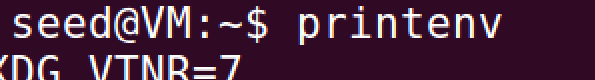
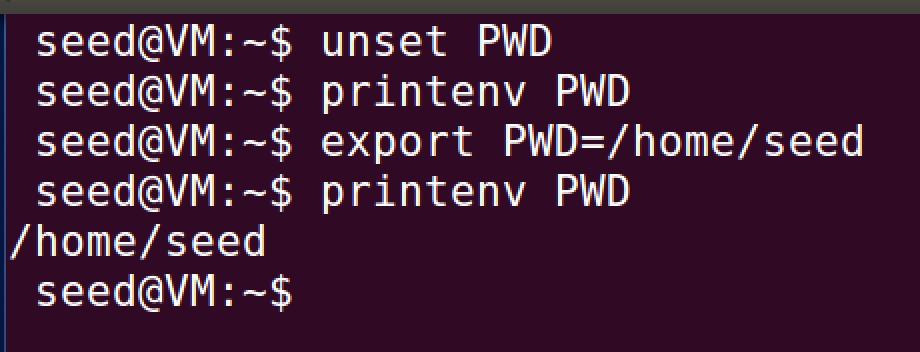
Lab 2

Task 1: Print Environment Variables and Export and Unset certain Environment Variables

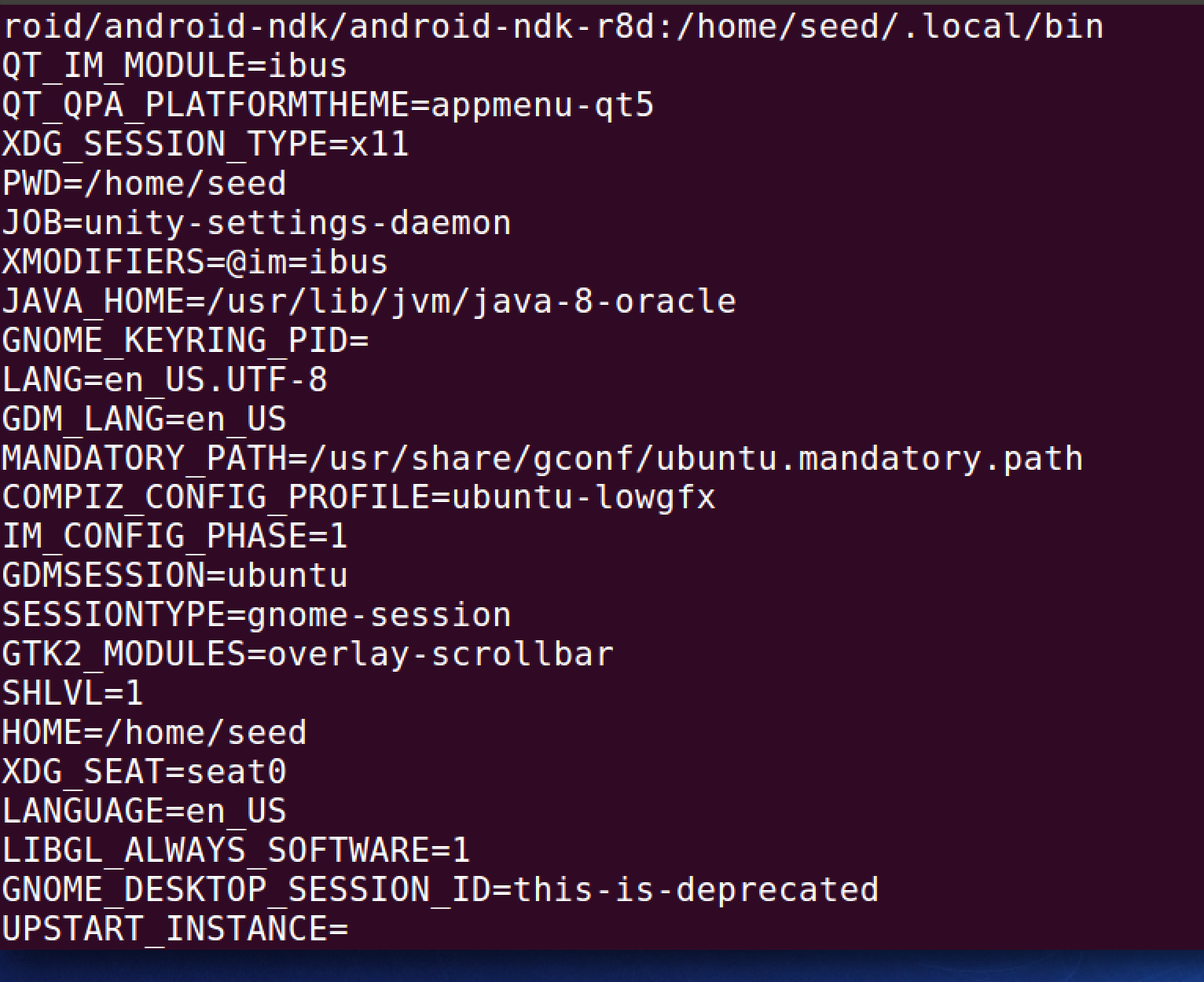
Commands Used:

1. 

Export and Unset Commands and output

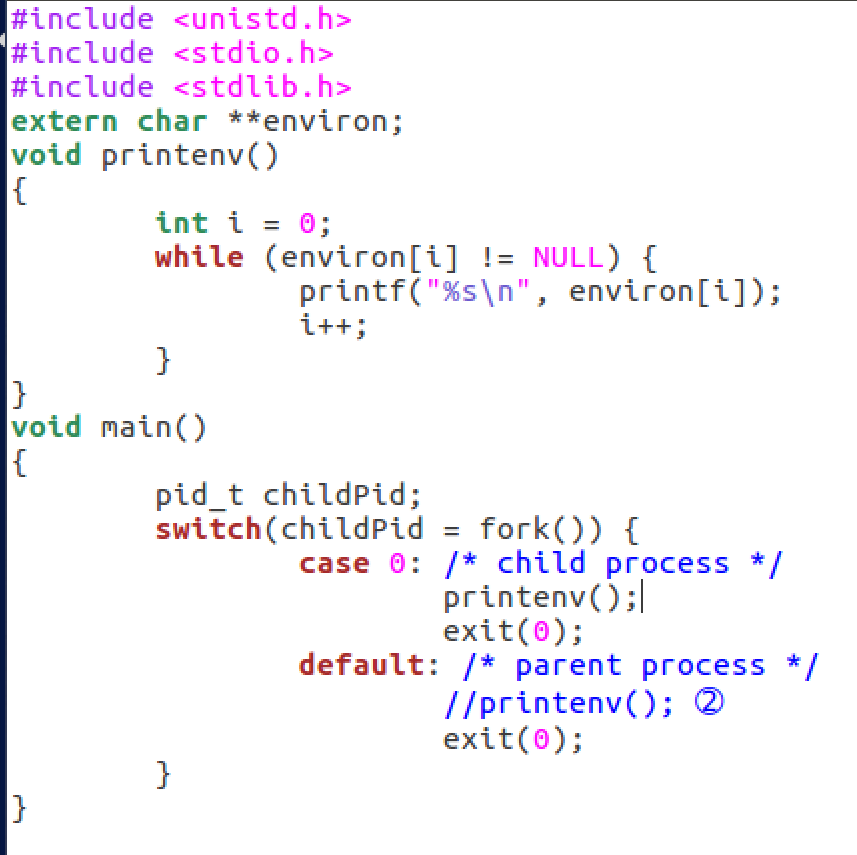
1. 

