# PDS\_Assignment Inference document

1. create the following two dimensional arrays

[[1., 0., 0., 0., 0.],

[0., 1., 0., 0., 0.],

[0., 0., 1., 0., 0.],

[0., 0., 0., 1., 0.],

[0., 0., 0., 0., 1.]]

**Inference**: This is identity matrix so used numpy standard identity() function to generate the array matrix

1. [[1, 0, 0, 0],

[0, 2, 0, 0],

[0, 0, 3, 0],

[0, 0, 0, 4]]

**Inference**: This is diagonal matrix so used numpy standard diag() function to generate the array matrix

1. Create a checkerboard pattern in an array using row and col slicing

**Inference:** Designed 8x8 zero matrix followed by slice method to assign alternate rows with 1

cb[::2,::2] = 1

cb[1::2,1::2] = 1

1. Compute the no of bytes occupied in the memory for an array of numbers from 1 to 10

**Inference:** used attribute numpy. Nbytes

**5. img=[[200,210,209],[213,0,214],[214,215,217]]**

**Given a snap shot of the image, estimate the centre pixel with mean, median values.**

**Inference:** step1: flatted the array for using standard numpy functions np.mean() and np.median(), finally assigned outputs to 1,1 In overall matrix

1. Create a Pandas series to store the marks of students and filter the marks >50,>70,>90

**Inference:** used common filter option like here marks is standard numpy array marks[marks>50]

1. Create data frame given below to store student info from the numpy arrays defined

**Inference: pd.DataFrame({<Passed dictionary as key and values}}**

Other one liner functions

print("#1.Filter the age")

print(df['age'])

print("\n\n#2.filter age,cgpa")

print(df[['age', 'cgpa']])

print("\n\n#3.Filter age>22")

print(df['age'][df['age']>22])

print("\n\n#4.names starts with j")

print(df['name'][df['name'].str.startswith('j')])

print("\n\n#5.names contains 'n'")

print(df['name'][df['name'].str.contains('j')])

1. Perform left, right and inner join on the dataframes

**Inference:** by modifying how='left' parameters performed all three merges.

1. 9. Apply sigmoid function to the array [2,5,6]
   1. 1.define sigmoid function
   2. 2.create the np array
   3. 3.Apply the function to the array

**Inference**: used lambda x: 1/(1 + np.exp(-x)) function and passed the array to lambda function

Out: array([0.88079708, 0.99330715, 0.99752738])