Project 1: Basic operations with matrices in Python

Goals: To introduce matrix input functions and basic operations with matrices in Python. The matrix operations to be studied include matrix addition and subtraction, scalar product, matrix product and element wise matrix product in Python, matrix concatenation, and selecting submatrices.

Python commands used: len, np.linspace, np.shape, np.max, np.min, np.mean, np.sum, np.random.randint (or np.random.rand), .T, np.dot, *, np.multiply, np.eye, np.zeros, np.ones, np.diag, np.spdiags.

Tasks to do:
Matrix Operations

1. Determine the length of a matrix or vector.
2. Add B as the fourth row of A to create C.
3. Create D from rows 2, 3, 4 and columns 3, 4 of C.
4. Transpose D to create E.
5. Check the size of E.
6. Create equally spaced vectors.
7. Find the max and min in each column of A.
8. Find the max and min in each row of A.
9. Calculate mean and sum in each column and row of A.
10. Create random matrices F and G.
11. Perform scalar multiplication, addition, subtraction, and element-wise multiplication on F and G.
12. Check the size of F and A, then perform matrix multiplication if dimensions are compatible.
13. Generate the following special matrices:
    - Identity matrix of size 3×3.
    - Zero matrix of size 5×3 and one matrix of size 4×2.
    - Diagonal matrix S with diagonal elements [1, 2, 7].
    - Extract diagonal elements from a random 6×6 matrix.
    - Create a sparse diagonal matrix and convert it to a dense matrix.

Project 2: Matrix operations and image manipulation

Goals: To apply elementary matrix operations such as matrix addition, subtraction, and element wise multiplication to image manipulation. We will look at the lightening/darkening of images, selecting subimages, changing contrast, rotating and flipping images.

To get started:
• Create a new Python script and save it as lab02.py
• Download the image file einstein.jpg and save it in your working directory

Python commands used: imageio.imread, matplotlib.pyplot.imshow, numpy.uint8, numpy.asarray, numpy.flip, numpy.shape, numpy.transpose, numpy.zeros, numpy.rot90, numpy.floor, numpy.ceil, numpy.round, numpy.fix

INTRODUCTION

An image in a computer memory can be stored as a matrix with each element of the matrix representing a pixel of the image and containing a number (or several numbers) which corresponds to the color of this pixel. If the image is a color image, then each pixel is characterized by three numbers corresponding to the intensities of Red, Green, and Blue (the so called RGB color system). If the image is a grayscale image, then only one number for the intensity of gray is needed. The intensity of each color typically ranges from 0 (black) to 255 (white). This allows for the representation of over 16 million colors in the RGB system which is more than a human eye can distinguish. The images stored in this way are called bitmaps (there are also images which do not use pixels, such as vector graphics, but we will not talk about those here). If you have ever used an image editor before, you may know that there are several fairly standard operations which you can perform with images. In this project we will replicate some of them using operations on matrices. By the end of the project you will understand how graphic editors process images to achieve different visual effects. As it turns out, many of these operations are accomplished by manipulating the values of the pixel colors.

Tasks to do:

1. **Load an Image:**This can be done by using the Python command imread. This command allows Python to read graphic files with different extensions. The output is an array with the dimensions equal to the dimensions of the image, and the values corresponding to the colors of the pixels. **https://commons.wikimedia.org/wiki/File:Albert_Einstein_Head.jpg** and has no known copyright restrictions. You can also use an alternative .jpg grayscale image of your choice. The answers and the variable values will differ from what is shown here in that case.
2. Use the size function to check the dimensions of the obtained array ImJPG:
3. Check the type of the array ImJPG by using the command isinteger:
4. Convert to Grayscale
5. Lightening & Darkening the Image
6. Adjust Contrast
7. Crop the Image
8. Apply Matrix Transformations(Flip horizontally,Rotate 90 degrees)
9. Display the Modified Image