Running Output: The values are changed by using the export and unset commands

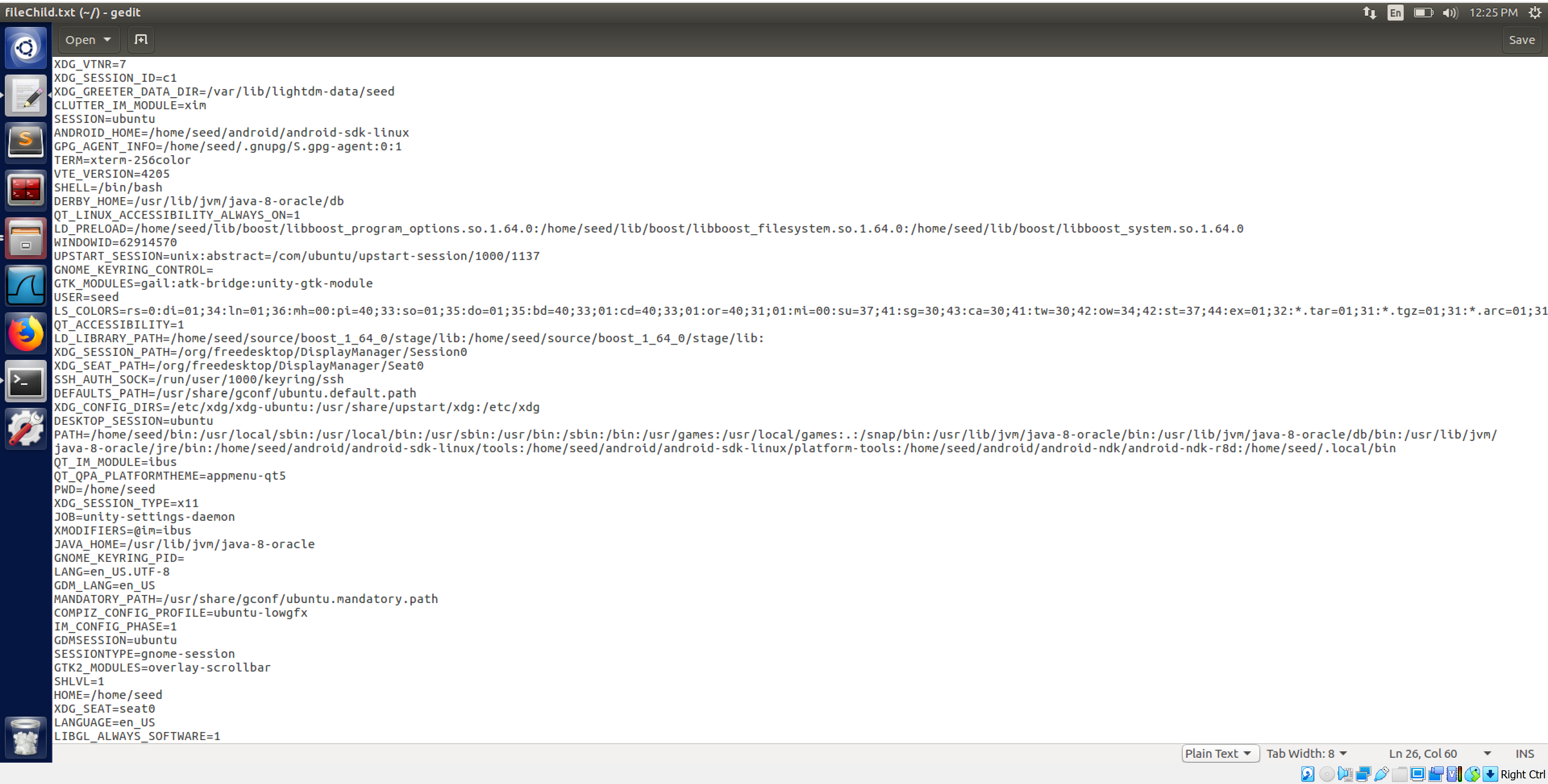
1. 

Task 2: Describe Observation of Following Code, Save Output into a File, and Find the Difference

Code 1:



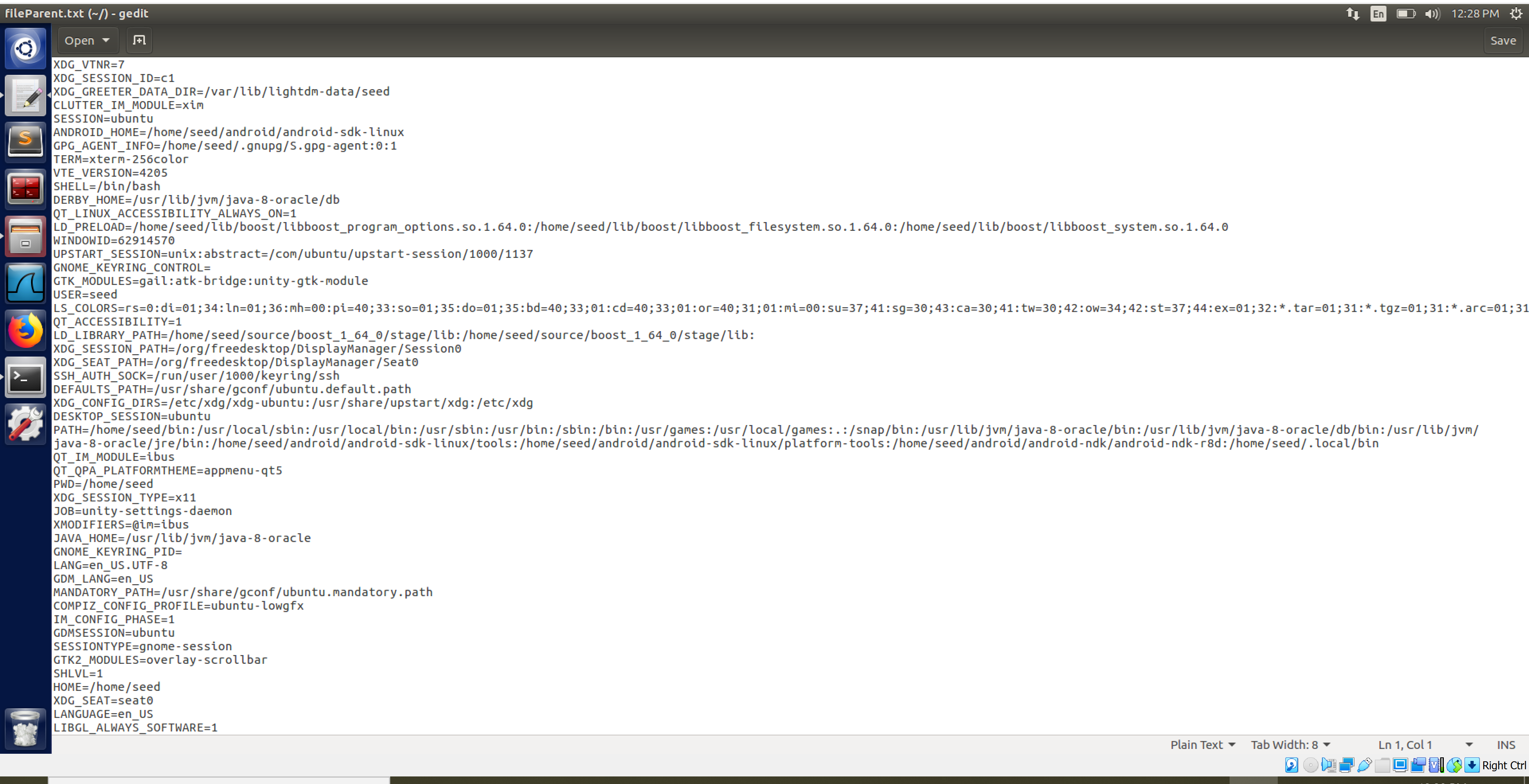
Output 1: It appears to print out all the environment variables



