

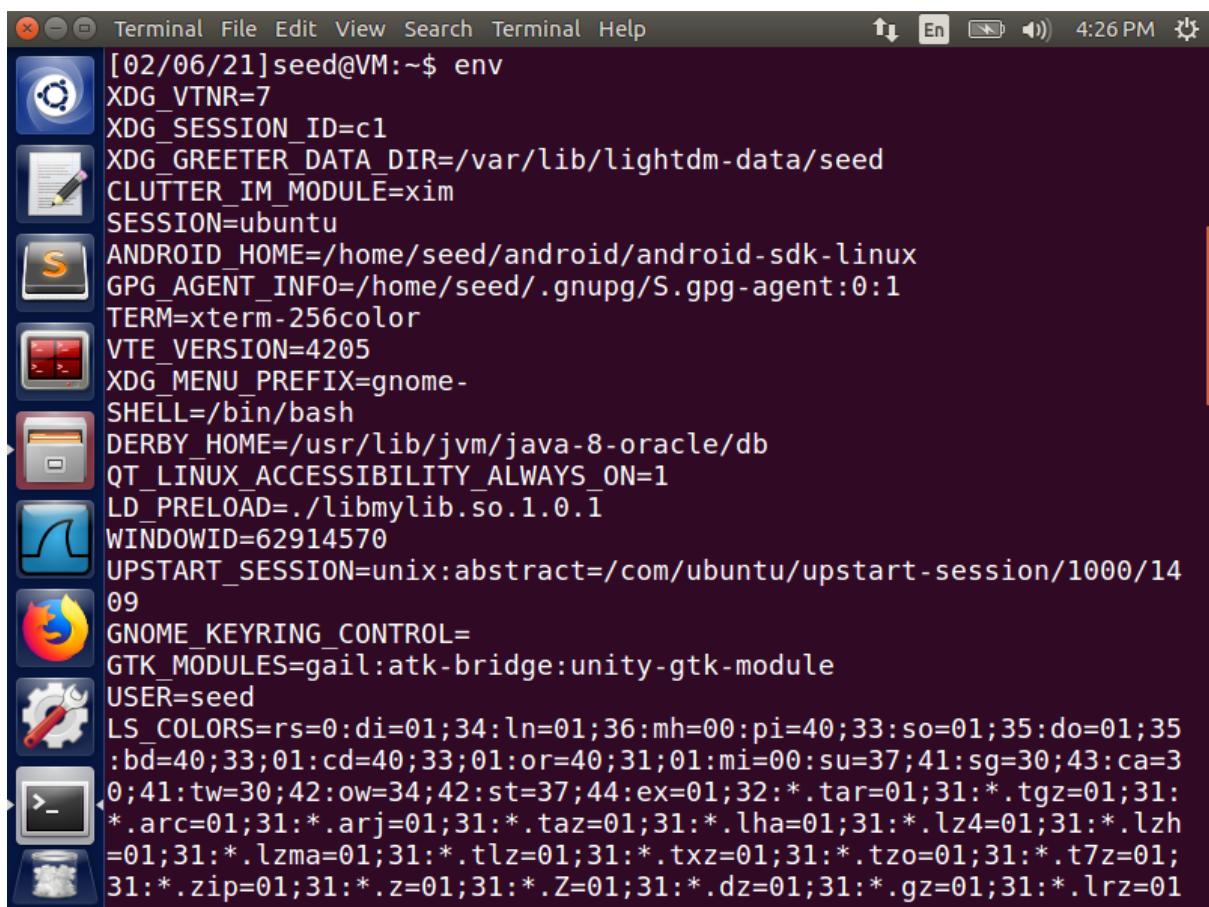
ASSIGNMENT -1

Name : Sudharsan Srinivasan

ID : 1001755919

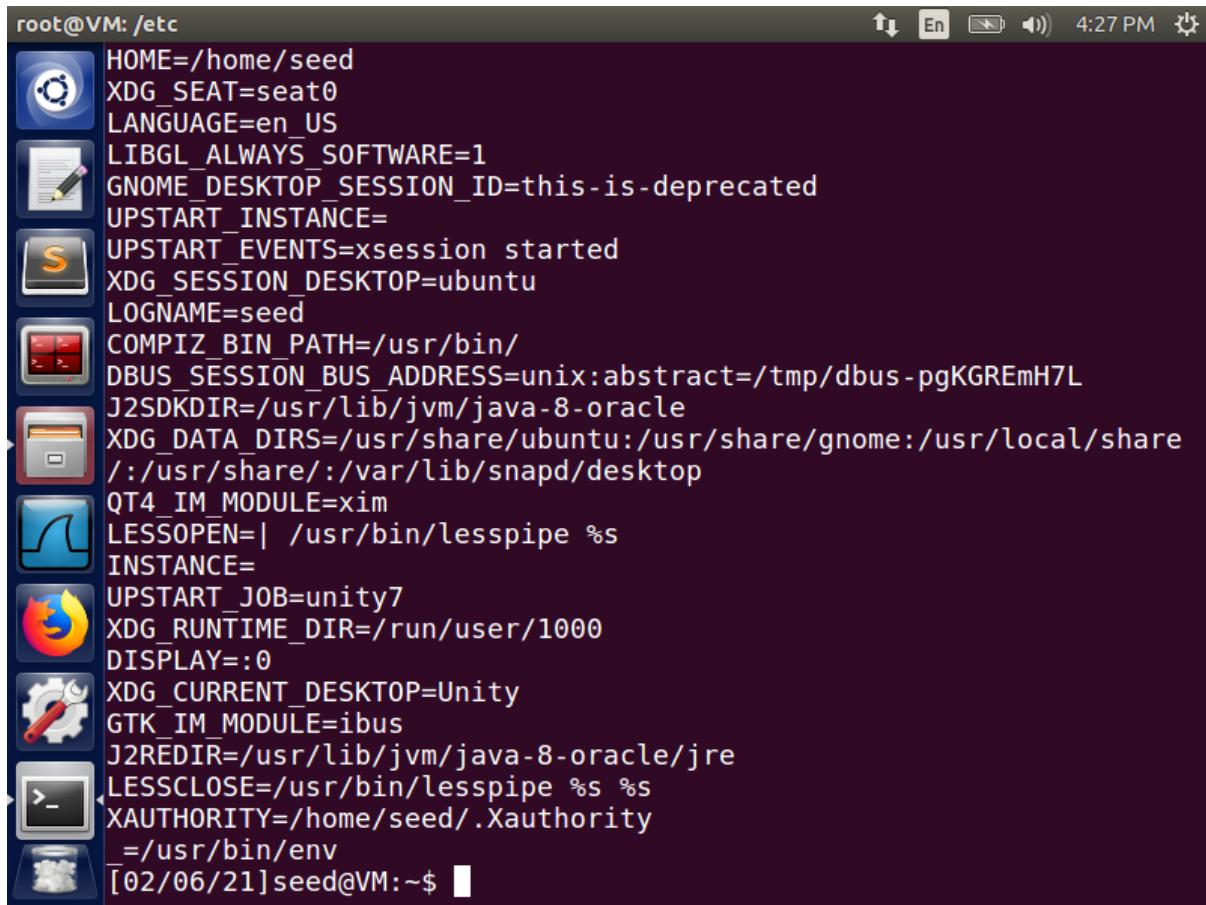
Task 1 – Manipulating Environment Variables

- Here, using the **env** command to display all the environment variables. It can also be noted that **printenv** command also does the same function as env (displaying environment variables)



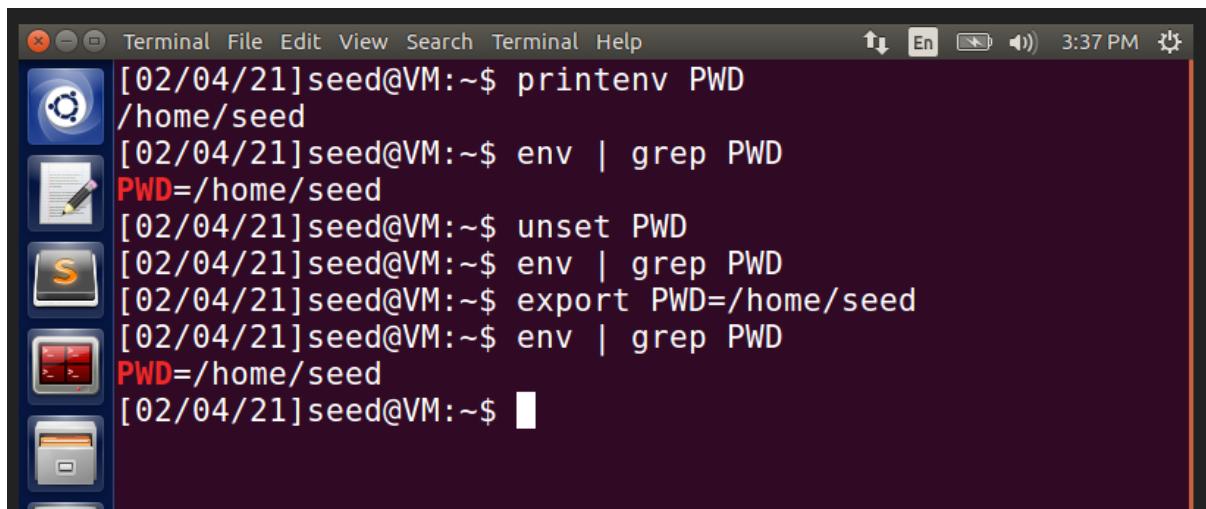
The screenshot shows a terminal window on a Linux desktop. The title bar reads "Terminal". The window contains the output of the "env" command, listing numerous environment variables. Some of the visible variables include XDG_VTNR, XDG_SESSION_ID, XDG_GREETER_DATA_DIR, CLUTTER_IM_MODULE, SESSION, ANDROID_HOME, GPG_AGENT_INFO, TERM, VTE_VERSION, XDG_MENU_PREFIX, SHELL, DERBY_HOME, QT_LINUX_ACCESSIBILITY_ALWAYS_ON, LD_PRELOAD, WINDOWID, UPSTART_SESSION, GNOME_KEYRING_CONTROL, GTK_MODULES, USER, and LS_COLORS. The terminal has a dark background with light-colored text. Icons for various desktop applications are visible along the left edge of the window.

```
[02/06/21] seed@VM:~$ env
XDG_VTNR=7
XDG_SESSION_ID=c1
XDG_GREETER_DATA_DIR=/var/lib/lightdm-data/seed
CLUTTER_IM_MODULE=xim
SESSION=ubuntu
ANDROID_HOME=/home/seed/android/android-sdk-linux
GPG_AGENT_INFO=/home/seed/.gnupg/S.gpg-agent:0:1
TERM=xterm-256color
VTE_VERSION=4205
XDG_MENU_PREFIX=gnome-
SHELL=/bin/bash
DERBY_HOME=/usr/lib/jvm/java-8-oracle/db
QT_LINUX_ACCESSIBILITY_ALWAYS_ON=1
LD_PRELOAD=./libmylib.so.1.0.1
WINDOWID=62914570
UPSTART_SESSION=unix:abstract=/com/ubuntu/upstart-session/1000/14
09
GNOME_KEYRING_CONTROL=
GTK_MODULES=gail:atk-bridge:unity-gtk-module
USER=seed
LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35
:bd=40;33;01:cd=40;33;01:or=40;31;01:mi=00:su=37;41:sg=30;43:ca=3
0;41:tw=30;42:ow=34;42:st=37;44:ex=01;32:*.tar=01;31:*.tgz=01;31
:*.arc=01;31:*.arj=01;31:*.taz=01;31:*.lha=01;31:*.lz4=01;31:*.lzh
=01;31:*.lzma=01;31:*.tlz=01;31:*.txz=01;31:*.tzo=01;31:*.t7z=01;
31:*.zip=01;31:*.Z=01;31:*.dz=01;31:*.gz=01;31:*.lrz=01
```



