I only got through 3 versions, so I will only be talking about my design decisions for versions 1-3

For version one, I decided to make a server struct, a distributor struct , and a data file struct. This ended up being a good decision in my opinion, because instead of having to pass a bunch of parameters to each distributor functions, I could just pass one distributor struct. It also made it easy to assign indexes to the distributors, as I could have each dist instance store their assigned start and end indexes. Each of them had a todolist property as well, which was a vector that stored all the indexes of files they were supposed to pass to the data processing function. That was helpful.

For version 2, My struct design did not transfer over so well, because the indexes had to be written to a text file now. So what I did instead was have each distributor process write to a file with their index appended onto the end (e.g “dist + n”). The distributor would then write to that file the files to be assigned to other indexes based on what line it was on. For example, if the file of index 7 was supposed to be given to dist 5, then 7 would be placed on the 5th line in the file. This way when I processed the data, I could loop through the files and find the correct indexes assigned. I used the system call exit to end the processes after they were done. I used wait(NULL) so that the parent would wait for its children processes would be done before continuing.For version 3, My design pretty much stayed the same as version 2, except I had exec calls instead of exit calls.

My program did not have a lot of error handling, so it could still run even though incorrect arguments had been passed. It also does not compensate for different end of line characters.