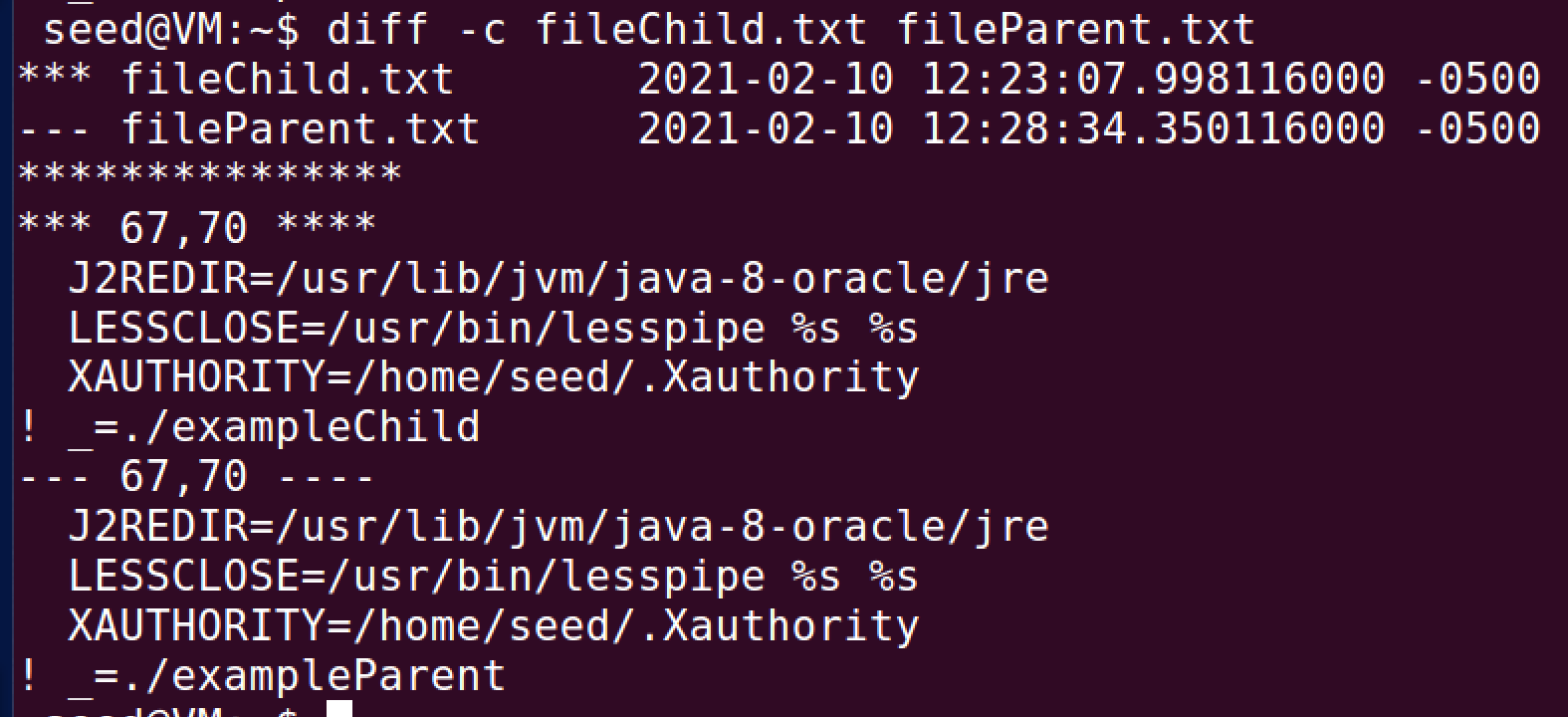
Code 2:



Output 2: It appears to print out all the environment variables and looks similar to the last one

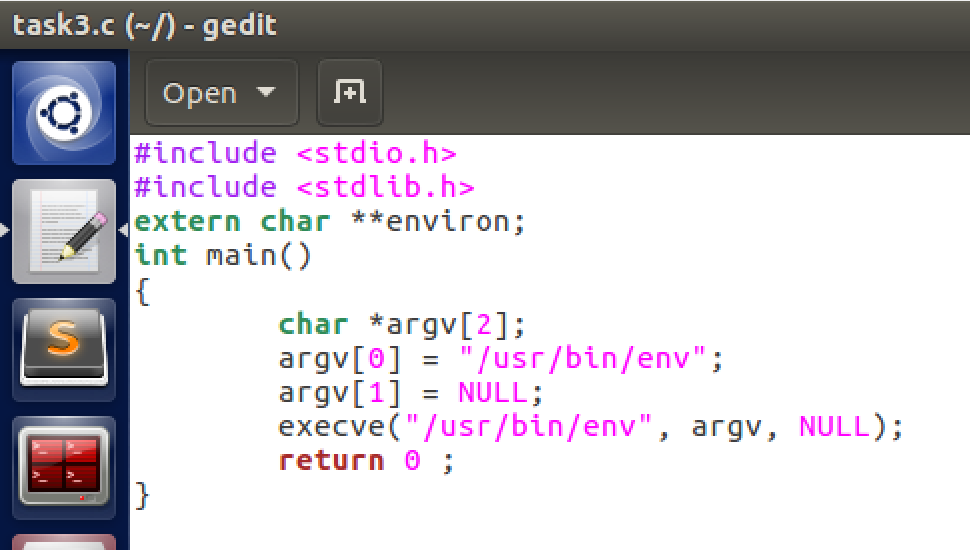


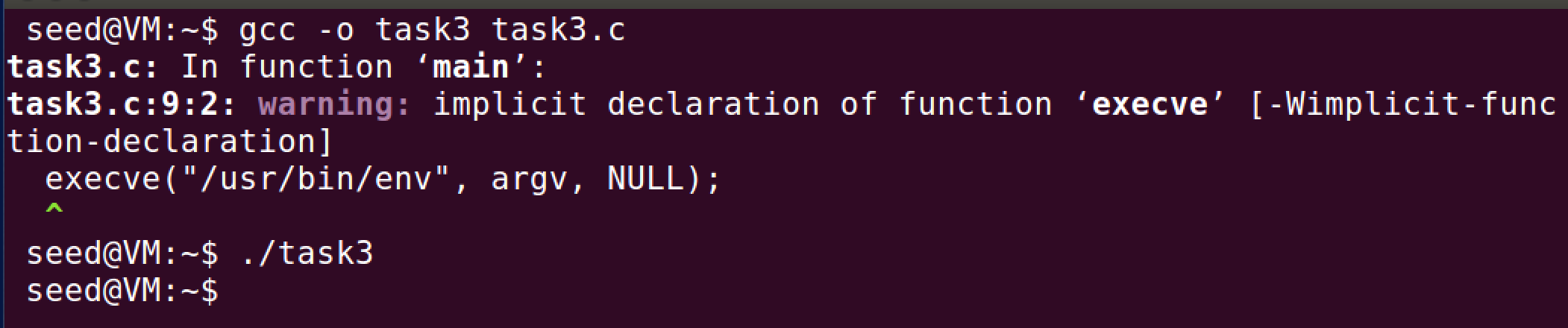
Difference: It seems the only difference is the XAUTHORITY variable



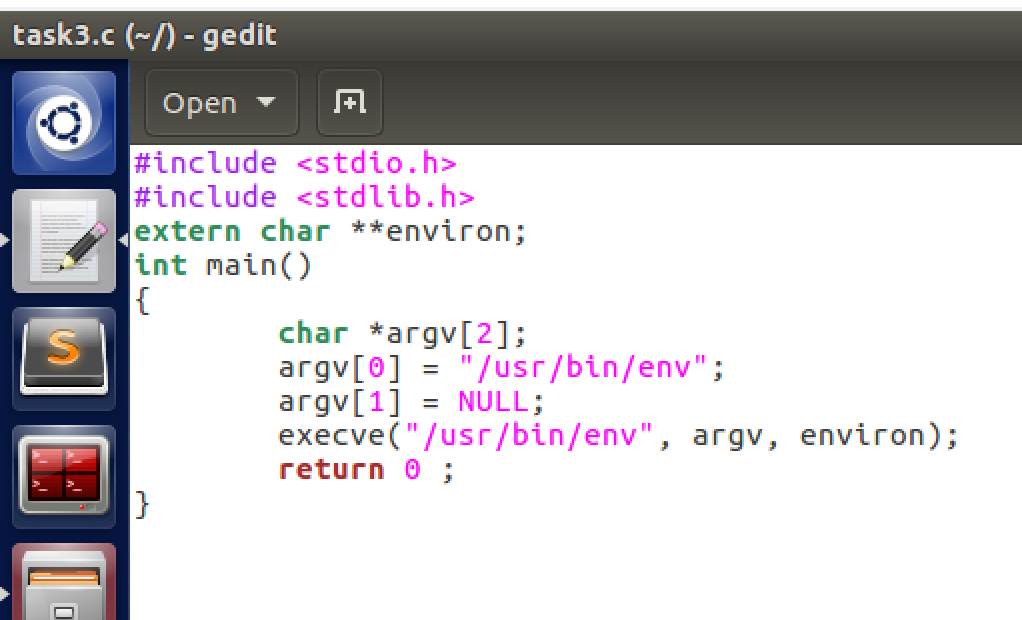
Task 3: Compile, Run, Observe, and Document Code

Code1:



Output1: Nothing is output with the code

Code2:



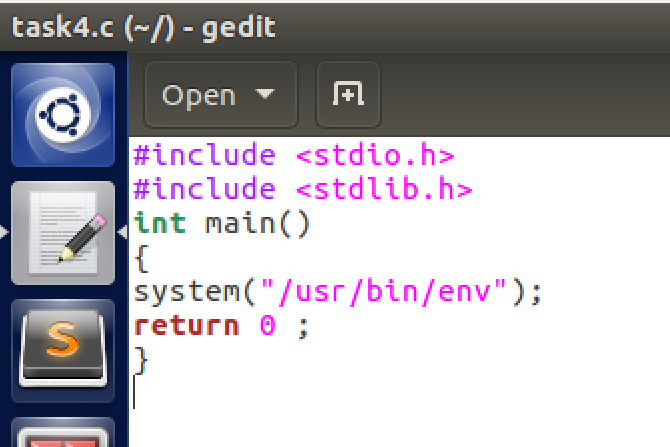
Output2:



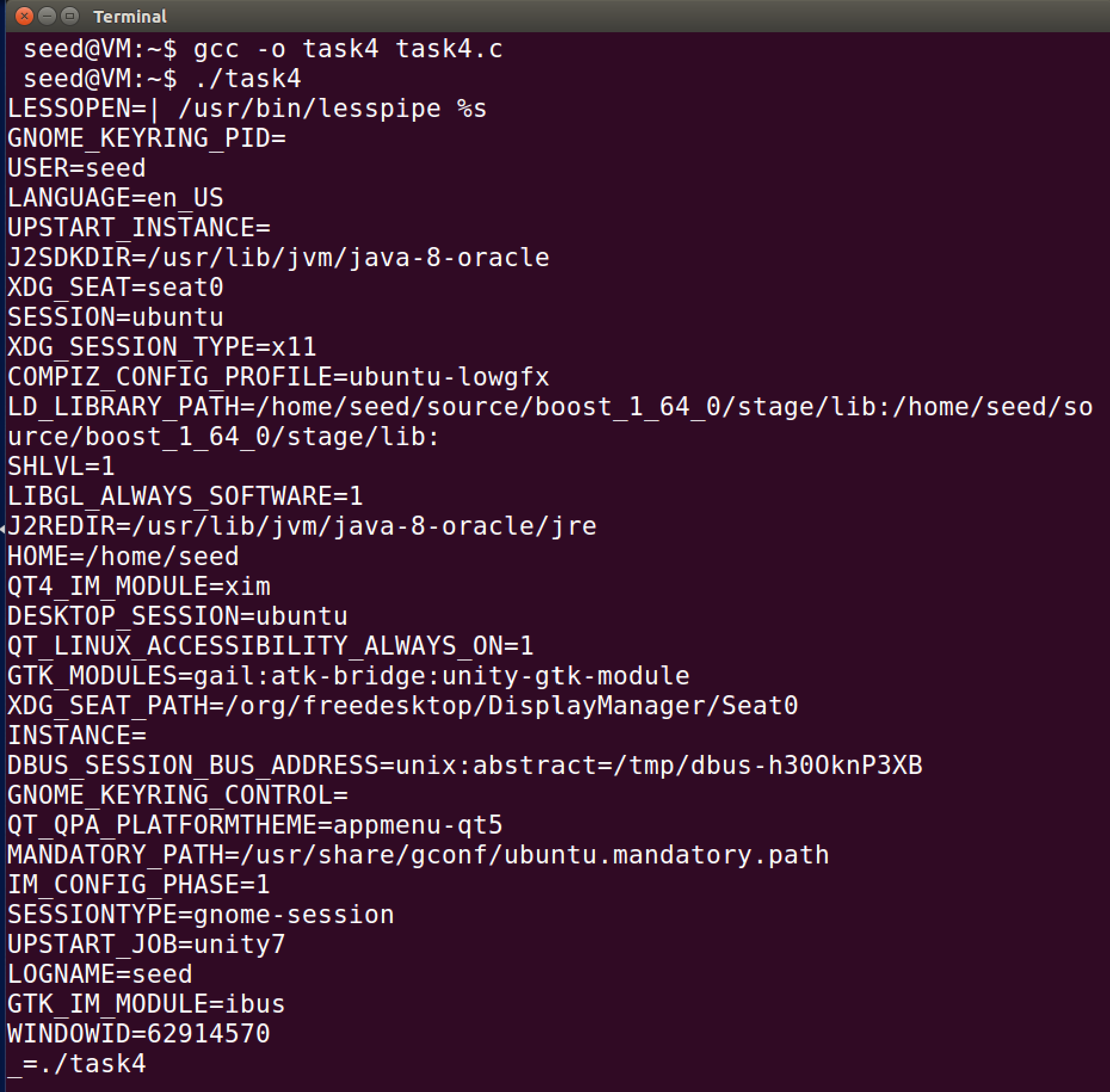
Conclusion: Based on the argument at the end of the execve() command it determines what is output. When it was NULL, nothing was printed. When it was environ it output the environment variables.

