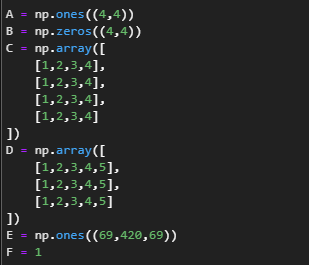


Figure 10 Sample Matrices and Scalar for part 2

Using the function that the researcher created it describes a set of matrices and scalar (Figure 8 ) that have different identities and form, which if the function takes two arguments in the set of matrices or scalar (Figure 8).

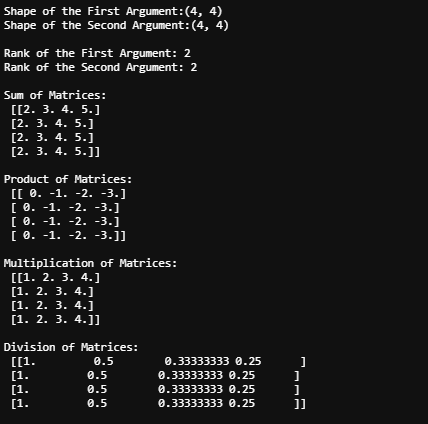


Figure Output for Part 2 Using the “A,C” Arguments

For the output(Figure 11) it shows that the inputted argument in the function is matrices, which first describes both the matrix by shape and dimensions and proceed to do al matric operations.

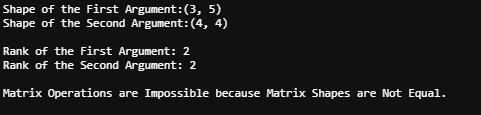


Figure Output for Part 2 Using the “D,C” Arguments

For the output(Figure 12), it shows that both inputted arguments are matrices, but it’s not the same shapes, so matrix operation is not possible.

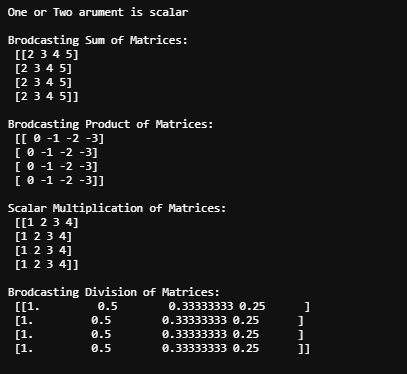


Figure Output for Part 2 Using the “F,C” Arguments

The last unique output (Figure 13) shows one matrix and one scalar as the inputted argument in the function; it proceeded to do a broadcasting matrix instead of a matrix operation.

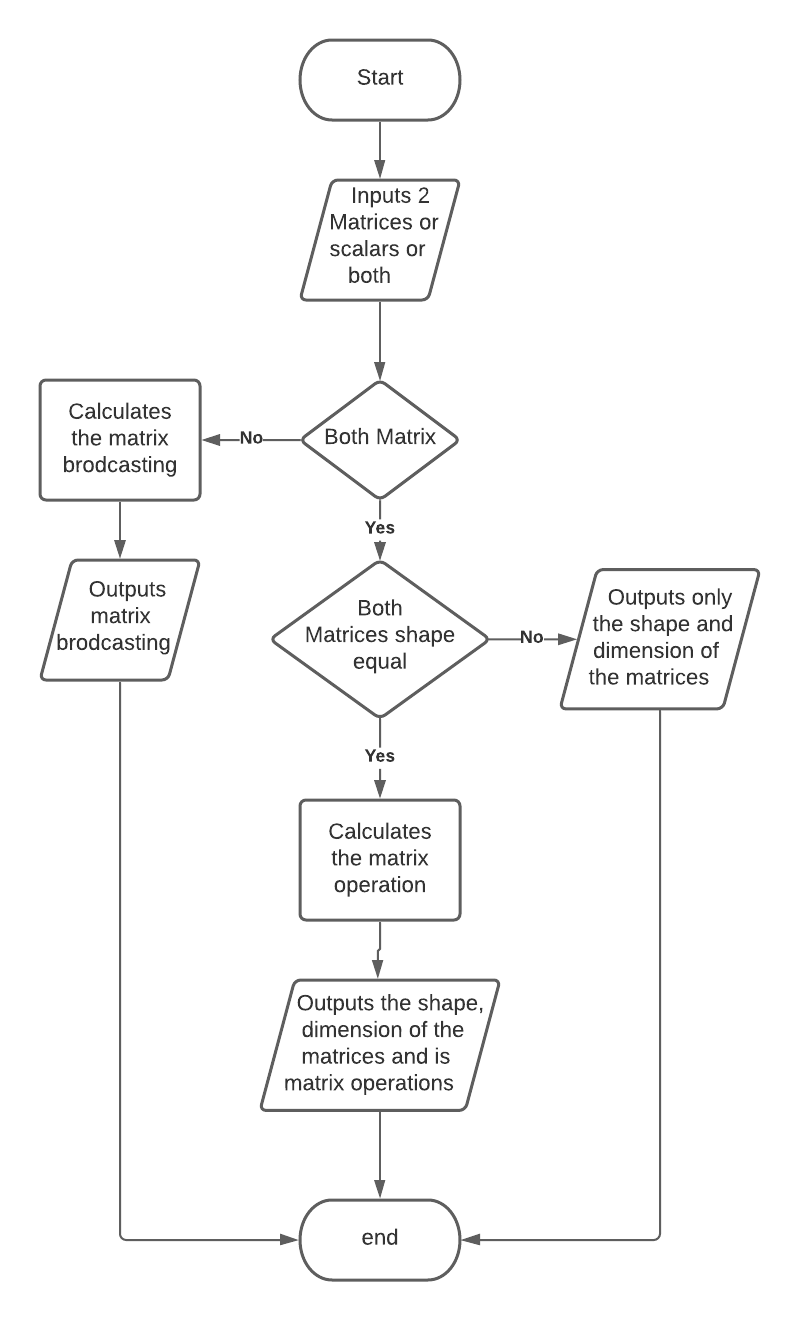


Figure Flowchart for Part2

The researcher created a simple flowchart to describe or summarize the function created.

# Conclusion

The researcher used Numpy functions and the “if \ else” statement to create two functions that describe a matrix also calculates the matrix operation or matric broadcasting, the researcher also learns that Discrete mathematics subject is important to understand and efficiently created a function. Also, matrix operations can be used in farming by developing an algorithm that uses matrix operations or matrix identity to plot out the land and maximize the given space.**References**