A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window. The terminal window has a dark background and contains the output of the command `env`. The output lists various environment variables, including `HOME=/home/seed`, `XDG_SEAT=seat0`, `LANGUAGE=en_US`, `LIBGL_ALWAYS_SOFTWARE=1`, `GNOME_DESKTOP_SESSION_ID=this-is-deprecated`, `UPSTART_INSTANCE=`, `UPSTART_EVENTS=xsession started`, `XDG_SESSION_DESKTOP=ubuntu`, `LOGNAME=seed`, `COMPIZ_BIN_PATH=/usr/bin/`, `DBUS_SESSION_BUS_ADDRESS=unix:abstract=/tmp/dbus-pgKGREmH7L`, `J2SDKDIR=/usr/lib/jvm/java-8-oracle`, `XDG_DATA_DIRS=/usr/share/ubuntu:/usr/share/gnome:/usr/local/share/:/usr/share/:/var/lib/snapd/desktop`, `QT4_IM_MODULE=xim`, `LESSOPEN=| /usr/bin/lesspipe %s`, `INSTANCE=`, `UPSTART_JOB=unity7`, `XDG_RUNTIME_DIR=/run/user/1000`, `DISPLAY=:0`, `XDG_CURRENT_DESKTOP=Unity`, `GTK_IM_MODULE=ibus`, `J2REDIR=/usr/lib/jvm/java-8-oracle/jre`, `LESSCLOSE=/usr/bin/lesspipe %s %s`, `XAUTHORITY=/home/seed/.Xauthority`, and `=/usr/bin/env`. The terminal prompt at the bottom is `[02/06/21] seed@VM:~$`.

- Using the `env | grep` with a variable next to the command will display all the substring which contains the variable. For example, in the above image, using the command `env | grep PWD` will display all the substrings in which PWD is present.



A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window. The terminal window has a dark background and contains the output of several commands related to the `PWD` environment variable. The sequence of commands is:
`[02/04/21]seed@VM:~$ printenv PWD`
`/home/seed`
`[02/04/21]seed@VM:~$ env | grep PWD`
`PWD=/home/seed`
`[02/04/21]seed@VM:~$ unset PWD`
`[02/04/21]seed@VM:~$ env | grep PWD`
`[02/04/21]seed@VM:~$ export PWD=/home/seed`
`[02/04/21]seed@VM:~$ env | grep PWD`
`PWD=/home/seed`
The terminal prompt at the bottom is `[02/04/21] seed@VM:~$`.

- `unset` command deletes the environment variable. As seen in the above image, after unsetting the `PWD` variable, if we try to view the variable using `env` command, it does not display anything.
- `export` command is used to set the environment variable. In the above image, using `export` command, environment variable is set, which can later be viewed using `env` command.

Task 2 – Passing Environment Variables from Parent Process to Child Process

- For this task, the given program is stored as **Task2.c**, compiled and its output is stored in file **task2prog**. The environment variables from this, is stored into **task2op** file. (Child Process)

```
[02/04/21]seed@VM:~$ gcc Task2.c -o task2prog
[02/04/21]seed@VM:~$ ./task2prog > task2op
```

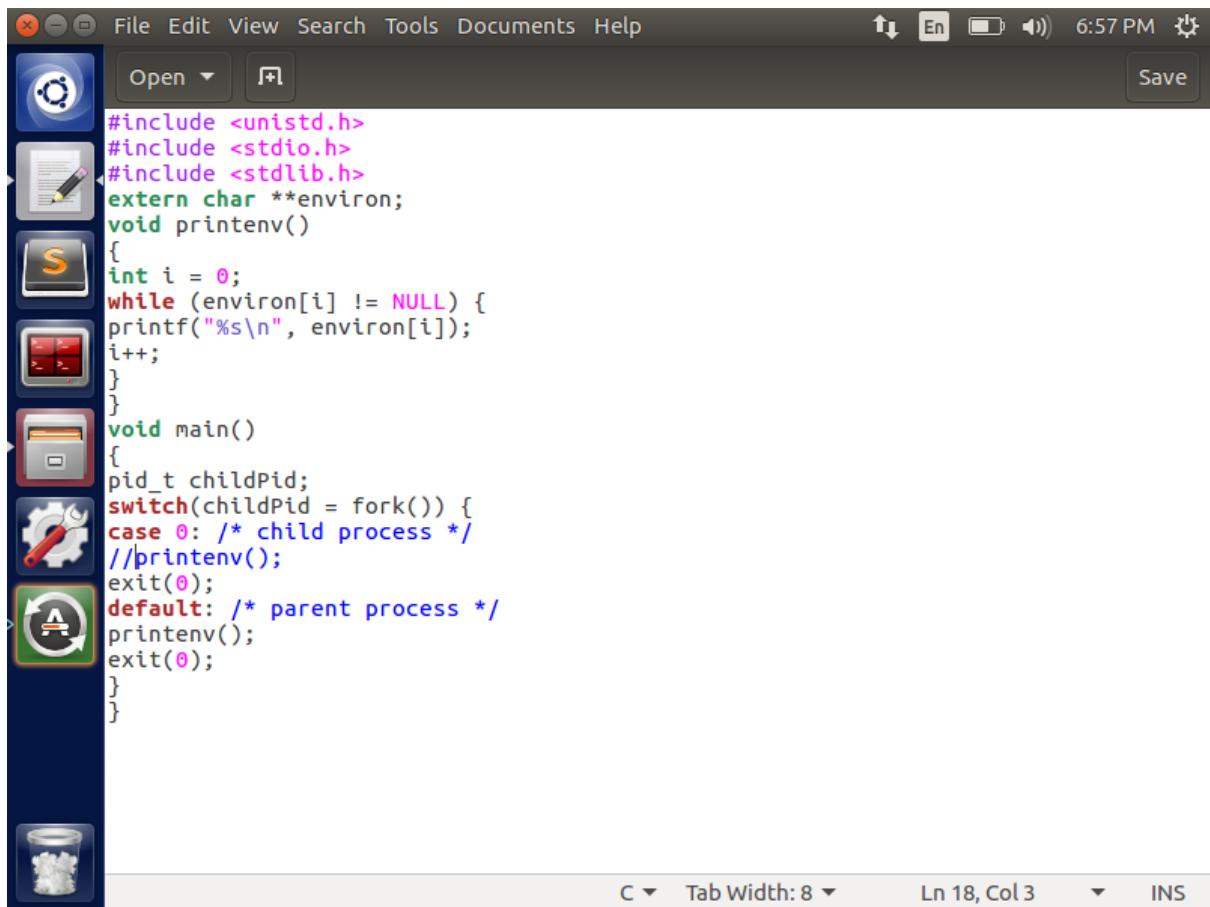
A screenshot of a terminal window titled "Terminal". The window shows the command "gcc Task2.c -o task2prog" being run at the prompt [02/04/21]seed@VM:~\$. The output of the command, "task2prog", is then piped into a file named "task2op". The terminal window has a dark theme with white text.