Task 4: Run the following code

Code: This code prints out the environment variables

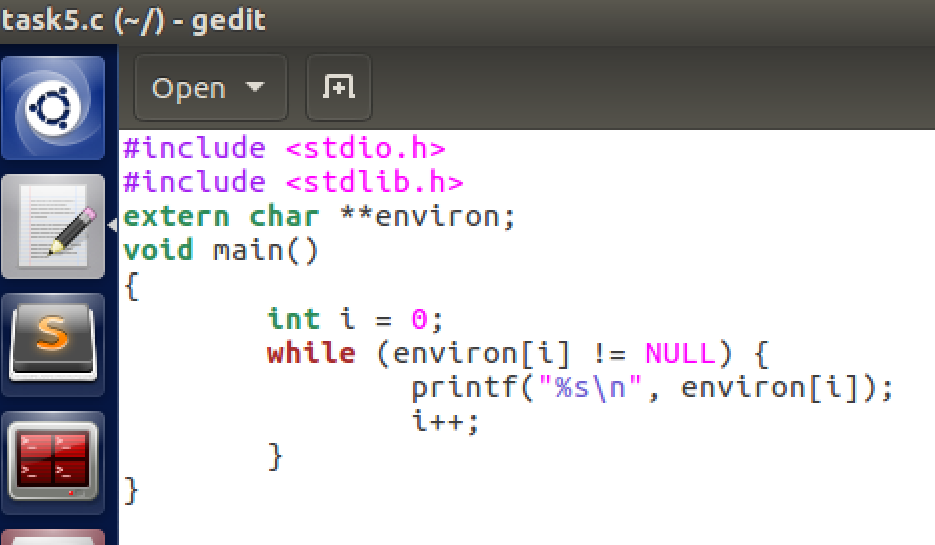


Output: Here is the output for the following code

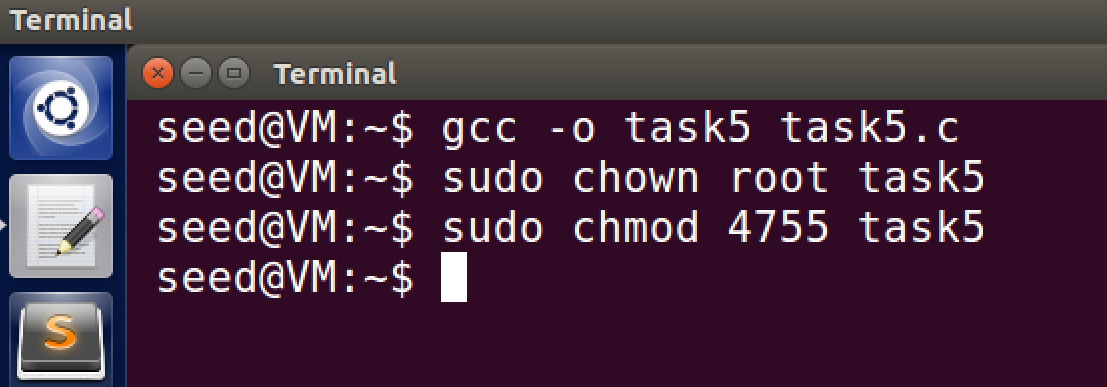


Task 5: Set-UID

Code: This code prints out the environment variables

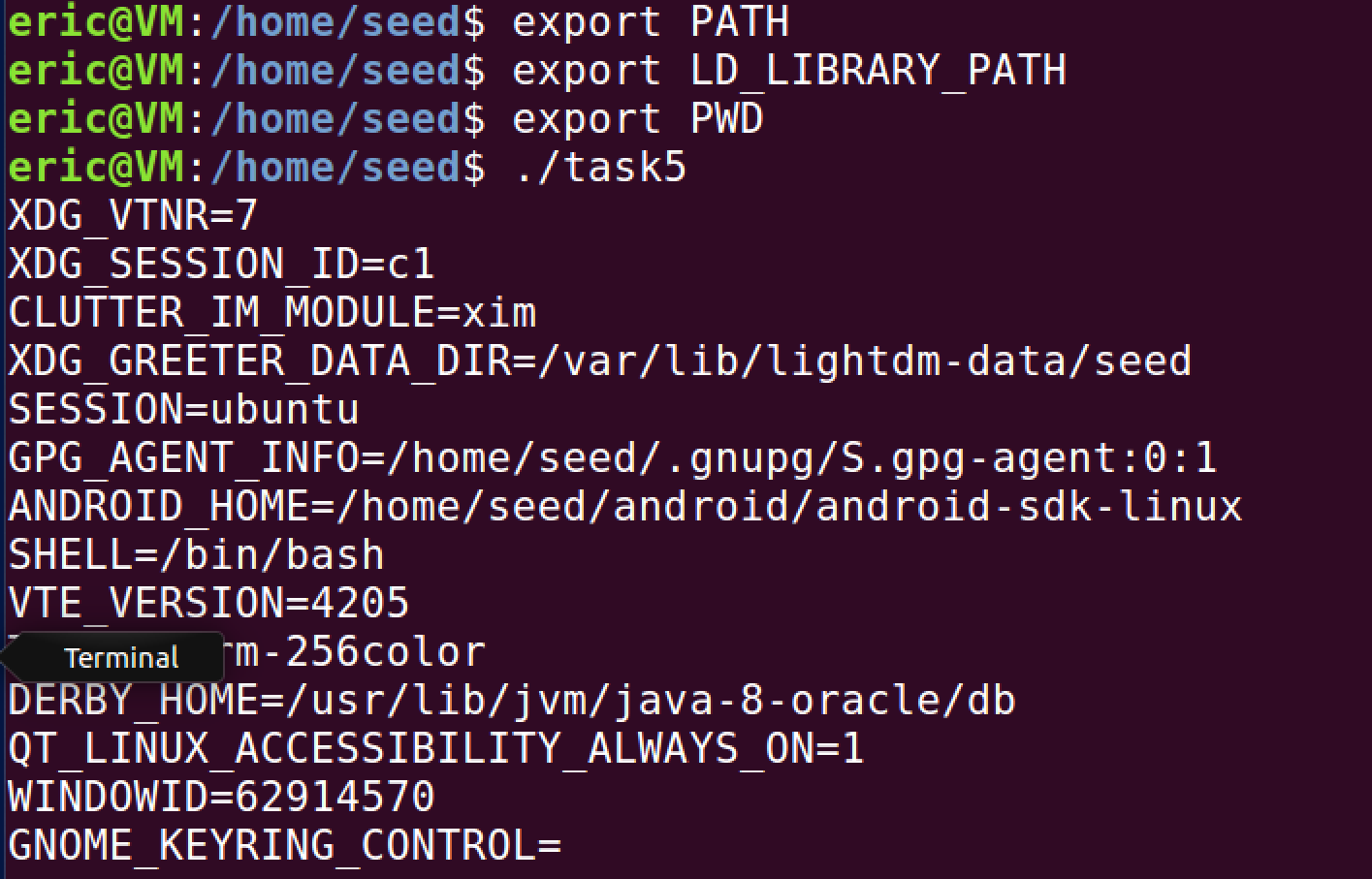


Compile and make it a Set-UID



I compiled the program under the name of task5 and made it a Set-UID program

Use Export on environment variables:



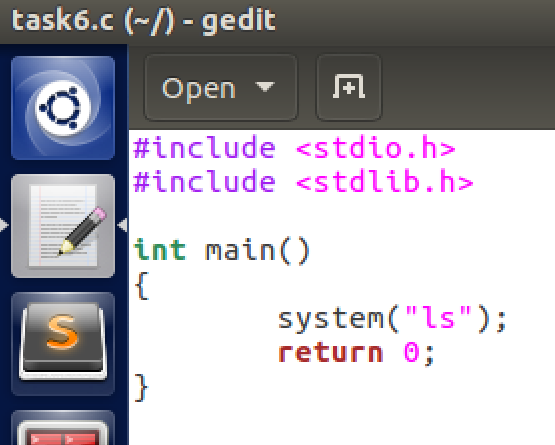
Here I exported the variables using the normal user.

Run the program and display output:

When I run the program, only the PWD and PATH variables are present. I do not see the LD\_LIBRARY\_PATH variable anywhere. I wonder if it is not possible to exclude those variables from the child process.

Task 6:

Code: Here the code is getting the list of all files and folders from the system

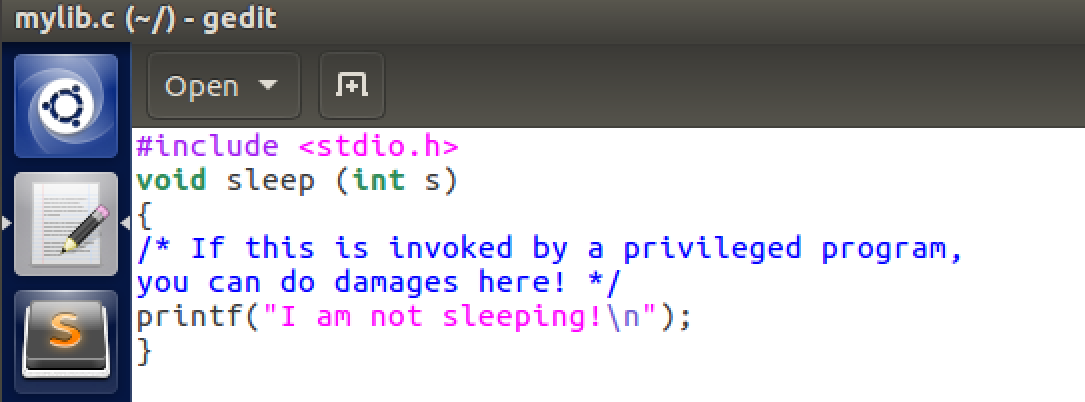


Output: I tested running the code and also ran the command and both gave the exact same output. Therefore, I believe it is running the root privilege.

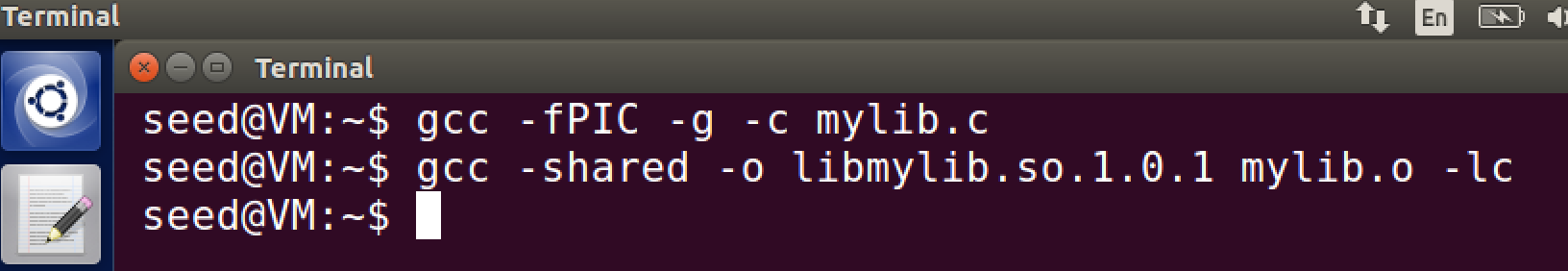


Task7:

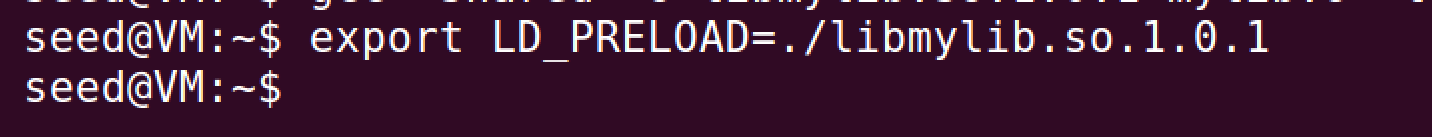
Code:



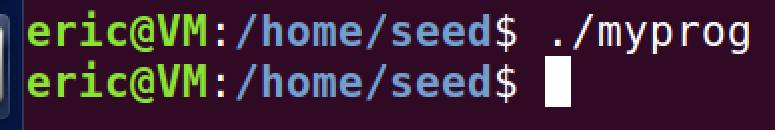
Compile the code with the following commands:



Next, we set the LD\_PRELOAD environment variable:

