The assignment's objective was to test our understanding of Python basics and enable us to explore the powerful libraries OpenCV and Numpy. OpenCV and Numpy are the most implementation-intensive libraries which are used in computer vision and machine learning. They have functional implementations for power algorithms (SIFT, Hough Transform, etc) which can be directly used using OpenCV.

The first task in the assignment was to calculate the pth order norm of an array. This task seems trivial at first but as we proceed, new challenges arise because of the way the problem was posed and the implementation was expected. I had not much prior experience working with command-line arguments, but I learned how to utilize that using the argparse module. Another challenge I faced was variable arguments, either keyword or non-keyword. I explored the internet to improve my understanding of these concepts in python and was able to accomplish the task.

The next task was to come up with a row manipulation matrix such that after transformation all the odd indexed elements were stacked on top of the even indexed elements. This transformation can be performed through matrix multiplication of the input 1D array with the row manipulation matrix. The first solution I came up with to generate an N-dimensional row manipulation matrix, used a single loop and gave the solution in O(n) time. It was mentioned in the task description that ideally, the solution should be implemented without loops. After looking at various row manipulation matrices of different orders. I found the general pattern that a row manipulation matrix follows. I looked into the NumPy documentation to see if I can find a function that can tweak an identity matrix of order<N to make it into the desired matrix after some additional changes. I found the functions (*numpy.insert() and numpy.roll()*) that helped me in getting a solution without any loops. This task introduced me to some powerful functions in the NumPy library which I was not aware of despite using it for so long.

The next task was implementing the K-means clustering algorithm on synthetically generated 2D data. The task came with its code stub which was to be edited in the precise manner as instructed in the assignment. As I had experience with implementing k-means from previous courses, I believe my prior knowledge helped in completing it smoothly. Also, I had to ensure that my code worked well for higher-dimensional data.

After implementation of the NumPy and Python basics, I started with the OpenCV tasks which I had the most fun doing. The first task was to read an image, normalize it, and plot it using Matplotlib and OpenCV both. During the initial implementations, I was able to get the desired plots using Matplotlib, but the normalized image plot using OpenCV was not matching with the desired result. In my initial attempts, the normalized image plot came to be a black image. I later realized that I was making a mistake while converting the normalized image to an 8-bit unsigned integer (as *cv.imshow()* takes input in *uint8* only). After I fixed the problem the code worked.

The next task was to display a set of images and toggle left or right between them using two buttons in the window such that there is a wraparound. I was not familiar with how window controlling works in OpenCV. So I read the documentation on OpenCV's official website and

understood how the display window can be controlled with the keyboard. Loading the images from the directory and displaying them in a wrap-around manner was then easier to do.

The last task of the assignment was to record the webcam input and annotate the feed with our name on the top right corner. Also, we had to display the grayscale version of the same window adjacent to it. For this task, I referred to the link in the assignment for the OpenCV tutorial on getting started with videos. I learned how it reads frames and how we can annotate the frames using different functions in OpenCV. I also figured out how to get the dimensions of my name text. So that it can be automatically placed at the top right corner with the desired margin (as I already knew the frame dimensions).

The assignment packed huge learning values and was also fun to work on.