Report Lab 1

In this lab, I have created a script called rgb_yuv.py in which I have implemented a series of functions to test how ffmpeg works. This script contains an interactive part where the user is asked what task they want to perform (by choosing options). For each run of the script, only one operation can be performed, meaning if we want to test more than one function, we have to run the code again.

Task 1. Convert values

The goal of this task is to create a translator that converts RGB values to YUV values and vice versa. The script, named rgb_yuv.py, is capable of performing these operations. This script includes an interactive section that allows the user to choose the values that they want to convert.

Task 2. Resize image

This task is performed on the function called resize_with_quality and is able to resize images into lower quality using ffmpeg. To run the command we need to import the subprocess. Our input image is named 'eddie.jpg', which can be found in the same folder, and the output will be named 'eddie_resized.jpg'.

Task 3. Serpentine Algorithm

For this task, I have created a new method that can read the bytes of a JPEG file in a serpentine way. This algorithm reads an image and performs a serpentine (zigzag) processing on the bytes, alternating between reading the bytes in direct order and in reverse order. The function returns the processed bytes. To test the function, we will use the same JPEG image from the previous exercise.

Task 4

The method called convert_bw is capable of converting and compressing an image into b/w using ffmpeg. Also in this case, we use the same input image that in the previous exercises.

Task 5

The method run_length_encode performs run-length encoding on a series of given bytes. This function traverses the input and counts consecutive occurrences of equal elements, storing each element and its count as a list [element, count] in encoded_data. Then, return a list of these sublists representing elements and their contiguous repetitions in the input.

Task 6

For the last task, I have a class capable of converting and decoding an input using the Discrete Cosine Transform (DCT). Within this class, there are two methods: one for computing the DCT and another for computing the inverse DCT. To test the class, I have created an 8x8 matrix of random numbers, calculated the DCT and its inverse, and printed the results on the screen.