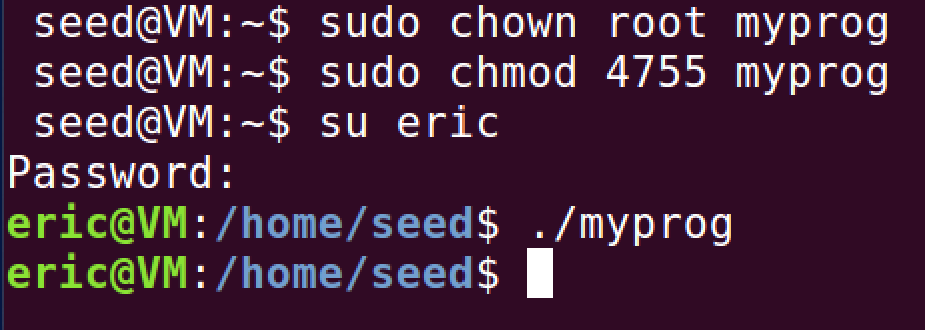


Step 1: Run myprog as a normal user



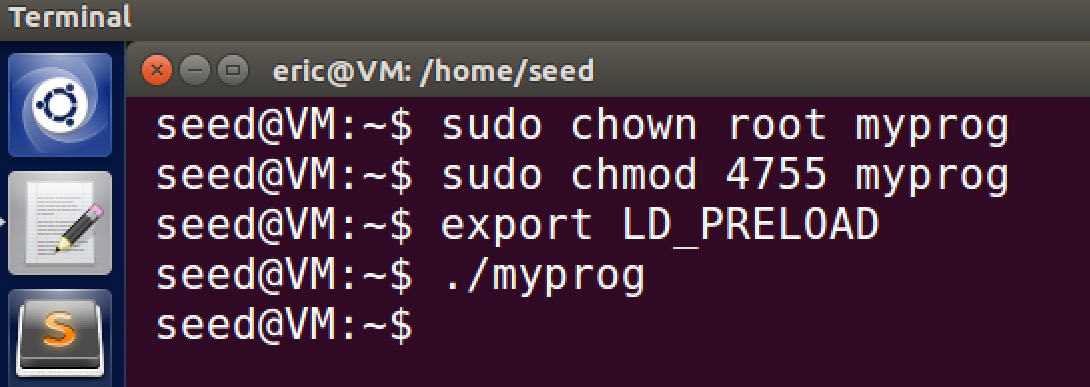
Nothing happened it seems, but it did pause for a about half a second before the next line was generated.

Step 2: Make myprog a Set-UID and run as a normal user



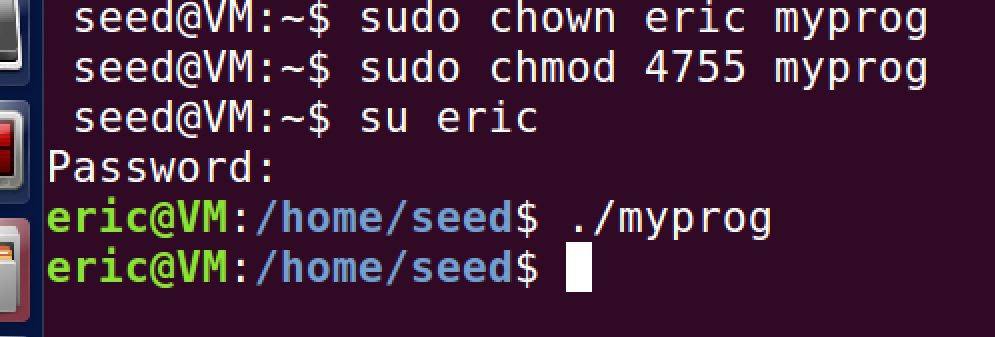
It paused again but it seemed like it did it for a bit longer than last time but it was hard to tell.

Step 3: Make myprog a Set-UID root program, export the LD\_PRELOAD environment variable again in the root account and run it.



It seems like all this program does is stall for like one second and then continue past that. Cause that is all I am seeing it do.

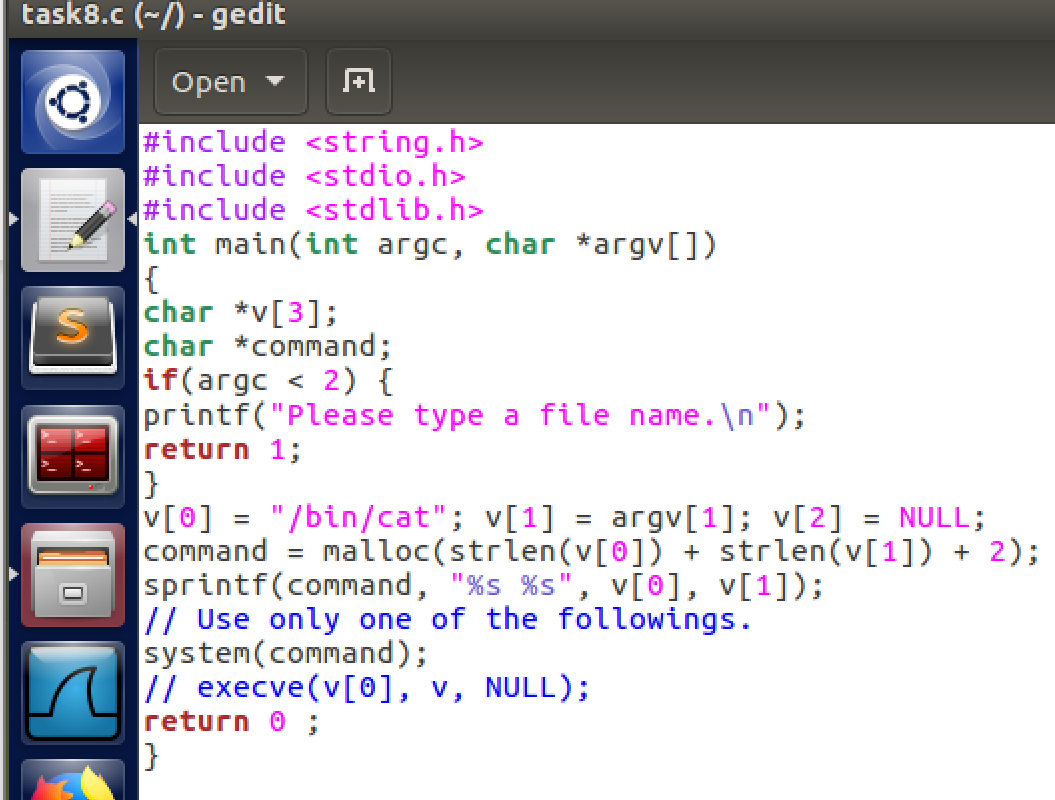
Step 4: Make myprog a Set-UID user1 program, export the LD PRELOAD environment variable again in a different user’s account (not-root user) and run it.



Observations: Some of them had maybe a little extra delay in the execution of the program but it is too hard to tell. They all did the same thing which was to delay the function.

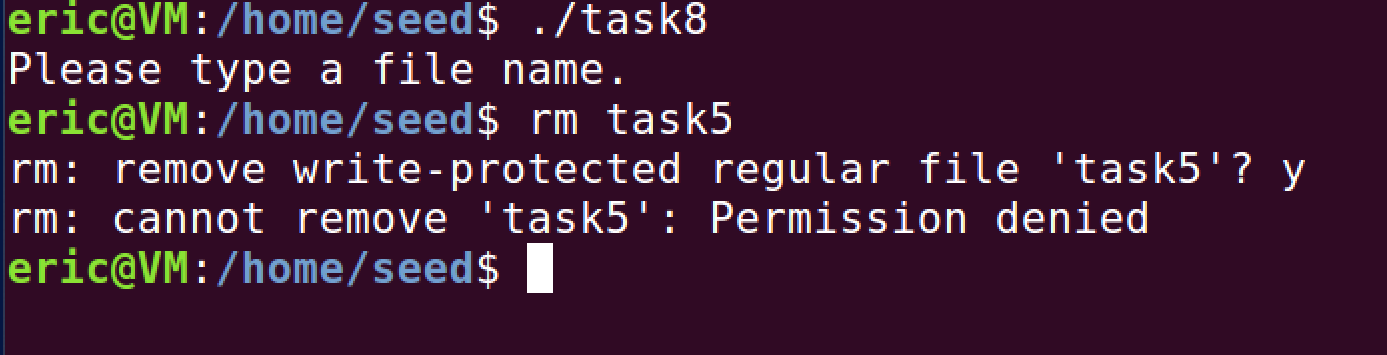
Task 8:

Code: If given the name of a file, it will be print the output of that file. If given a command it checks for the permissions of the user to see if the command can be executed