- In the task2op file, in the 70th line of the file, it reads **_=./task2prog**

```
GDMSESSION=ubuntu
SESSIONTYPE=gnome-session
GTK2_MODULES=overlay-scrollbar
SHLVL=1
HOME=/home/seed
XDG_SEAT=seat0
LANGUAGE=en_US
LIBGL_ALWAYS_SOFTWARE=1
GNOME_DESKTOP_SESSION_ID=this-is-deprecated
UPSTART_INSTANCE=
UPSTART_EVENTS=xsession started
XDG_SESSION_DESKTOP=ubuntu
LOGNAME=seed
COMPIZ_BIN_PATH=/usr/bin/
DBUS_SESSION_BUS_ADDRESS=unix:abstract=/tmp/dbus-4NNfbG1B00
J2SDKDIR=/usr/lib/jvm/java-8-oracle
XDG_DATA_DIRS=/usr/share/ubuntu:/usr/share/gnome:/usr/local/share/:/usr/share/::
var/lib/snapd/desktop
QT4_IM_MODULE=xim
LESSOPEN=| /usr/bin/lesspipe %
INSTANCE=
UPSTART_JOB=unity7
XDG_RUNTIME_DIR=/run/user/1000
DISPLAY=:0
XDG_CURRENT_DESKTOP=Unity
GTK_IM_MODULE=ibus
J2REDIR=/usr/lib/jvm/java-8-oracle/jre
LESSCLOSE=/usr/bin/lesspipe %s %
XAUTHORITY=/home/seed/.xauthority
_=./task2prog
```

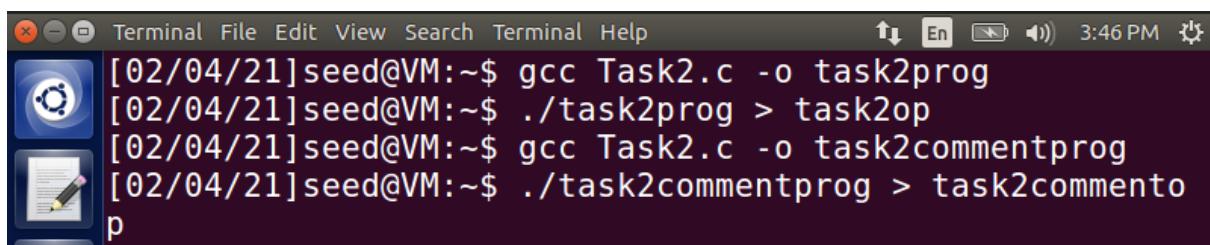
A screenshot of a gedit text editor window titled "task2op (~/) - gedit". The window displays a list of environment variables. The 70th line of the file contains the command **_=./task2prog**. The gedit interface includes a toolbar with icons for Open, Save, and other file operations, and a status bar at the bottom showing "Plain Text", "Tab Width: 8", "Ln 1, Col 1", and "INS".

- Next, we comment out the **printenv** of child process and uncomment the **printenv** of Parent process as seen below in the code. Then run the program again.



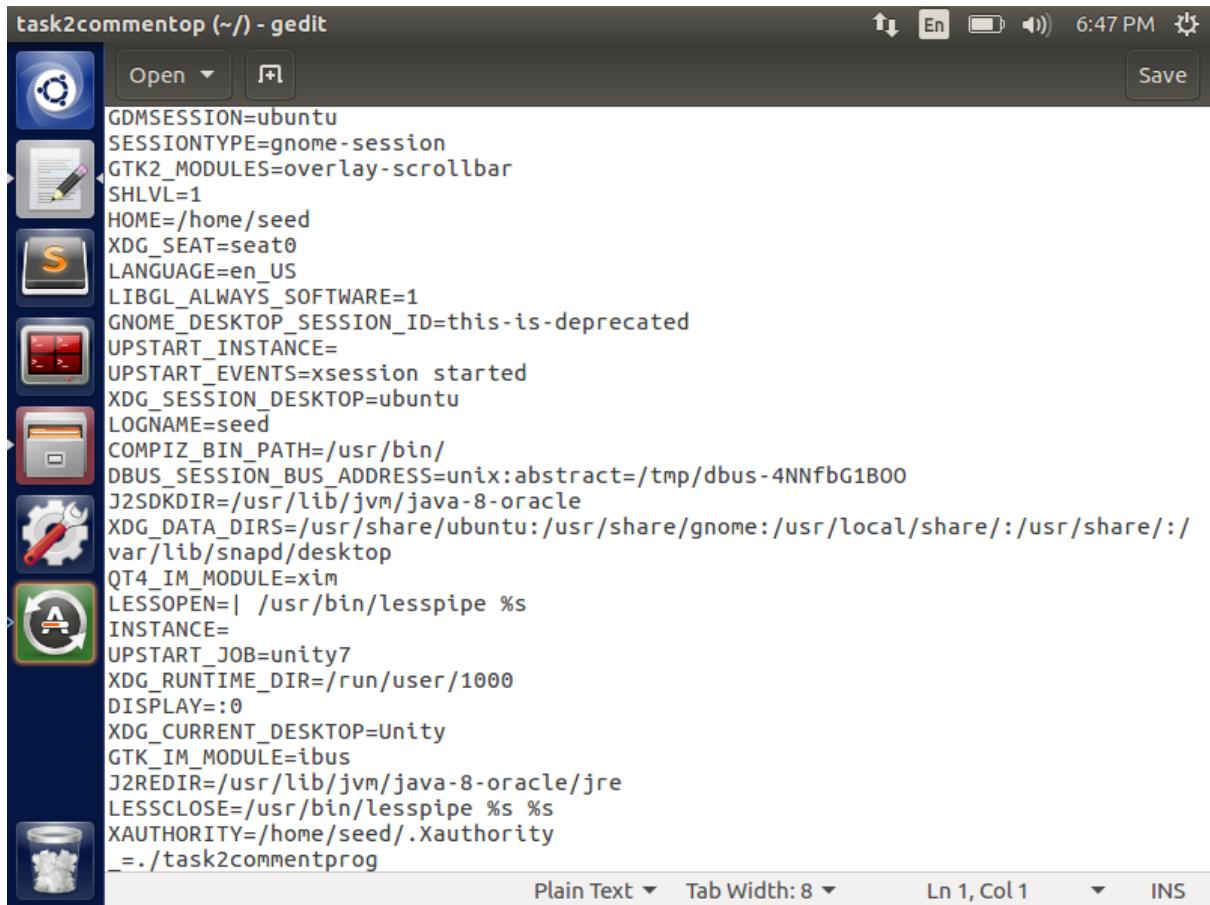
```
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
extern char **environ;
void printenv()
{
int i = 0;
while (environ[i] != NULL) {
printf("%s\n", environ[i]);
i++;
}
}
void main()
{
pid_t childPid;
switch(childPid = fork()) {
case 0: /* child process */
//printenv();
exit(0);
default: /* parent process */
printenv();
exit(0);
}
}
```

- Task2.c is compiled again and stored as **task2commentprog** and its environment variables are stored in **task2commentop** file.



```
[02/04/21]seed@VM:~$ gcc Task2.c -o task2prog
[02/04/21]seed@VM:~$ ./task2prog > task2op
[02/04/21]seed@VM:~$ gcc Task2.c -o task2commentprog
[02/04/21]seed@VM:~$ ./task2commentprog > task2commento
p
```

- In the task2commentop file, the 70th line reads as **_=./task2commentprog**



The screenshot shows a Gedit text editor window with the title "task2commentop (~/) - gedit". The content of the editor is a list of environment variables:

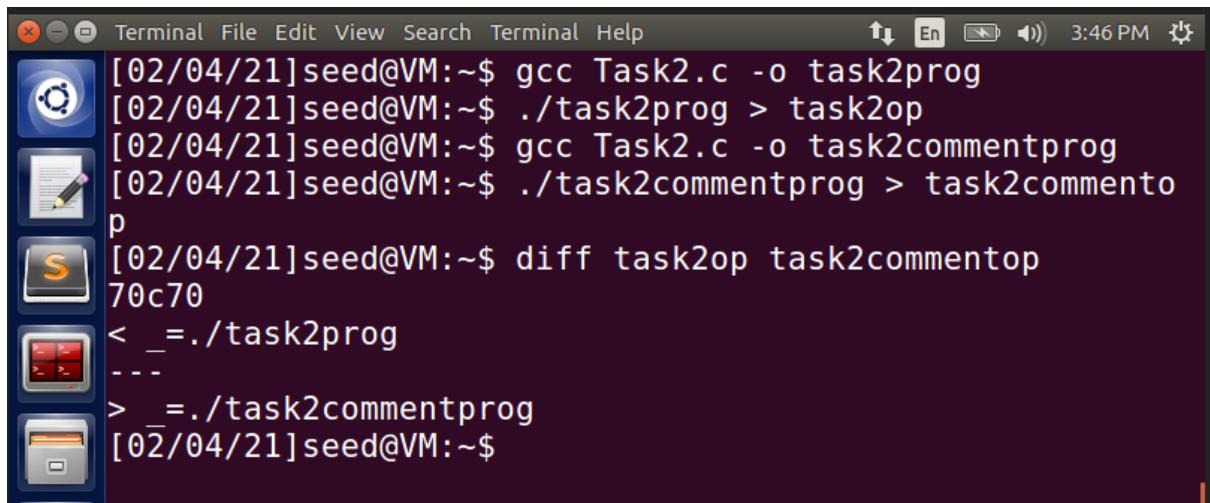
```

GDMSESSION=ubuntu
SESSIONTYPE=gnome-session
GTK2_MODULES=overlay-scrollbar
SHLVL=1
HOME=/home/seed
XDG_SEAT=seat0
LANGUAGE=en_US
LIBGL_ALWAYS_SOFTWARE=1
GNOME_DESKTOP_SESSION_ID=this-is-deprecated
UPSTART_INSTANCE=
UPSTART_EVENTS=xsession started
XDG_SESSION_DESKTOP=ubuntu
LOGNAME=seed
COMPIZ_BIN_PATH=/usr/bin/
DBUS_SESSION_BUS_ADDRESS=unix:abstract=/tmp/dbus-4NNfbG1B00
J2SDKDIR=/usr/lib/jvm/java-8-oracle
XDG_DATA_DIRS=/usr/share/ubuntu:/usr/share/gnome:/usr/local/share/:/usr/share/::
var/lib/snapd/desktop
QT4_IM_MODULE=xim
LESSOPEN=| /usr/bin/lesspipe %
INSTANCE=
UPSTART_JOB=unity7
XDG_RUNTIME_DIR=/run/user/1000
DISPLAY=:0
XDG_CURRENT_DESKTOP=Unity
GTK_IM_MODULE=ibus
J2REDIR=/usr/lib/jvm/java-8-oracle/jre
LESSCLOSE=/usr/bin/lesspipe %s %
XAUTHORITY=/home/seed/.Xauthority
_=./task2commentprog

```

At the bottom of the editor, there are buttons for "Plain Text", "Tab Width: 8", "Ln 1, Col 1", and "INS".

