Andrew Ryan Burcroff

Operating Systems

December 9th, 2017

**Operating Systems Final Project**

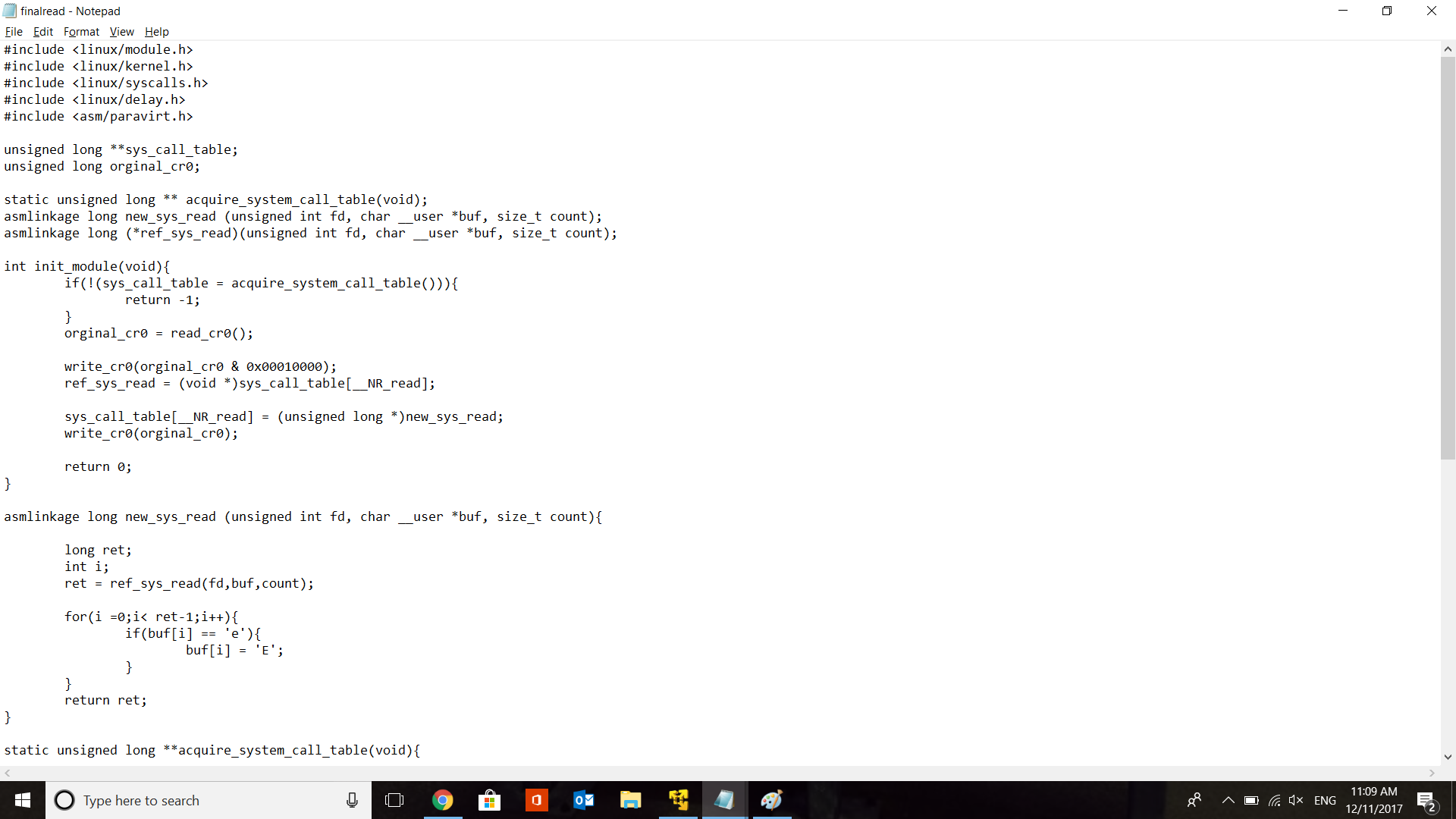
**Overall Design:**

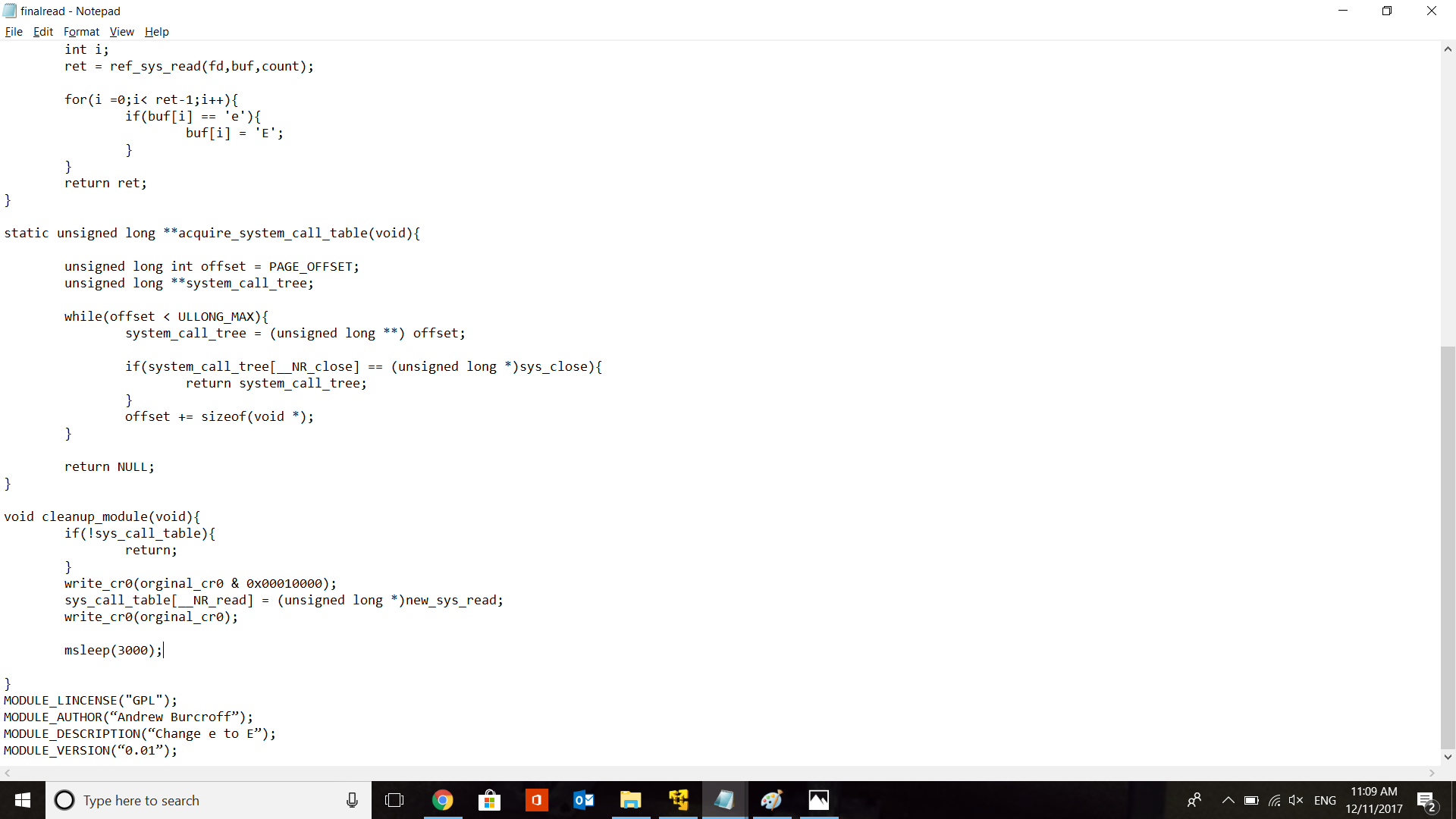
This is the CS 4540 Kernel Filter Assignment that is required for Operating Systems class. In this file, I look at the overall concept of how I finished this assignment. We will look at the code, make file and screenshots of it was completed. For this project, we are supposed to write our own read function or to modify this function to where it we read a file that contains any "e" it will change this character to a capital "E". I am using a text editor to read the filter called input and output. This code runs in kernel space, either via a loadable module.

After my research I came up with six major steps to complete this process. First, we need to find the system call table and get access to it. Find the read function in the system call table, save this function as a variable to restore back. Then get the ability to write to this function, then store a new variable to the new system call of read. Then set the new function of read to the old version of read in the system call table. Finally, place or write the new function to the table.