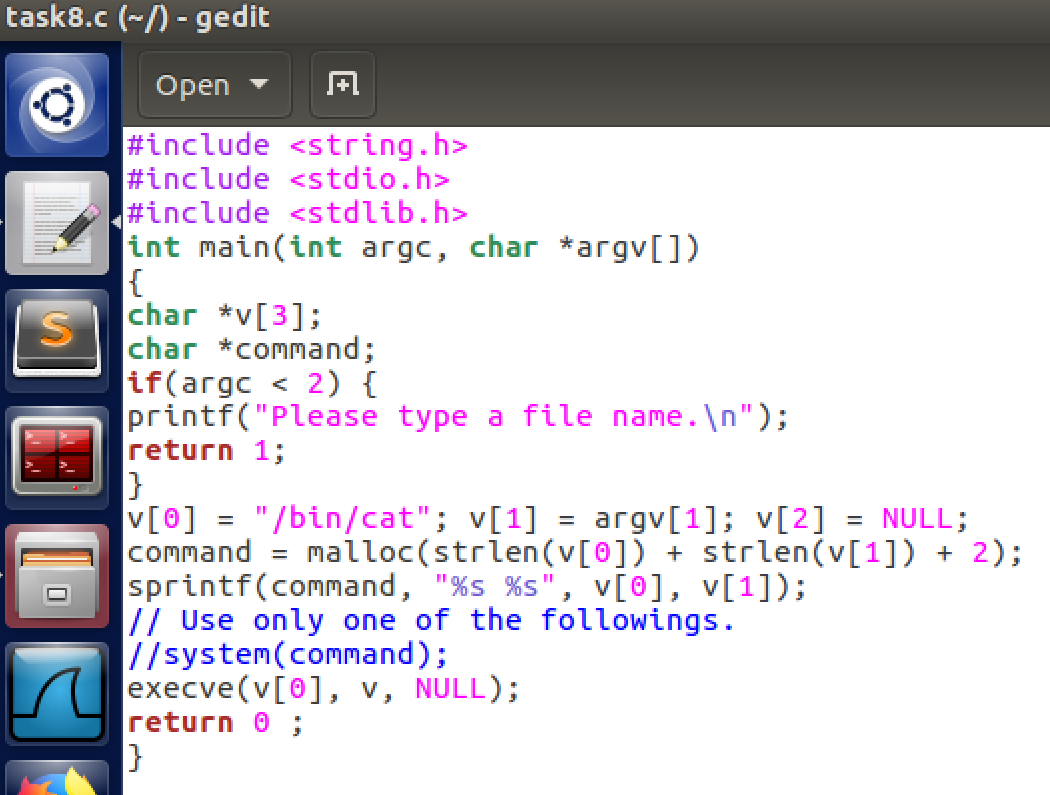


Results:

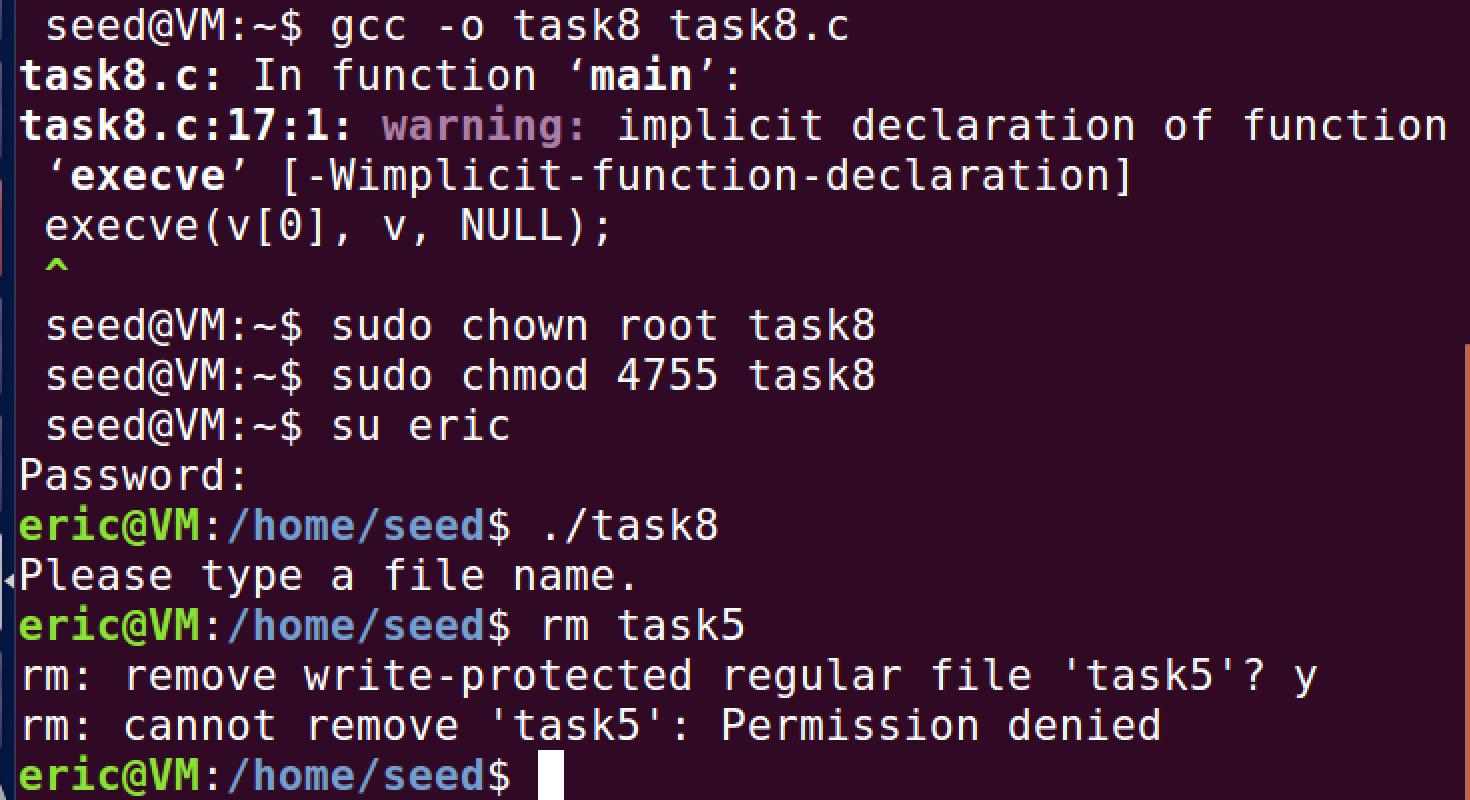
Trying to remove a file when I am not the root user does not work