- To compute the difference between the two files, we use **diff** command, which says **70c70**, as shown in the below image which means that the 70th line of child process is changed into 70th line of parent process. If the difference is compiled into the same file name, it would not result in any difference because even though they are from different process, they still point to the same output file.



The screenshot shows a terminal window with the title "Terminal". The user has run the following commands:

```

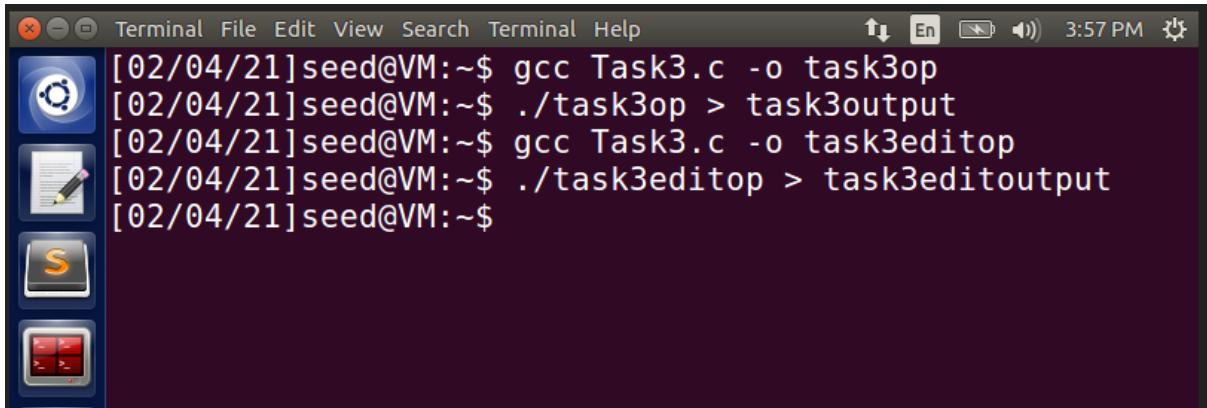
[02/04/21]seed@VM:~$ gcc Task2.c -o task2prog
[02/04/21]seed@VM:~$ ./task2prog > task2op
[02/04/21]seed@VM:~$ gcc Task2.c -o task2commentprog
[02/04/21]seed@VM:~$ ./task2commentprog > task2commentop
[02/04/21]seed@VM:~$ diff task2op task2commentop
70c70
< _=./task2prog
---
> _=./task2commentprog
[02/04/21]seed@VM:~$

```

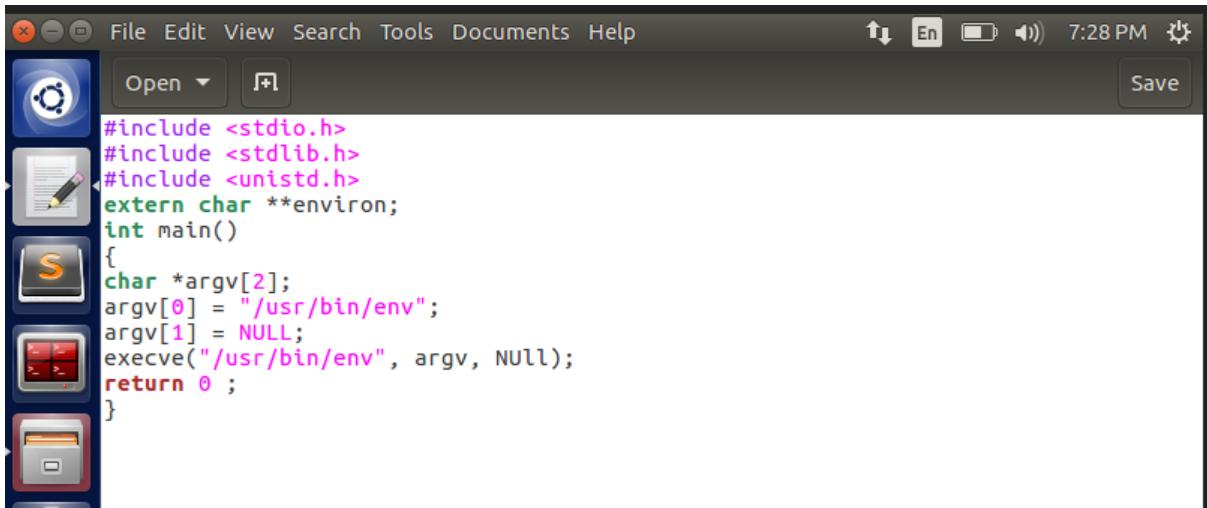
- This shows that the **printenv** passed from whichever function gets populated into the output file. Here, the environment variables are passed from Parent process (default) into child process.

Task 3: Environment Variables and execve()

- For this task, the given program is stored as **Task3.c**, compiled and its output is stored in file **task3op**. The environment variables from this, is stored into **task3output** file.

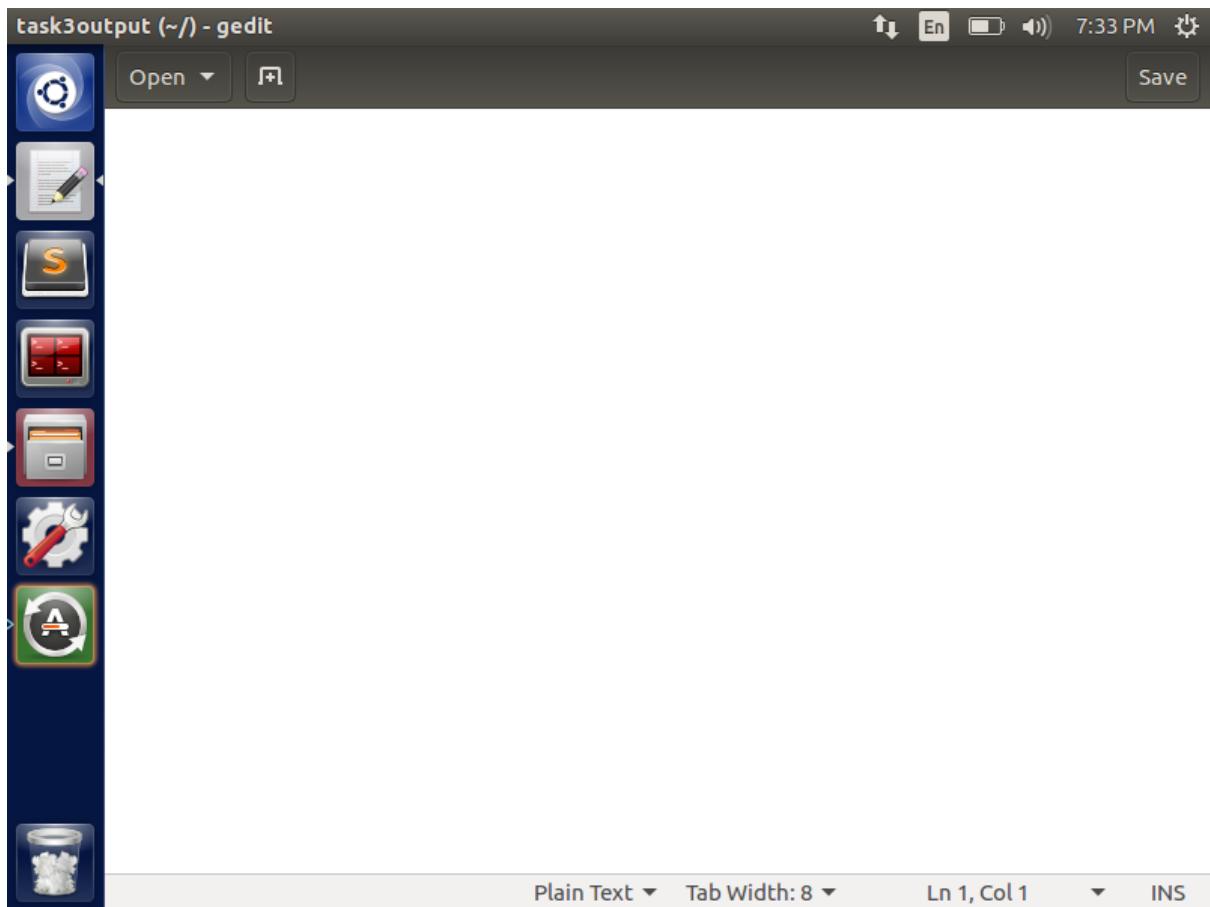


```
[02/04/21]seed@VM:~$ gcc Task3.c -o task3op
[02/04/21]seed@VM:~$ ./task3op > task3output
[02/04/21]seed@VM:~$ gcc Task3.c -o task3editop
[02/04/21]seed@VM:~$ ./task3editop > task3editoutput
[02/04/21]seed@VM:~$
```



```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
extern char **environ;
int main()
{
    char *argv[2];
    argv[0] = "/usr/bin/env";
    argv[1] = NULL;
    execve("/usr/bin/env", argv, NULL);
    return 0 ;
}
```