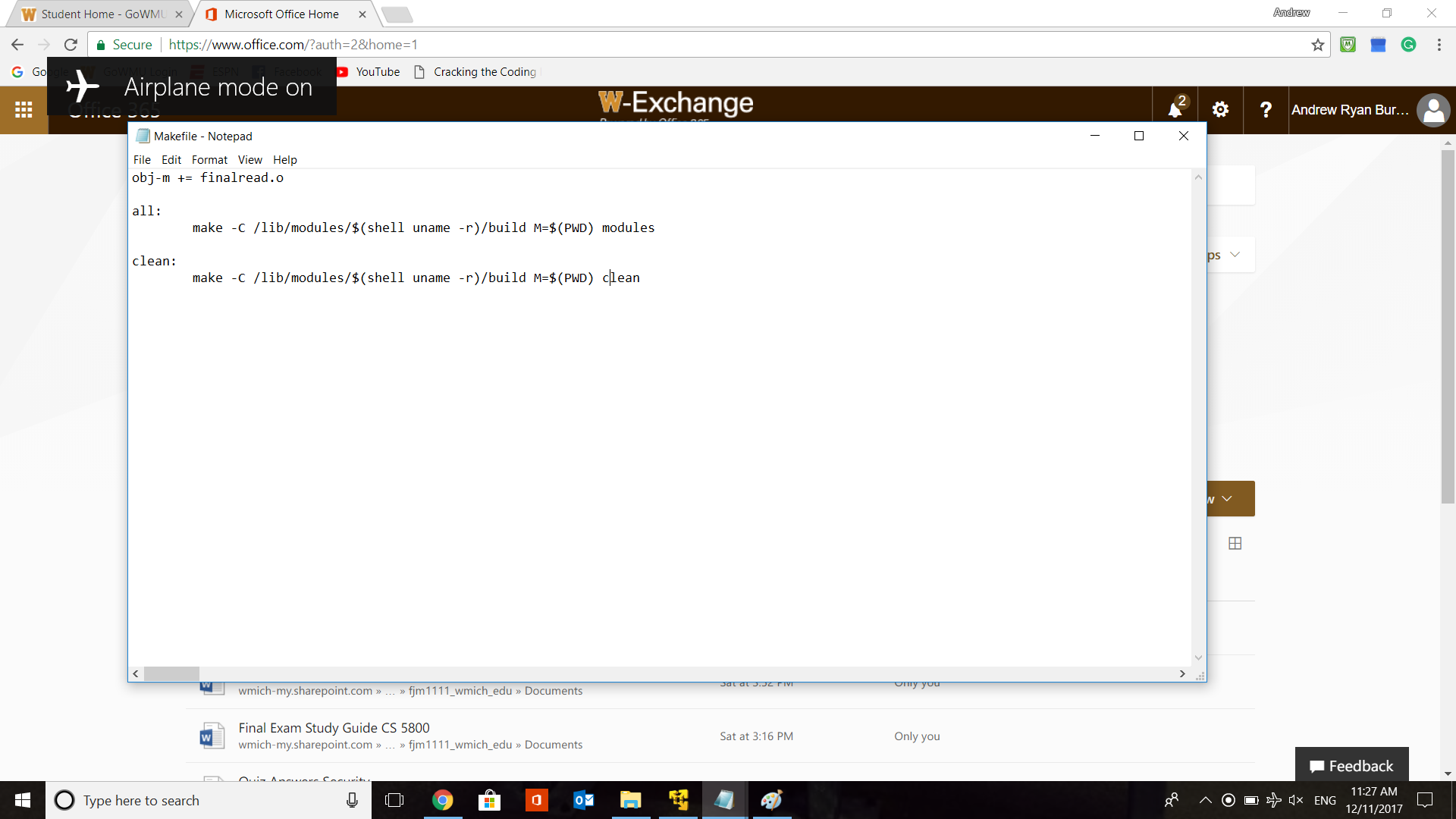
**Code:**

Here is a screenshot of my code.

