To fulfill the requirements in the assignment:

1. Judge how many files to be processed
2. Fork processes for every file
3. Set a handling signal to act on interrupts like ctrl+c which end the process immediately.
4. For each file, read and process the document, remove the comments and empty lines, and store it into a new file with the original filename.cleaned.
5. The parent process waits for all the child processes are ended then the parents exit.
6. For the whole process, information of every step is stored in a log file and print the information on the screen as well.
7. For errors occurred, print specific notification.

Problems and solutions during programming:

1. How to remove comments.

The program checks the content letter by letter. When find a '/' '*' memorize the address, decide if it is a line comment or a block comment, when find the other end symbol, set the memory between the two addresses empty.

2. How to remove empty lines.

When the comments have been removed, the code is still unsatisfied with the requirement. Empty lines seprate the code into blocks.

To solve the problem, when '\n' is recognized, I set a state symbol to mark that a new line is started. Then check if the next letter is space. if so, do not write it into new memory space. if not, write the content into the new memory space and clear the state symbol . Thus the empty lines are removed from the code.

3. How to write the cleaned code into new files.

First, for each file, create a new file name (i.e. filename.cleaned).

Second, for each child process, save the buffered content into the new file.

Third, free the name space memory.

4. How to write log information into another file?

To solve the problem. Because the log should include different steps, so there should be a function to write the specific information to the log file together with its time stamp.

So another function is needed to show the local time in string.

5. How to decide that all the child processes are exited?

An active state should be defined for every process, and another function is needed to decide if all of the child processes are exited, the return value of this function should be a flag that indicts if all the active states are active.

6. How to make sure that if ctrl+C is pressed, the program quite immidiatly.

A signal is needed for this function. When an interrupt happened, check the signal and force the program to quit and print a notification.

7. How to lock the file

In the logger function, I used the file locking feature. I looked for related materials on the internet, and used the fcntl function to do the work, before writing to the logger file, check if the file is locked, and after writing, set the file lock.

8. How to calculate the processing time .

As the minimum value of `asctime` function is second, it is not sufficient to measure the processing time of the program, so I used the function gettimeofday in the sys/time.h and calculate the time interval at the end of process.

Function test:

features work	features don't work
Writing to a file - Create cleaned files for each input file. - Create a log file. Reading from a file - Read and process input files.	Code replacement with exec ().
Signal handling - Controlled shutdown when CTRL-C is pressed.	Use of memory mapped files.
Creation of child process(es).	Asynchronous or non-blocking I/O.
File locking.	
Remove comments - One starting with characters "//" and ending with new line (naturally) - One starting with characters "/*" and ending with corresponding "*/". Remove empty lines	
Parent process waits until the child processes have died before quitting.	

Future development:

1. Add the features not used in this assignment, for example, use memory mapped files which can manage the memory better.
2. The program did not consider the situation with ‘ and “, but in reality we have to consider if the // is between two double quotations, it is not a line comment.

3. A debug function can be added in this program, so that we can see what has been removed from the original code.
4. Add function to process all files under the same directory at a single time.