1. For this first problem, I started off with deciding what the root process would do. I decided that it should read in the filename, calculate the number of integers in the file using the file size, and then also read in the array as well which would be a global variable so that it can be used later by this root process. Then I split up the work of counting the occurrence of each element throughout the number of processes specified. Then using the reduce the function, I would calculate the sums of all occurrences and then sort the array accordingly. In the end, I wrote the entire array to the same file that was read in first.
2. The second problem involved me to first make the worker function which would do the partial sum calculation based on the thread ID and then use a mutex when adding to the global sum. In the main function, I first check if the arguments are given correctly, then I read in the number of threads and terms. Using this, I make an array of pthread_t and another array holding thread IDs. Then I create each thread passing in the worker function and its ID as the argument. After this, I join the threads to make sure the process does not finish prematurely. At the end, I print out the approximation, the actual value of PI, and also the error.