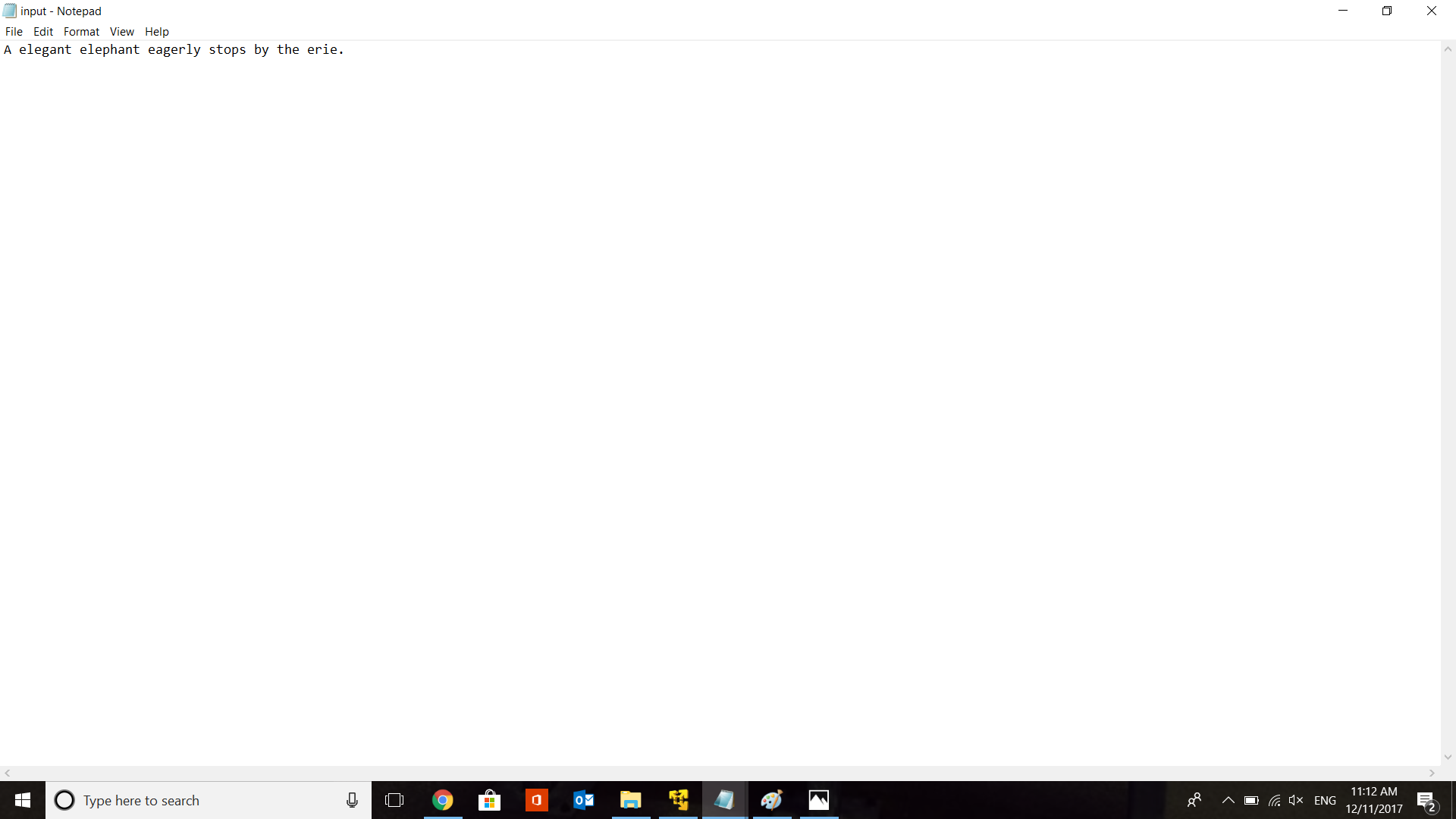




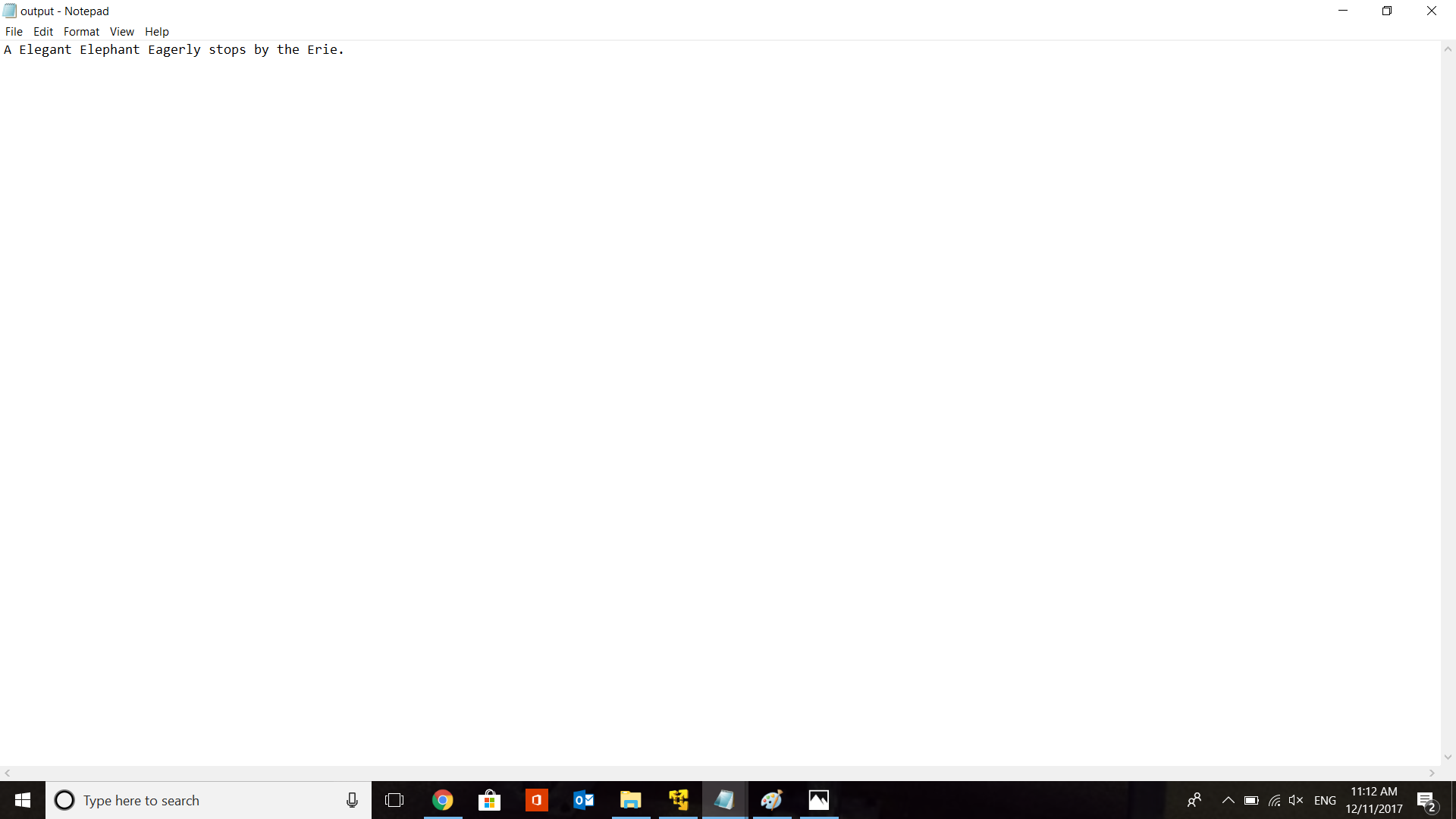
Screenshot for the Make file.

