Over the course of the project, we took the same serial code essentially of turning a color image to grayscale. With each timeline, we used different methods of parallelization to try to speed up and make the process more efficient. The first one that we used was by using an MPI oriented code. With the MPI, it would take the serial code and then it would broadcast and scatter and gather so that it can scatter the work between the different processes. This allows the image to be converted quicker using a few processes but adding too many processes makes it less efficient.

The next method of parallelizing the code that we used is by using a shared memory way of multithreading. This method allows us to share the memory of the threads when using them separately to split up the work of converting the image. When run, the time it took kept decreasing as more threads were added up to 16 threads then it began to become slower again. Even though it sped up until 16, it was far more efficient when it was 2-4 threads used.

Finally, the last method that we used to parallelize this code was by using an openMP version of it. This required us to find the parallel region and with that, we can apply the pragma lines so that it knows how to split the load over the threads. This was more efficient as more threads were introduced with more efficiency around the middle of the tested values.

All of these methods can be used to parallelize the code based on the amount of threads that you use and the results may even be different in terms of speed and efficiency with different programs as well.