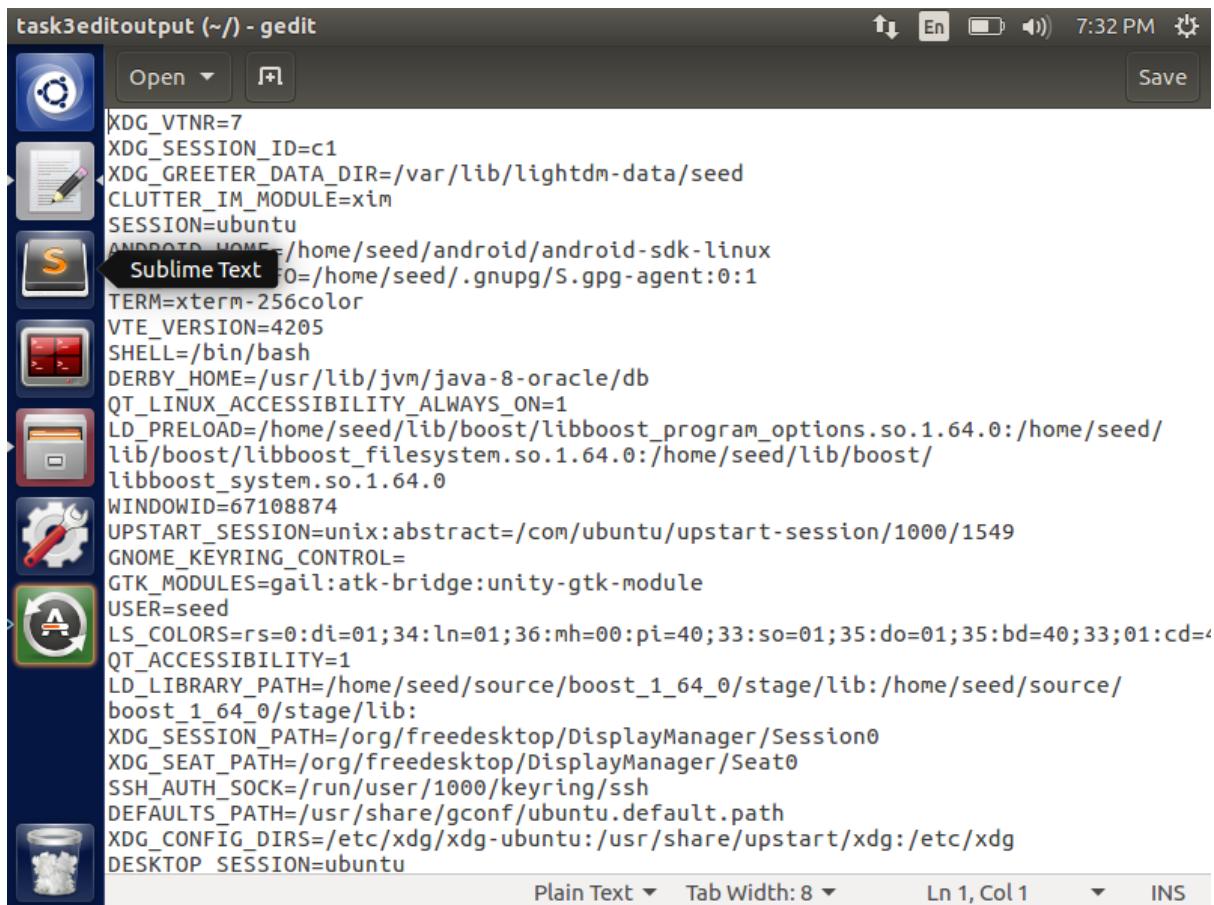
- In the task3output file, there is no output to display as seen below. An empty file.



- Now, the Task3.c code is modified and the modified **execve** function is shown below, where the argument **environ** is passed.

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
extern char **environ;
int main()
{
    char *argv[2];
    argv[0] = "/usr/bin/env";
    argv[1] = NULL;
    execve("/usr/bin/env", argv, environ);
    return 0 ;
}
```

Now the output file is populated with environment variables.



A screenshot of a terminal window titled "task3editoutput (~) - gedit". The window displays a list of environment variables. A mouse cursor is hovering over the entry "Sublime Text 3=0=/home/seed/.gnupg/S.gpg-agent:0:1". The terminal interface includes standard Linux icons in the sidebar and a toolbar with "Open", "Save", and other options. The status bar at the bottom shows "Plain Text", "Tab Width: 8", "Ln 1, Col 1", and "INS".

```
XDG_VTNR=7
XDG_SESSION_ID=c1
XDG_GREETER_DATA_DIR=/var/lib/lightdm-data/seed
CLUTTER_IM_MODULE=xim
SESSION=ubuntu
ANDROID_HOME=/home/seed/android/android-sdk-linux
Sublime Text 3=0=/home/seed/.gnupg/S.gpg-agent:0:1
TERM=xterm-256color
VTE_VERSION=4205
SHELL=/bin/bash
DERBY_HOME=/usr/lib/jvm/java-8-oracle/db
QT_LINUX_ACCESSIBILITY_ALWAYS_ON=1
LD_PRELOAD=/home/seed/lib/boost/libboost_program_options.so.1.64.0:/home/seed/lib/boost/libboost_filesystem.so.1.64.0:/home/seed/lib/boost/libboost_system.so.1.64.0
WINDOWID=67108874
UPSTART_SESSION=unix:abstract=/com/ubuntu/upstart-session/1000/1549
GNOME_KEYRING_CONTROL=
GTK_MODULES=gail:atk-bridge:unity-gtk-module
USER=seed
LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd=4
QT_ACCESSIBILITY=1
LD_LIBRARY_PATH=/home/seed/source/boost_1_64_0/stage/lib:/home/seed/source/boost_1_64_0/stage/lib:
XDG_SESSION_PATH=/org/freedesktop/DisplayManager/Session0
XDG_SEAT_PATH=/org/freedesktop/DisplayManager/Seat0
SSH_AUTH_SOCK=/run/user/1000/keyring/ssh
DEFAULTS_PATH=/usr/share/gconf/ubuntu.default.path
XDG_CONFIG_DIRS=/etc/xdg/xdg-ubuntu:/usr/share/upstart/xdg:/etc/xdg
DESKTOP_SESSION=ubuntu
```

- This is because, in the first function, the argument **NULL** was passed to `execve()` function, which means there is nothing for the function to display. But after edit, `environ` (environment variables) were passed as arguments to the same function, which populates the same into the output file.

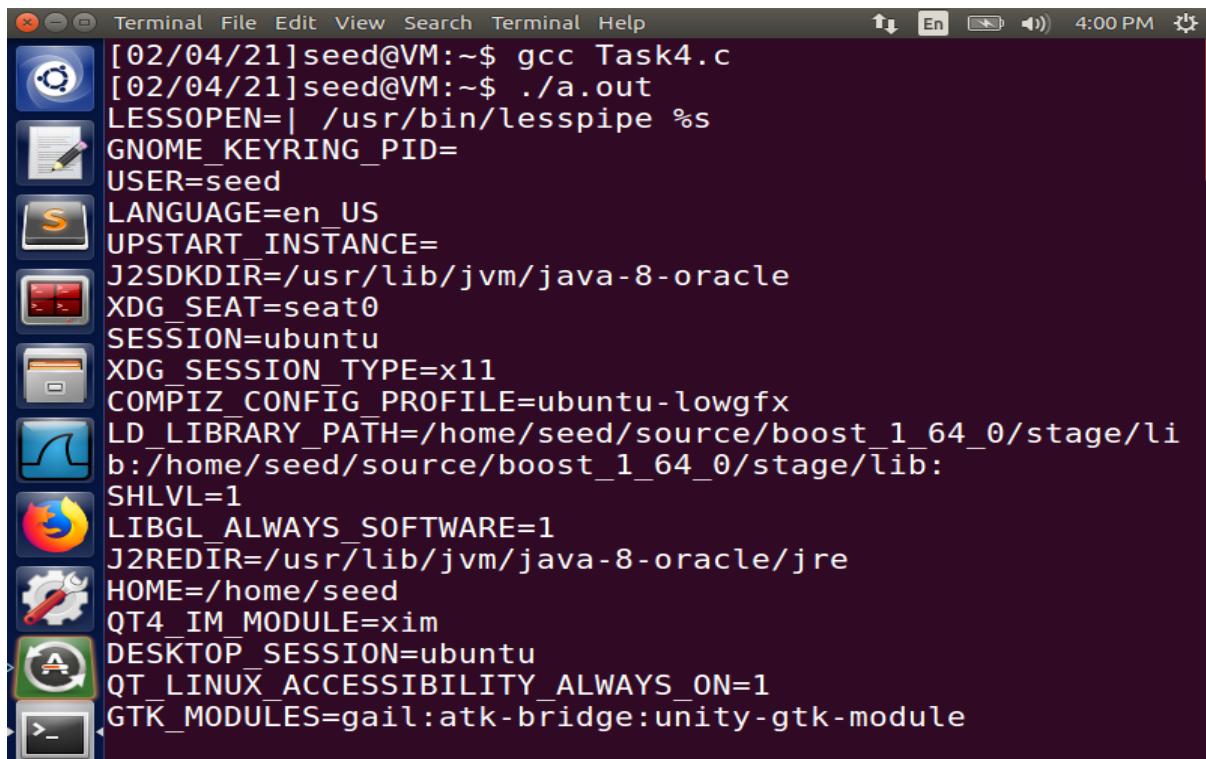
Task 4: Environment Variables and system()

- For this task, the given program is stored as Task4.c, compiled and run.

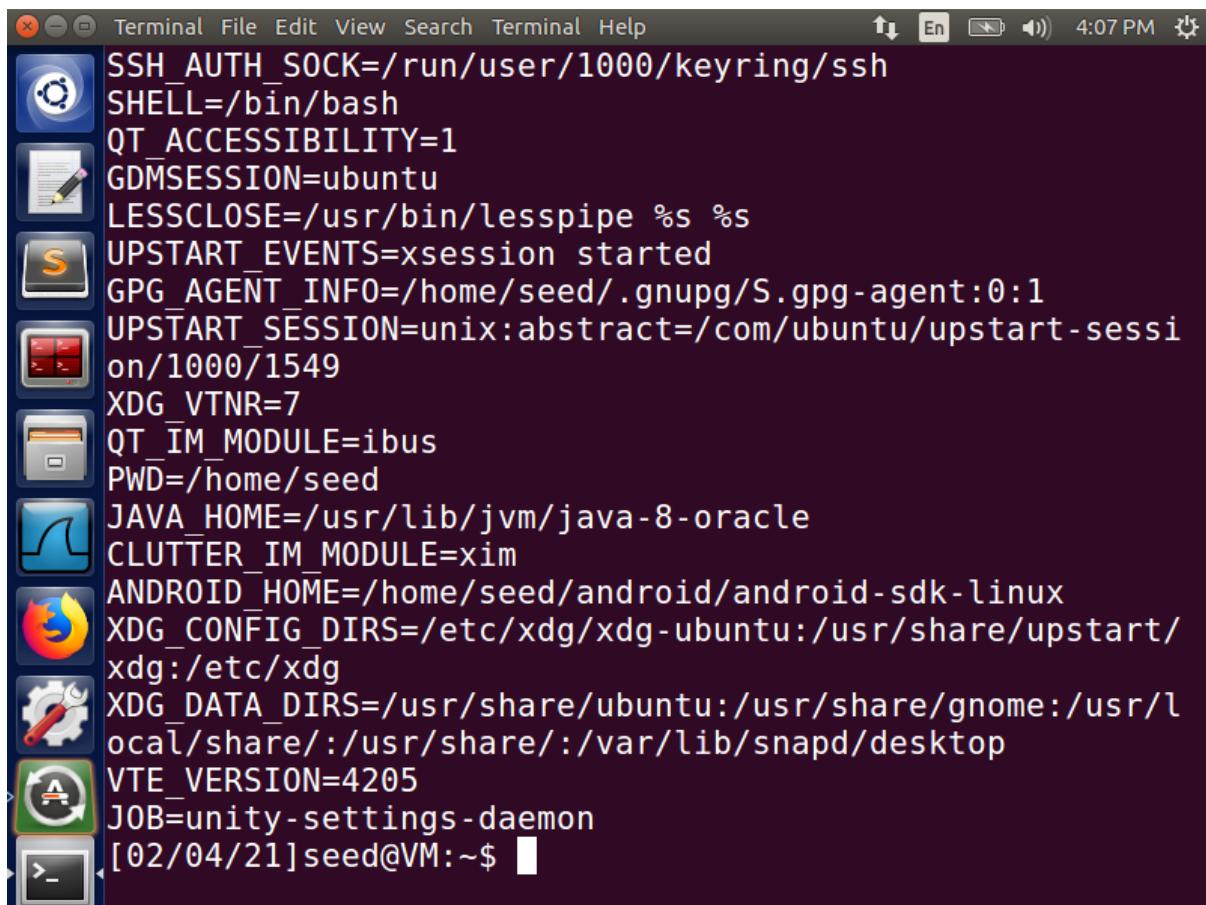


A screenshot of a terminal window titled "Task4.c (~) - gedit". The window displays a simple C program. It includes the `<stdio.h>` and `<stdlib.h>` headers, defines a main function, and uses the `system` function to execute the command `"/usr/bin/env"`. The terminal interface includes standard Linux icons in the sidebar and a toolbar with "Open", "Save", and other options. The status bar at the bottom shows "Plain Text", "Tab Width: 8", "Ln 1, Col 1", and "INS".

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    system("/usr/bin/env");
    return 0 ;
}
```



```
[02/04/21]seed@VM:~$ gcc Task4.c
[02/04/21]seed@VM:~$ ./a.out
LESSOPEN=| /usr/bin/lesspipe %s
GNOME_KEYRING_PID=
USER=seed
LANGUAGE=en_US
UPSTART_INSTANCE=
J2SDKDIR=/usr/lib/jvm/java-8-oracle
XDG_SEAT=seat0
SESSION=ubuntu
XDG_SESSION_TYPE=x11
COMPIZ_CONFIG_PROFILE=ubuntu-lowgfx
LD_LIBRARY_PATH=/home/seed/source/boost_1_64_0/stage/lib:/home/seed/source/boost_1_64_0/stage/lib:
SHLVL=1
LIBGL_ALWAYS_SOFTWARE=1
J2REDIR=/usr/lib/jvm/java-8-oracle/jre
HOME=/home/seed
QT4_IM_MODULE=xim
DESKTOP_SESSION=ubuntu
QT_LINUX_ACCESSIBILITY_ALWAYS_ON=1
GTK_MODULES=gail:atk-bridge:unity-gtk-module
```