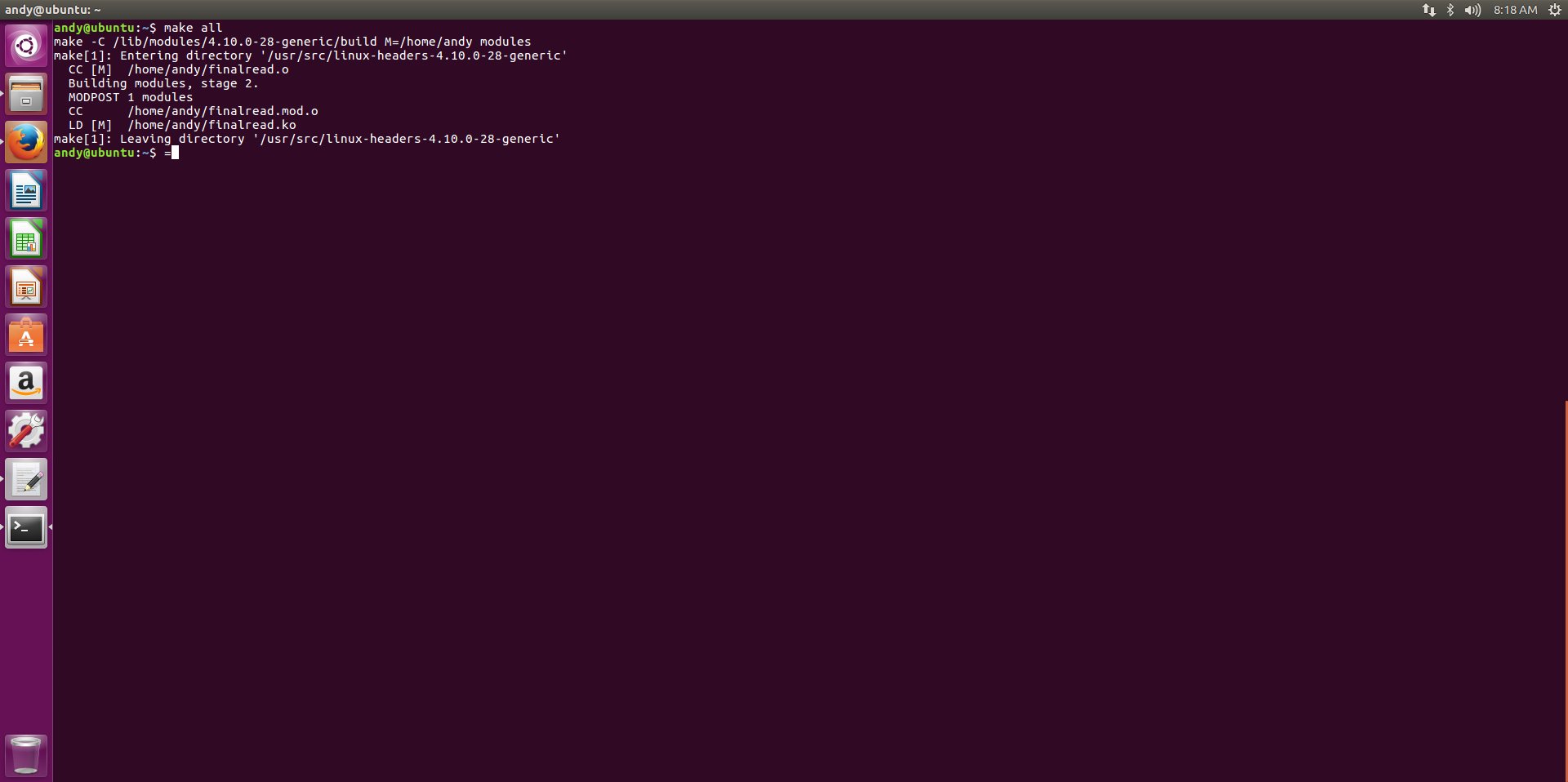


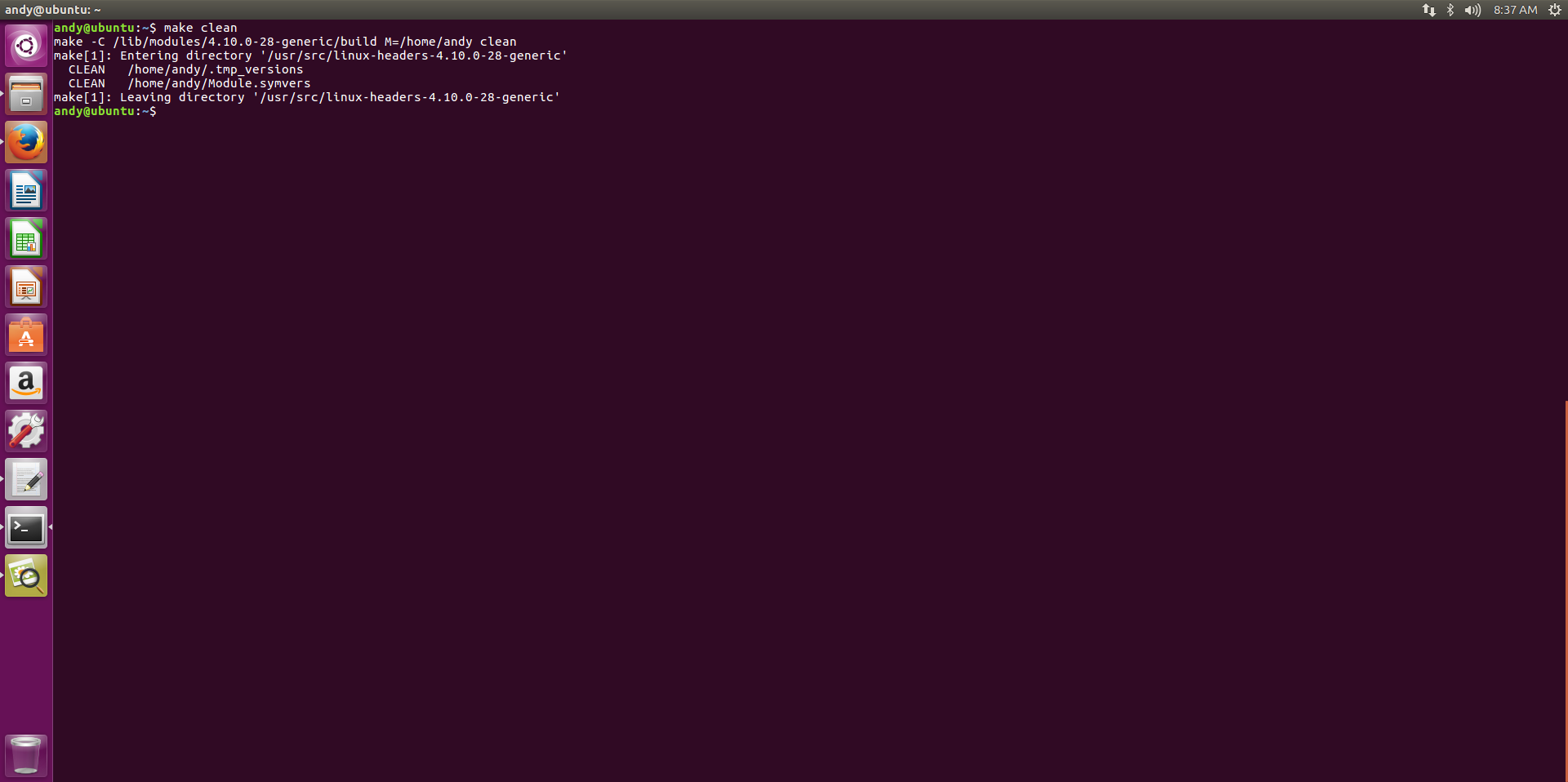
Screenshot of the input.txt. That I tested for my code.