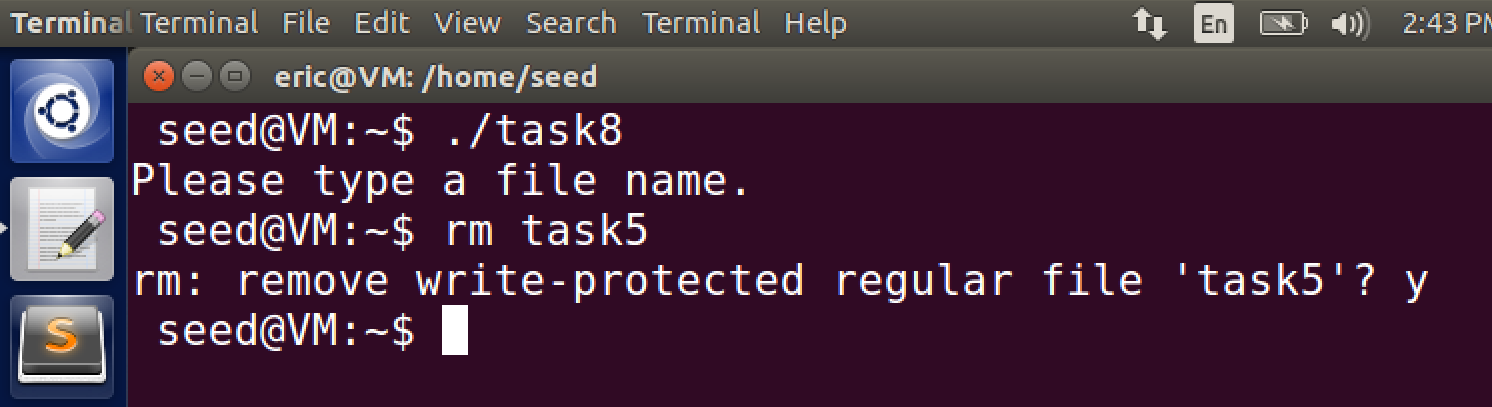
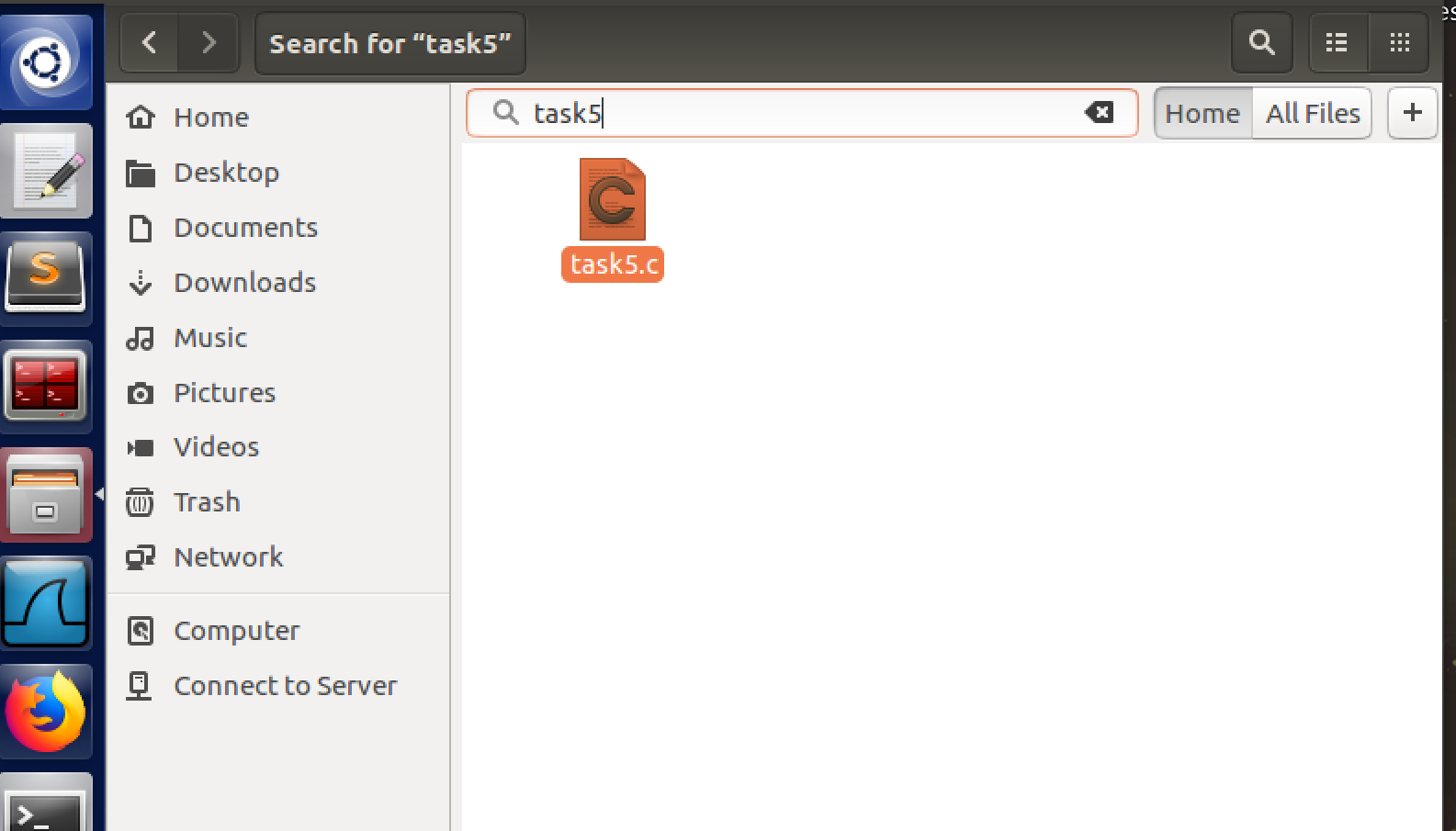
Code Revised: Instead of using the system call it uses execve which changes it from the root privilege to a normal user privilege.



Results: Trying to remove the file again with the revised code



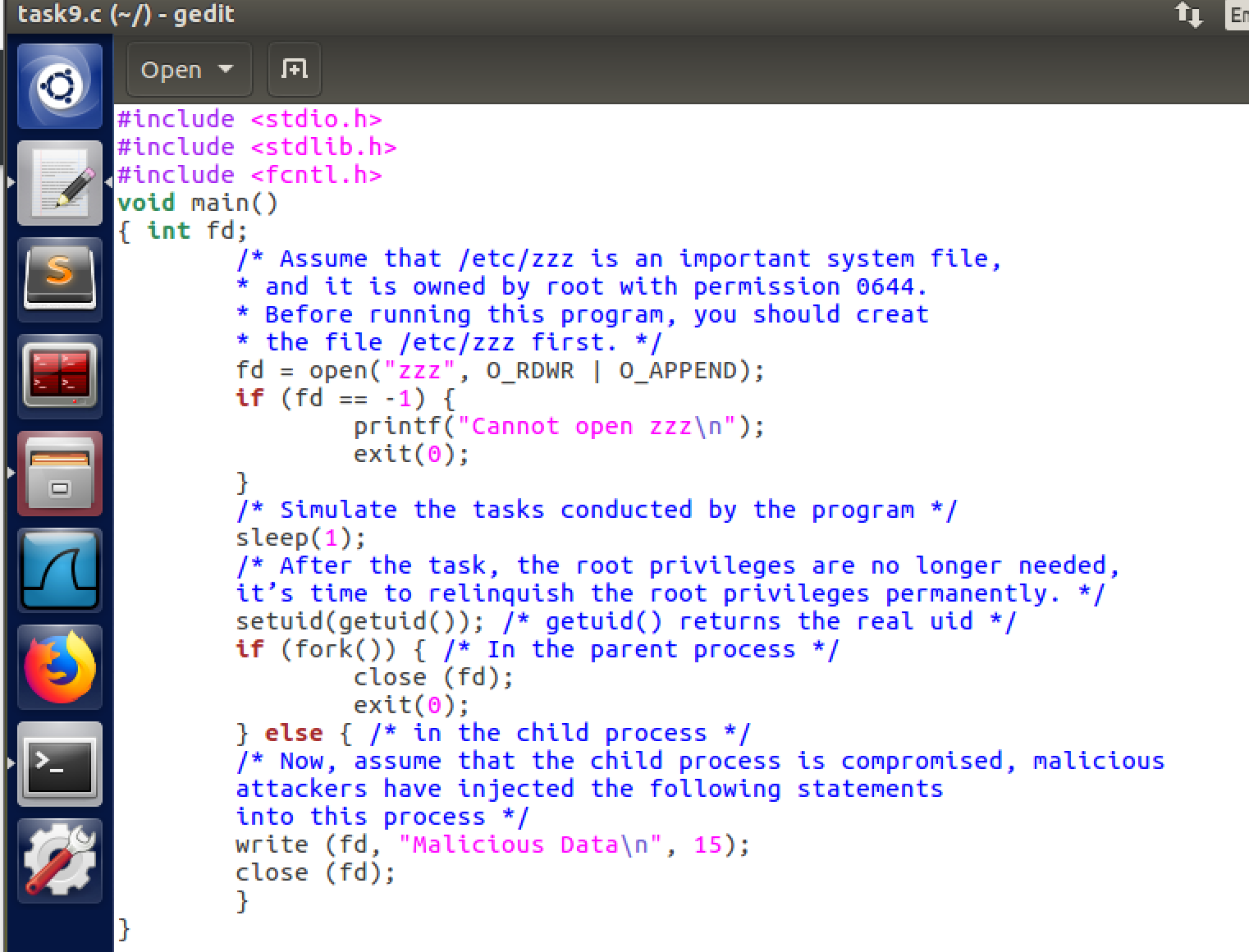
Conclusion: After I recompiled the code after revising the program. It still would not allow me to remove a file. I believe this is because I am not the root owner. So I, as a different user does not have the permissions to do this. When I do this with the rood user, it allows me to do so:

File task5 is not present anymore.

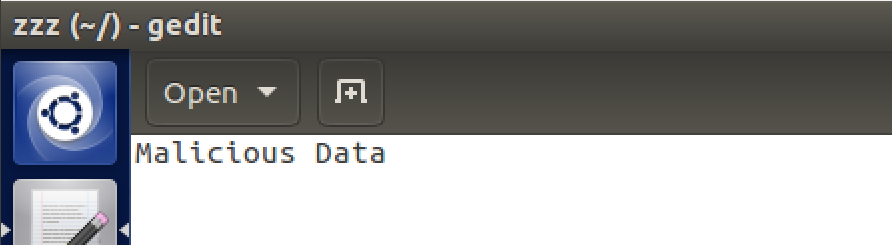
Task 9: Capability Leaking

Code: I had to change the file name because it said that file names were not allowed to have slashes ( / ) so that is why it looks different. After it called the sleep function the root privileges were no longer needed and were relinquished. This allowed for them to set the uid to the RUID and allowed for a normal user to have access to the file and modify it based on the program.



Results:





Observation: The file “zzz” was modified which would prove right as to what the text was saying in the fact that the privileges were downgraded and allowed for someone besides the root owner to access and modify the file after it was compiled by the root.