LAB 05: Matrix Handling

CS211 – Data Structures and Algorithms
Usman Institute of Technology
Fall 2019

- **How to submit:**
  - Create an account on http://www.turnitin.com/ as a Student (if you don't have already)
  - Use following information at time of sign-up

    CS Section A
    - Class ID: 22664649
    - Enrollment Key: DSFALL19CSA

    CS Section B
    - Class ID: 22664651
    - Enrollment Key: DSFALL19CSB

A. For this lab we will be re-using the content of Lab 02.

Create a class Matrix which takes two parameters to initialize: rows and cols and write functions in Python whose parameters and return value are given below.

1. Add a constructor of the class must initialize a list containing rows * cols element. All element must be declared 0 by default. You can use the following code to initialize the elements.

   data = [0 for j in range(cols*rows)]

```
class Matrix:
    def __init__(self, rows, cols):
        // your code goes here
```

2. Add a function **SetValues** which takes three prameters i, j and v, for row, column, and value respectively. The function set the value at i^{th} row and j^{th} column. The function is supposed to convert these two-dimension value into a linear dimension.

   The following equation can be used for conversion:

   Location = i * R + j (memory addresses have been omitted from the equation)

```
    def SetValues():
        // your code goes here
```

3. Add a function **GetValue()** which takes two parameters **i** and **j** and returns the value for i^{th} row and j^{th} column. You have to convert two dimensional values into a single dimension value, as discussed in above question.

4. The class should also have a function **PrintValues**() that print the values of the array in Row and Column format.

```
    def PrintValues():
        // your code goes here
```

5. Add a function **MultValues()** that takes two parameters Matrix A and Matrix B and returns a matrix containing multiplication of two given matrices.

```
    def MultValues(array1, array2):
        // your code goes here
```

6. Add a function **transpose()** that returns a matrix containing the transpose of the matrix.

```
    def transpose():
        // your code goes here
```

Example: Matrix A = $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$

Transpose = $\begin{bmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{bmatrix}$

B. Create a Python script by using the following functions by importing NumPy Library of Python.

1. Create a Numpy Array.

```
import numpy as np

array1 = np.array([[1,2,3,4],[5,6,7,8]], dtype=np.int64)
print(array1)
```

2. Create an array of ones

```
x = np.ones((3,4),dtype=np.int64)
print(x)
```

3. Create an array of zeros

```
y = np.zeros((2,3,4),dtype=np.int16)
print(y)
```

4. Create an array with random values

```
array2 = np.random.random((2,2))
print(array2)
```

5. Create a full array

```
array3 = np.full((3,3),7)
print(array3)
```

6. Create an identity matrix

```
array4 = np.identity(3,dtype=np.int64)
print(array4)
```

7. Find sum of two matrices

```
add = np.add(x,y)
print(add)
```

8. Find difference of two matrices

```
diff = np.subtract(x,y)
print(diff)
```

9. Find product of two matrices

```
mult = np.multiply(x,y)
print(mult)
```

10. Find division of two matrices

```
div = np.divide(y,x)
print(div)
```

11. Find remainder of two matrices

```
rem = np.remainder(y,x)
print(rem)
```

12. Check if two arrays are equal

```
result = np.array_equal(x,y)
print(result)
```