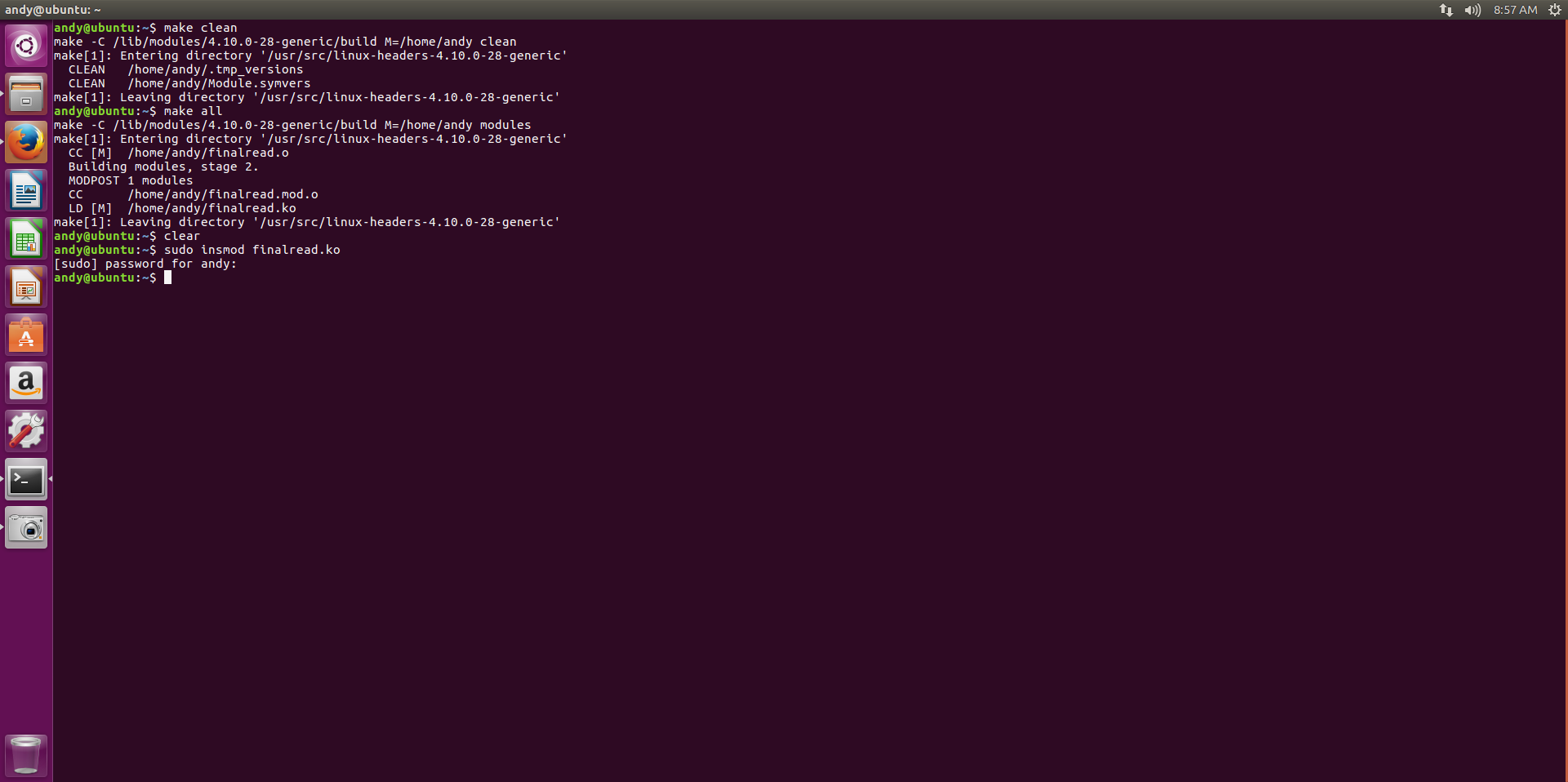


Screenshot of the output.txt. That I test on.

