1) # random matrix generated

~~A~~

A = np.random.randint(0, 100, (4,4))

generates a random matrix of int values between 0 and 99 of shape 4x4.

$$\begin{bmatrix} 1 & 2 & 3 & 4 \\ 4 & 2 & 3 & 1 \\ 1 & 4 & 1 & 2 \\ 3 & 2 & 4 & 1 \end{bmatrix}$$

~~call~~ A_inv = inv(A)

~~d~~ det = np.linalg.det(A)

np.linalg.inv(A) # gives the inverse of a matrix

adj = inv x det

I4 = np.eye(4) # np.eye(n) gives the nxn identity matrix

If (A x A_inv == I4)

print A x A^-1 = I4

#np.allclose is used to compare ~~two~~ two ~~tab~~ two arrays ~~elet~~ element

3) $m=8$, $n=6$

$8-6=2$

e^{2x}

~~Taylor Series~~

def taylor_exp # defining a function taylor_exp that takes two arguement x and n. This calculates the Taylor approx. of e^x to 6 terms using for loop and np.math. factorial function.

- def taylor_prod ~~that~~ takes x and n. This function calculate e^{2x} by multiplying the Taylor approximation of e^{8x} and e^{-6x} using Taylor_exp function.

- Using np.arange to create an array of x values from -1 to 1 with interval 0.01 using np.arange function.

- ~~Ge~~ Then we create three arrays of y values for actual and approximated e^{2x} using numpy's exp function, taylor_exp function, and taylor_prod function.

- Then ~~we plot~~ from line 22 ~~and~~ to 29, this part of the code is used to plot. we plot three arrays of y values on the same graph using plt.plot function.