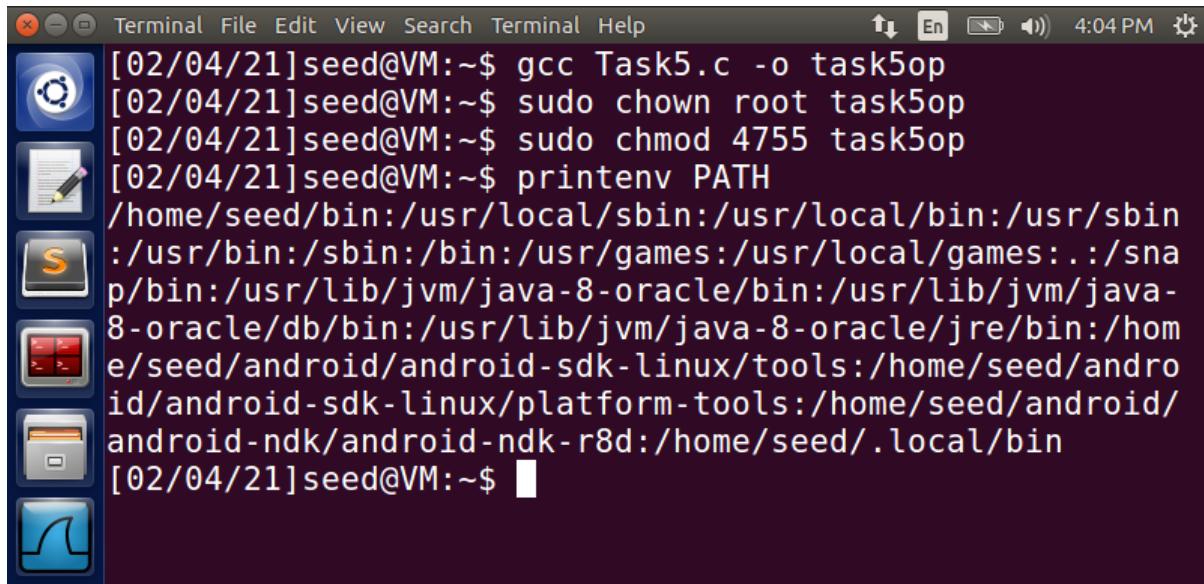


```
SSH_AUTH_SOCK=/run/user/1000/keyring/ssh
SHELL=/bin/bash
QT_ACCESSIBILITY=1
GDMSESSION=ubuntu
LESSCLOSE=/usr/bin/lesspipe %s %s
UPSTART_EVENTS=xsession started
GPG_AGENT_INFO=/home/seed/.gnupg/S.gpg-agent:0:1
UPSTART_SESSION=unix:abstract=/com/ubuntu/upstart-session/1000/1549
XDG_VTNR=7
QT_IM_MODULE=ibus
PWD=/home/seed
JAVA_HOME=/usr/lib/jvm/java-8-oracle
CLUTTER_IM_MODULE=xim
ANDROID_HOME=/home/seed/android/android-sdk-linux
XDG_CONFIG_DIRS=/etc/xdg/xdg-ubuntu:/usr/share/upstart/xdg:/etc/xdg
XDG_DATA_DIRS=/usr/share/ubuntu:/usr/share/gnome:/usr/local/share/:/usr/share/:/var/lib/snapd/desktop
VTE_VERSION=4205
JOB=unity-settings-daemon
[02/04/21]seed@VM:~$
```

Even though there is nothing explicitly passed from our side to the program, just compiling the program and displaying the output, display the following as showing in the images above. This is because, the **system("usr/bin/env")** function loads the environment variables implicitly during run time.

