For my project the changes I made were: I sped up the original code, which enabled it to run much faster than before using multithreading, I added light sources, and I added rectangles to the raytracer. To speed up the code, I took the code which was responsible for writing to the file, and I instead made the code write to a 2-D vector, with each entry being a tuple representing the red, green and blue color values of the pixel. I then moved the code to write out to the array into its own separate function, which would return void, and would fill out the 2-D vector which stored the color values for the pixel. To actually write out the color values to the .ppm image file, I created a separate function which would loop over the newly filled in 2-D vector, and write out the color values to the image file. Initially, I also added multithreading utilizing the omp.h library, however, I later found that simply compiling with the -Ofast flag, combined with my changes, was significantly faster. The speed up allowed my code to be about 100 times faster than the code given with the textbook. The other two changes that I made, adding rectangles, and adding light sources, I implemented using the second volume of the Ray Tracing In One Weekend Textbook. To add rectangles, I created a rectangle class for each axis. The class took in the 4 corners of each rectangle, and the position it had along the 3rd axis. To create the light sources, I followed the Ray Tracing: The Next Week textbook section, where we create the light sources.