Name: ADARSH KUMAR	Date: 14/08/2022
SRN: PES2UG20CS016	Section: B

CODE:

```
#This weeks code focuses on understanding basic functions of pandas and numpy
#This will help you complete other lab experiments


# Do not change the function definations or the parameters
from array import array
import numpy as np
import pandas as pd

#input: tuple (x,y)    x,y:int
def create_numpy_ones_array(shape):
    #return a numpy array with one at all index
    array=None
    #TODO
    array = np.ones(shape)
    return array

#input: tuple (x,y)    x,y:int
def create_numpy_zeros_array(shape):
    #return a numpy array with zeros at all index
    array=None
    #TODO
    array=np.zeros(shape)
    return array

#input: int
def create_identity_numpy_array(order):
    #return a identity numpy array of the defined order
    array=None
    #TODO
    array=np.identity(order)
    return array

#input: numpy array
def matrix_cofactor(array):
    #return cofactor matrix of the given array
    array =  np.matrix.getH(array).transpose()
    return array
```

```
#Input: (numpy array, int ,numpy array, int , int , int , int , tuple,tuple)
#tuple (x,y)    x,y:int
def f1(X1,coef1,X2,coef2,seed1,seed2,seed3,shape1,shape2):
   #note: shape is of the forst (x1,x2)
   #return W1 x (X1 ** coef1) + W2 x (X2 ** coef2) +b
   # where W1 is random matrix of shape shape1 with seed1
   # where W2 is random matrix of shape shape2 with seed2
   # where B is a random matrix of comaptible shape with seed3
   # if dimension mismatch occur return -1

   np.random.seed(seed1)
   w1 = np.random.rand(shape1[0], shape1[1])
   np.random.seed(seed2)
   w2 = np.random.rand(shape2[0], shape2[1])
   np.random.seed(seed3)
   shape3 = (np.shape(w1)[0],np.shape(X1)[1])
   B = np.random.rand(shape3[0],shape3[1])
   if not(shape3 == (shape2[0], np.shape(X2)[1])):
      return -1
   if not(shape1[1] == np.shape(X1)[0] and shape2[1] == np.shape(X2)[0]):
      return -1
   ans = np.matmul(w1, X1 ** coef1) + np.matmul(w2, X2 ** coef2) + B
   return ans

def fill_with_mode(filename, column):
   """
   Fill the missing values(NaN) in a column with the mode of that column
   Args:
      filename: Name of the CSV file.
      column: Name of the column to fill
   Returns:
      df: Pandas DataFrame object.
      (Representing entire data and where 'column' does not contain NaN
values)
      (Filled with above mentioned rules)
   """
   df=pd.read_csv(filename)
   mode = df[column].mode()[0]
   df[column].fillna(mode,inplace=True)
   return df

def fill_with_group_average(df, group, column):
   """
   Fill the missing values(NaN) in column with the mean value of the
   group the row belongs to.
   The rows are grouped based on the values of another column
```

```
    Args:
        df: A pandas DataFrame object representing the data.
        group: The column to group the rows with
        column: Name of the column to fill
    Returns:
        df: Pandas DataFrame object.
        (Representing entire data and where 'column' does not contain NaN
values)
        (Filled with above mentioned rules)
    """
    df[column].fillna(df.groupby(group)[column].transform('mean'),inplace=True)
    return df

def get_rows_greater_than_avg(df, column):
    """
    Return all the rows(with all columns) where the value in a certain 'column'
    is greater than the average value of that column.

    row where row.column > mean(data.column)

    Args:
        df: A pandas DataFrame object representing the data.
        column: Name of the column to fill
    Returns:
        df: Pandas DataFrame object.
    """
    df=df.loc[df[column] > df[column].mean()]
    return df
```

Output Screenshot:

```
D:\Course\Semester 5th\MI\MI Lab\Week1>python SampleTest.py --SRN PES2UG20CS016
Test Case 1 for create_numpy_ones_array PASSED
Test Case 2 for create_numpy_zeros_array PASSED
Test Case 3 for create_identity_numpy_array PASSED
Test Case 4 for matrix_cofactor PASSED
Test Case 5 for f1 PASSED
Test Case 6 for f1 PASSED
Test Case 7 for the function fill_with_mode PASSED
Test Case 8 for the function fill_with_group_average PASSED
Test Case 9 for the function get_rows_greater_than_avg PASSED
```