Task 5: Environment Variables and SET-UID programs

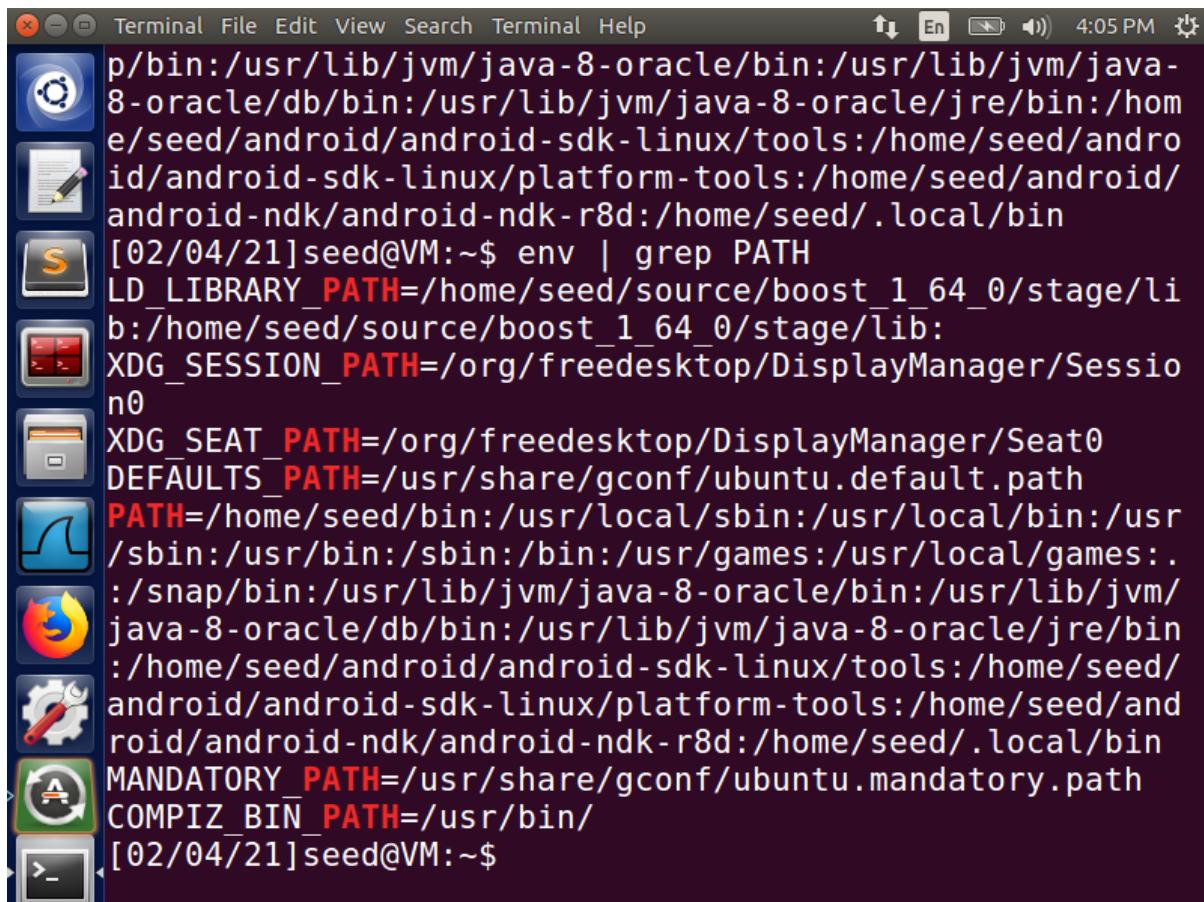
- For this task, the given program is stored as Task5.c, compiled and output is stored in task5op. Now, to access the op file the ownership of the file is changed and changed to root using **chown root** command.



A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window. The terminal window title is "Terminal". The terminal content shows the following commands and their output:

```
[02/04/21]seed@VM:~$ gcc Task5.c -o task5op
[02/04/21]seed@VM:~$ sudo chown root task5op
[02/04/21]seed@VM:~$ sudo chmod 4755 task5op
[02/04/21]seed@VM:~$ printenv PATH
/home/seed/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin
:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:./sna
p/bin:/usr/lib/jvm/java-8-oracle/bin:/usr/lib/jvm/java-
8-oracle/db/bin:/usr/lib/jvm/java-8-oracle/jre/bin:/hom
e/seed/android/android-sdk-linux/tools:/home/seed/andro
id/android-sdk-linux/platform-tools:/home/seed/android/
android-ndk/android-ndk-r8d:/home/seed/.local/bin
[02/04/21]seed@VM:~$
```

- Here, using **env | grep PATH**, we check the **LD_LIBRARY_PATH** and it is seen that the child process inherits the **PATH** variable.



A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window. The terminal window title is "Terminal". The terminal content shows the following environment variables and their values:

```
p/bin:/usr/lib/jvm/java-8-oracle/bin:/usr/lib/jvm/java-
8-oracle/db/bin:/usr/lib/jvm/java-8-oracle/jre/bin:/hom
e/seed/android/android-sdk-linux/tools:/home/seed/andro
id/android-sdk-linux/platform-tools:/home/seed/android/
android-ndk/android-ndk-r8d:/home/seed/.local/bin
[02/04/21]seed@VM:~$ env | grep PATH
LD_LIBRARY_PATH=/home/seed/source/boost_1_64_0/stage/li
b:/home/seed/source/boost_1_64_0/stage/lib:
XDG_SESSION_PATH=/org/freedesktop/DisplayManager/Sessio
n0
XDG_SEAT_PATH=/org/freedesktop/DisplayManager/Seat0
DEFAULTS_PATH=/usr/share/gconf/ubuntu.default.path
PATH=/home/seed/bin:/usr/local/sbin:/usr/local/bin:/usr
/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:.
:/snap/bin:/usr/lib/jvm/java-8-oracle/bin:/usr/lib/jvm/
java-8-oracle/db/bin:/usr/lib/jvm/java-8-oracle/jre/bin
:/home/seed/android/android-sdk-linux/tools:/home/seed/
android/android-sdk-linux/platform-tools:/home/seed/and
roid/android-ndk/android-ndk-r8d:/home/seed/.local/bin
MANDATORY_PATH=/usr/share/gconf/ubuntu.mandatory.path
COMPIZ_BIN_PATH=/usr/bin/
[02/04/21]seed@VM:~$
```

- But if we check for the LD PATH variable in the output, it is not present. This is because of ownership issues where the child process may not be able to inherit all the environment variables from the parent and that is the reason why LD_LIBRARY_PATH is not inherited in the output file.

```

task5output (~/) - gedit
Open Save
QT_QPA_PLATFORM=appmenu-qts
XDG_SESSION_TYPE=x11
PWD=/home/seed
JOB=unity-settings-daemon
XMODIFIERS=@im=ibus
JAVA_HOME=/usr/lib/jvm/java-8-oracle
GNOME_KEYRING_PID=
LANG=en_US.UTF-8
GDM_LANG=en_US
MANDATORY_PATH=/usr/share/gconf/ubuntu.mandatory.path
COMPIZ_CONFIG_PROFILE=ubuntu-lowgfx
IM_CONFIG_PHASE=1
GDMSESSION=ubuntu
SESSIONTYPE=gnome-session
GTK2_MODULES=overlay-scrollbar
SHLVL=1
HOME=/home/seed
XDG_SEAT=seat0
LANGUAGE=en_US
LIBGL_ALWAYS_SOFTWARE=1
GNOME_DESKTOP_SESSION_ID=this-is-deprecated
UPSTART_INSTANCE=
UPSTART_EVENTS=xsession started
XDG_SESSION_DESKTOP=ubuntu
LOGNAME=seed
COMPIZ_BIN_PATH=/usr/bin/
DBUS_SESSION_BUS_ADDRESS=unix:abstract=/tmp/dbus-4NNfbG1B00
J2SDKDIR=/usr/lib/jvm/java-8-oracle
XDG_DATA_DIRS=/usr/share/ubuntu:/usr/share/gnome:/usr/local/share/:/usr/share/:/var/lib/snapd/desktop
QT_IM_MODULE=xim

```

Plain Text Tab Width: 8 ▾ Ln 34, Col 37 ▾ INS

Task 6: The PATH Environment Variable and SET-UID Programs

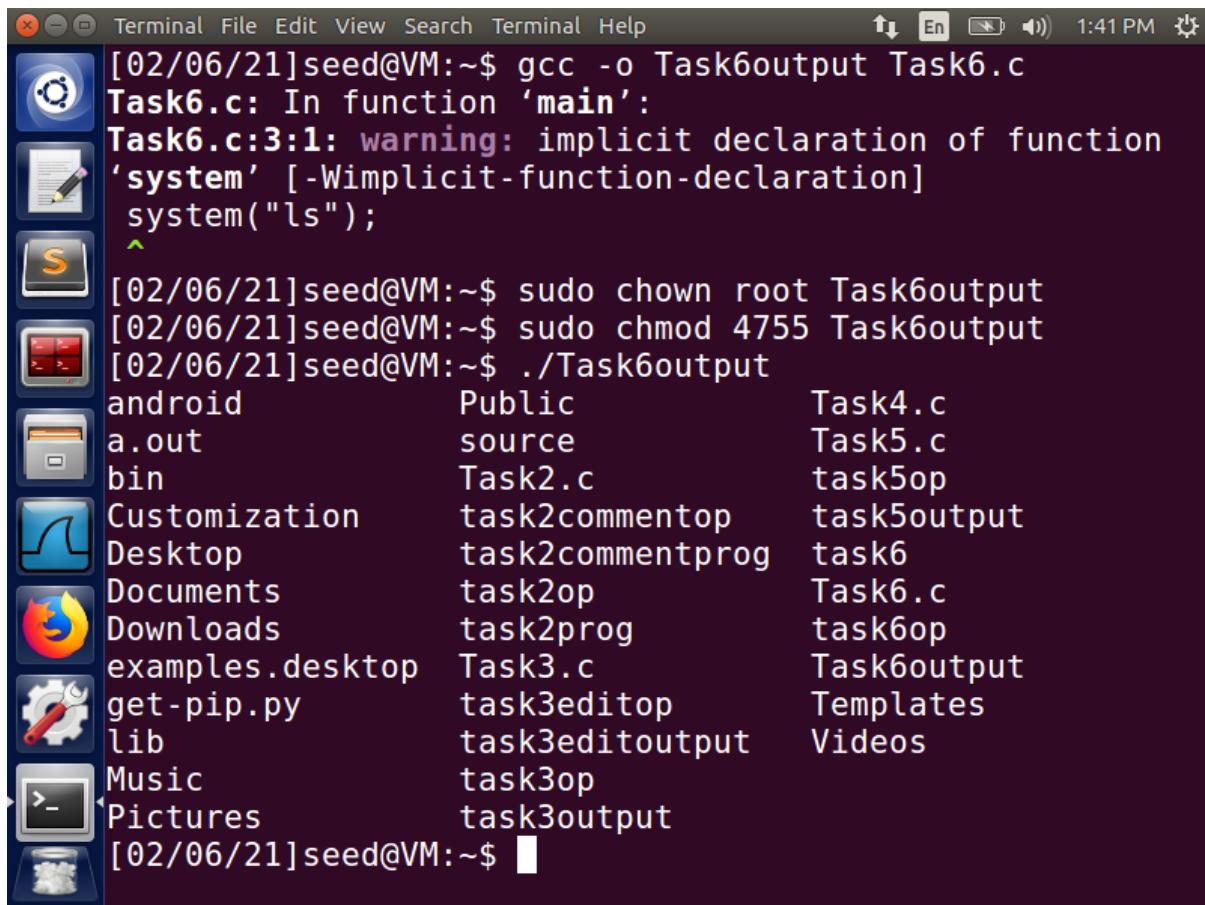
- For this task, the given program is stored as Task6.c, compiled and output is stored in Task6output. Now, to access the op file the ownership of the file is changed and changed to root using **chown root** command.

```

Task6.c (~/) - gedit
Open Save
int main()
{
    system("ls");
    return 0;
}

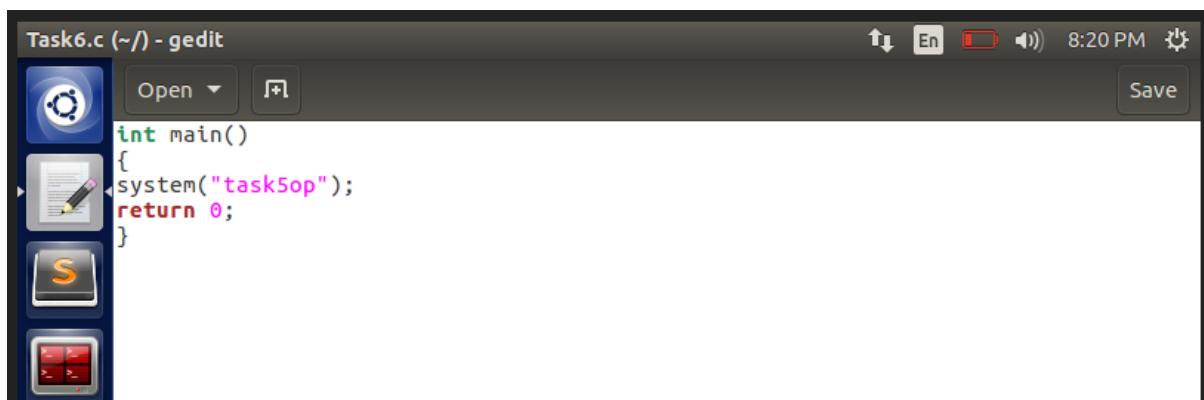
```

