While working on this project, I had to do additional research on exactly how to implement a child process using pipe() and fork() keywords in C++. I chose to program in C++ because it is the language I am the most familiar with out of C, Java, and C++. For the design of my code, I used the iostream library in order to open the input file and write to the output file. In order to create a pipe, I used the pipe() command and made sure to error handle in case the pipe could not be created. The fork() command is also used to create a child process to write to the output file, while the parent process read from the input file. A reminder that the other end of the pipe has to be closed before one end of the pipe can be read/written in. I also used the read/write commands to read and write to each end of the pipe, and I had a character array in order to store the text from the input file in order for the pipe to write to the virtual file. As far as error handling goes, I implemented a synchronous exception handler that is built in C++ (try throw catch) in order to test if the input file that the user wants to read from exists. If it does not, the program throws an exception and outputs the error.

Some difficulties I had while working on this program was understanding the exact inner workings of the pipe command. I had to do some additional research to find out the rules of using the pipe() command such as closing each end and the conditions in order for the child process/parent process to execute. I also had to make sure to initialize the character array that stored the text from the input file to the exact size of the text in the file, or else the read() and write() command would not work.

The link to the project video is as follows: https://youtu.be/t-w1zpKnsHk