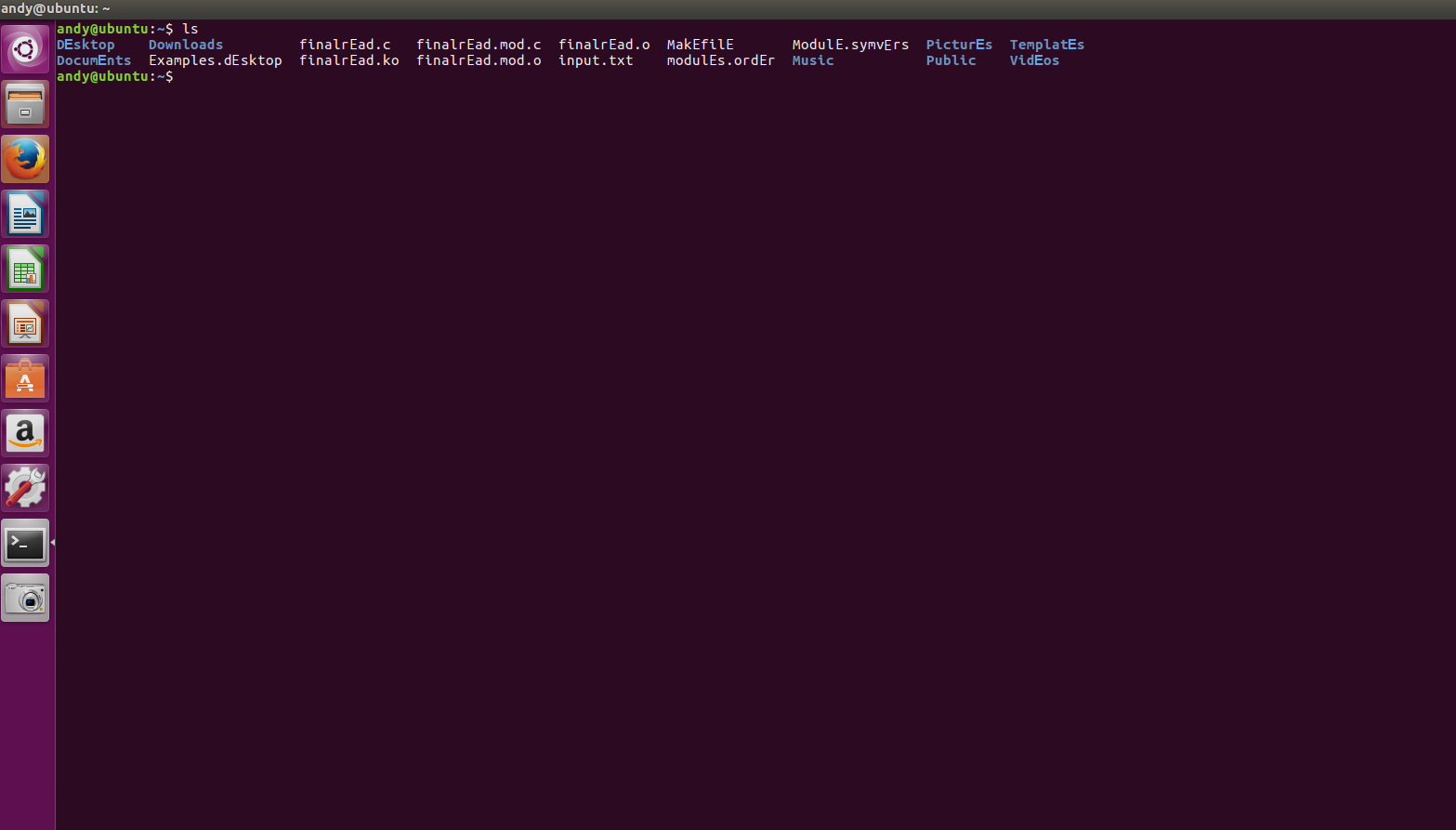


**Running:** These screenshots of the program running from start to clean. Make allMake clean

Showing that code complies.



**Testing:**  Shown in the screenshots of the testing.



**Debugging:** I have screen captures single stepping into my code in a virtual machine using a debugger.

I attempted to use GDB to step through of the code.

**References:**

1. <http://se7so.blogspot.com/2012/07/hijacking-linux-system-calls-rootkit.html>
2. <https://wiki.archlinux.org/index.php/kernel_modules>
3. <http://learnlinuxconcepts.blogspot.com/2014/03/what-is-loadable-kernel-module.html>
4. <http://tldp.org/LDP/lkmpg/2.6/html/>
5. <https://linux.die.net/lkmpg/x40.html>
6. <http://syprog.blogspot.com/2011/10/hijack-linux-system-calls-part-iii.html>
7. <http://derekmolloy.ie/writing-a-linux-kernel-module-part-1-introduction/>
8. <https://www.howtoforge.com/reading-files-from-the-linux-kernel-space-module-driver-fedora-14>
9. <http://derekmolloy.ie/writing-a-linux-kernel-module-part-2-a-character-device/>
10. <http://mixeduperic.com/ubuntu/how-to-set-the-root-password-in-ubuntu.html>
11. <http://edoceo.com/howto/kernel-modules>
12. <https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/6/html/deployment_guide/ch-working_with_kernel_modules>
13. <https://tnichols.org/2015/10/19/Hooking-the-Linux-System-Call-Table/>
14. <https://linux.die.net/man/2/read>
15. <http://haifux.org/lectures/86-sil/kernel-modules-drivers/kernel-modules-drivers.html>
16. <http://nairobi-embedded.org/access_dev_files_from_krn.html>
17. <http://www.makelinux.net/books/lkd2/ch11lev1sec4>
18. <https://www.linux.com/learn/kernel-newbie-corner-your-first-loadable-kernel-module>
19. <http://www.tldp.org/LDP/lkmpg/2.6/html/lkmpg.html>
20. <https://bbs.archlinux.org/viewtopic.php?id=139406>
21. <http://www.thegeekstuff.com/2013/07/write-linux-kernel-module/>
22. <https://memset.wordpress.com/2010/12/03/syscall-hijacking-kernel-2-6-systems/>
23. <https://linux.die.net/lkmpg/x121.html>