ELEC-C7310 ASSIGNMENT 1

Lari Haapaniemi – 651624

Program code “cleaning”

Introduction:

Our goal for this assignment was to create a program that cleans the given code or text file from comments starting with // or with /*, and removes all the empty space between rows, even those where comments used to be. The program had to be modular, compile without warning using a Makefile, write a log, take into account possible errors and make a clean exit using CTRL-C.
The program works correctly, and handles the error scenarios that I could think to test. I will go into more detail about the programs structure, process of making it and testing in this learning diary.

How the program works:

The program (after compiled with Makefile, instructions at howtouse.txt) asks for you to input file names to it for cleaning (either a singular one can be typed or multiple). The files have to be located at the root of the program (same place as Makefile), and it will clean the comments and rows, resulting in a file with its original name, but an .clean added at the end. It does this by first removing comments at the same time as moving it into a temporary file. The temporary file is then read into the final file (with the .clean ending), while removing the empty spaces both originally in it and as a result of deleted comments. After this process is finished it will remove the temporary file, and ask for another one to be cleaned.

Starting the program:

The assignment started with trying to find the correct task for myself, since my previous experience with C-programming was 2-years ago, so I needed something that could be easy to build at the start, and after that I could create the required features on top of the working product. This was the correct way to start the process, and I am happy with the results, even though there is a couple of things I would like to add, if I had the time to do.

The first step was to create a single module program that used no libraries, to remind myself how this language worked, considering I had been using mostly JavaScript for my programming

needs for the past year or so. The first draft of the program was just a file opening program, that would write to another file. After this I added the comment deleting feature, which worked decently for a single file, but I quickly found out my first issue: if I had a multi line comment that had another comment inside it, the program would not behave correctly, only removing the space between the first comment and the start of another comment.

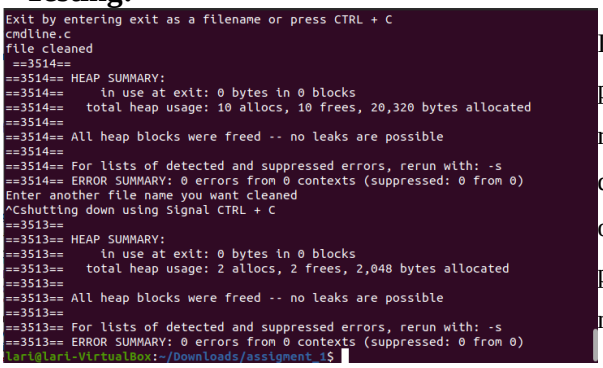
I quickly realized this was because finding the end comment was not its own loop, causing it to switch to another part of the function as soon as it saw something resembling the end of the comment. I decided to make finding the end comment its own while loop, that would only exit if it found the end of an comment (and later realized if there was no end comment, it should also check for the EOF). After this I added the line removing function, which was pretty simple, it just checked for 2 consecutive line changes from the temp file it had created after removing comments. This was not perfect either as I quickly noticed, since it could not delete the spaces that previously had comments. This was fixed towards the end, when I realized it was a lot easier taking a single line and trying to see if it contained anything other than space, instead of taking constant stream from the file.

Adding more features:

At this point I added the extra features, such as multiple modules, library and Makefile. These were new things to me at C-programming, except the library file which I had done for another course. This was probably the fastest part of the program, and I did not really encounter errors at this point. The program has 3 modules, clean_file.c which is in charge of cleaning the file given to it by clean_make.c, which is the main file. I decided to make the signal receiver a separate file called clean_sig.c, which is just in charge of shutting the program down if CTRL-C is pressed.

Then I added the flock system to the original source file, which took some googling, but it should work correctly, even though it was kind of hard to test with my knowledge. Log file was created shortly after this, where you can observe the steps the program took in the main file. After this I just commented the code, fixed the empty spaces that removing the comments were leaving, and tested different files to see what breaks, and added error reports when needed.

Testing:



I started off by testing valgrind, which I remember from a previous course. This did not show me any memory leaks when running different commands, which I was surprised by, but quickly realized I did not use commands such as malloc which often caused memory issues for me before (I know this would probably have been a better and more efficient way to use memory.

After this I tested files with problematic comments, such as the one below. On the left is original file, on the right the cleaned file.

```
strcat(temp_name,file_name);
//opening source file
/*/*
source = fopen(file_name, "r");
if (source == NULL) {
        perror("Opening file failed");
        return 0;
}*/
//locking the source file
flock(fileno(source), 1);
```

```
10          strcat(temp_name,file_name);
11          flock(fileno(source), 1);
```

Here is an example how the program is run:

```
lari@lari-VirtualBox:~/Downloads/assigment_1/651624as1$ make all
cc -Wall -pedantic   -c -o clean_make.o clean_make.c
cc -Wall -pedantic   -c -o clean_file.o clean_file.c
cc -Wall -pedantic   -c -o clean_sig.o clean_sig.c
gcc -o clean_files clean_make.o clean_file.o clean_sig.o -I.
lari@lari-VirtualBox:~/Downloads/assigment_1/651624as1$ ./clean_files
Enter a file name you want cleaned
Exit by entering exit as a filename or press CTRL + C
testing_clean.c
file cleaned
 Enter another file name you want cleaned
^Cshutting down using Signal CTRL + C
lari@lari-VirtualBox:~/Downloads/assigment_1/651624as1$ make clean
rm -rf *.o *~ *clean_files *.log
lari@lari-VirtualBox:~/Downloads/assigment_1/651624as1$
```

If the program is getting problematic files (e.g. they don't exist or are missing an end comment):

```
ter a file name you want cleaned
Exit by entering exit as a filename or press CTRL + C
problematic_file.c
Opening file failed: No such file or directory
cleaning file was not succesful
Enter another file name you want cleaned
```

```
 Enter another file name you want cleaned
testing_clean.c
Your file is missing an end comment, so the clean file might be corrupted
file cleaned
 Enter another file name you want cleaned
```

^ In this case the file will still be cleaned, but does not function properly.

What could be added:

One of the obvious ones would be to have a separate folder for the files to be cleaned and files cleaned. This would make the program a lot cleaner and more professional. This could be done with commands such as mkdir() and opendir(). Another one would be adding an option to leave spaces between functions for example, or just adding an option to only delete comments, or only delete spaces.

Conclusion:

This program and diary took me around 3 days to make, and I am happy with the results. Considering I always found C-programming difficult, making this program gave me a lot of confidence to work with this language in the future. Obviously the program is not perfect, but considering I did not find any problems in my testing, and managed to fix every problem I encountered, I find this assignment to be successful.