- After providing ownership to the file, on running the output file, it displays all the contents of the file, since the system() has ls.
- The output is displayed below as shown:



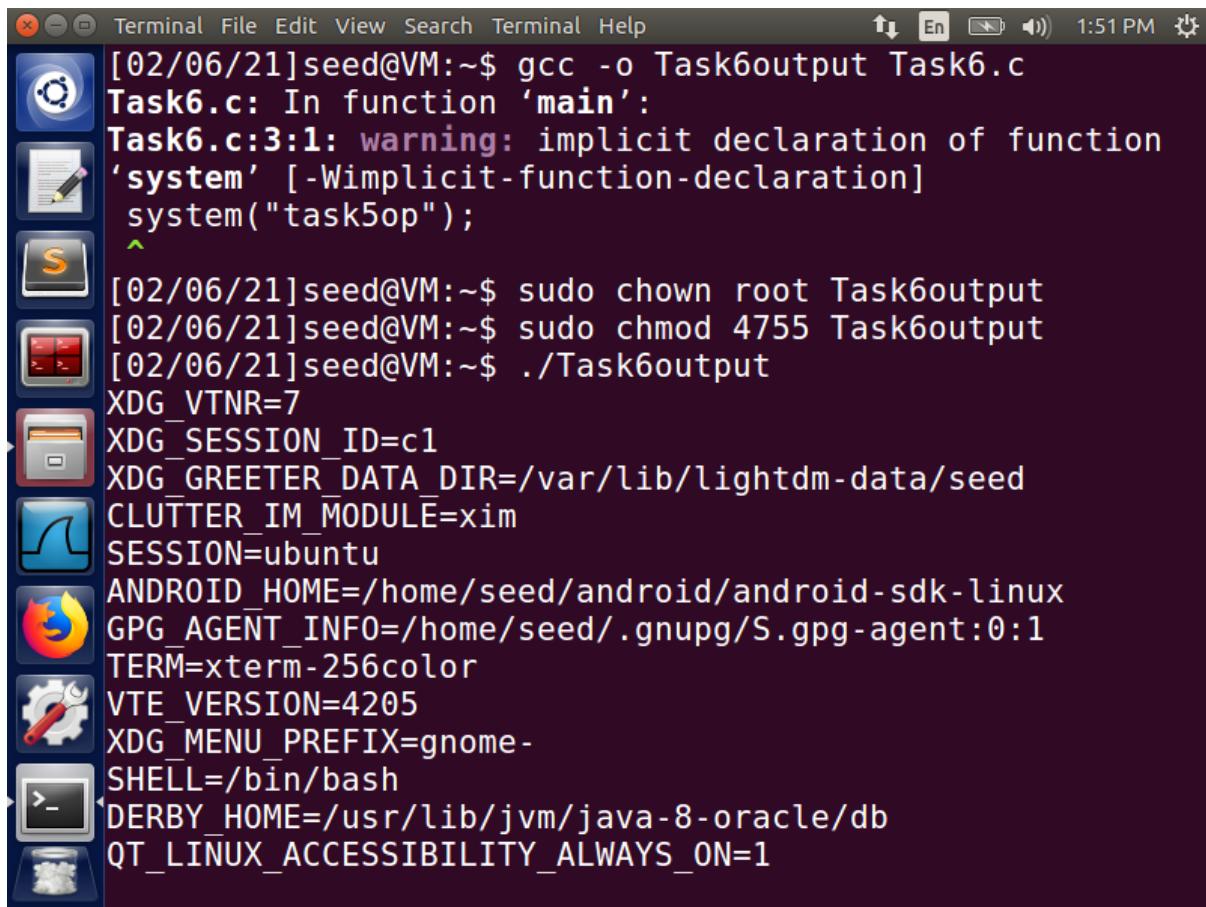
```
[02/06/21]seed@VM:~$ gcc -o Task6output Task6.c
Task6.c: In function 'main':
Task6.c:3:1: warning: implicit declaration of function
'system' [-Wimplicit-function-declaration]
system("ls");
^
[02/06/21]seed@VM:~$ sudo chown root Task6output
[02/06/21]seed@VM:~$ sudo chmod 4755 Task6output
[02/06/21]seed@VM:~$ ./Task6output
android           Public          Task4.c
a.out             source         Task5.c
bin               Task2.c        task5op
Customization     task2commentop task5output
Desktop           task2commentprog task6
Documents         task2op        Task6.c
Downloads         task2prog      task6op
examples.desktop  Task3.c        Task6output
get-pip.py        task3editop    Templates
lib               task3editoutput Videos
Music             task3op        task3output
Pictures          task3output
[02/06/21]seed@VM:~$
```

- Now, the contents of the file Task6.c is changed and “task5op” file is passed as an argument to the system() function.



```
Task6.c (~/) - gedit
Open ▾  Save
int main()
{
system("task5op");
return 0;
}
```

- Now, the code Task6.c is run again and the output is again overwritten into the file Task6output. Then the root privileges are provided to the file using **chown root** command.



A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window. The terminal window title is "Terminal". The terminal content shows the following:

```
[02/06/21]seed@VM:~$ gcc -o Task6output Task6.c
Task6.c: In function 'main':
Task6.c:3:1: warning: implicit declaration of function
'system' [-Wimplicit-function-declaration]
    system("task5op");
^
[02/06/21]seed@VM:~$ sudo chown root Task6output
[02/06/21]seed@VM:~$ sudo chmod 4755 Task6output
[02/06/21]seed@VM:~$ ./Task6output
XDG_VTNR=7
XDG_SESSION_ID=c1
XDG_GREETER_DATA_DIR=/var/lib/lightdm-data/seed
CLUTTER_IM_MODULE=xim
SESSION=ubuntu
ANDROID_HOME=/home/seed/android/android-sdk-linux
GPG_AGENT_INFO=/home/seed/.gnupg/S.gpg-agent:0:1
TERM=xterm-256color
VTE_VERSION=4205
XDG_MENU_PREFIX=gnome-
SHELL=/bin/bash
DERBY_HOME=/usr/lib/jvm/java-8-oracle/db
QT_LINUX_ACCESSIBILITY_ALWAYS_ON=1
```

- On running the Task6output file, it runs **task5op** and displays the environment variables of Task 5. Here, we use the `system()` function to run another program's output file and display the same instead of `ls` command. It can be observed that the output file is run only after providing the root privileges.

Task 7: The LD_PRELOAD Environment Variable and SET-UID Programs

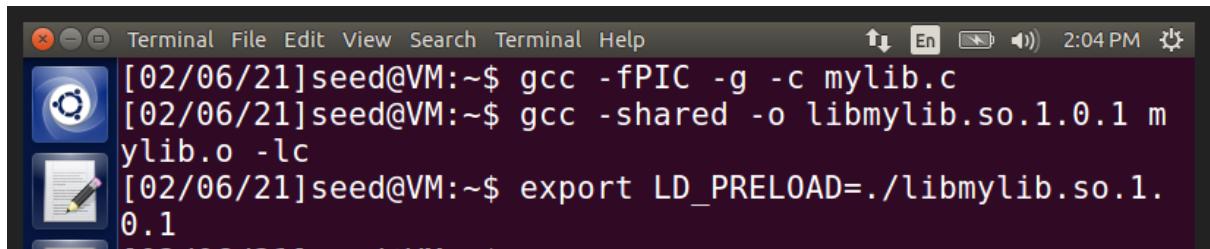
- For this task, the given program is stored as `mylib.c`



A screenshot of a Linux desktop environment, likely Ubuntu, showing a code editor window titled "mylib.c (~/)" - gedit. The code editor content is as follows:

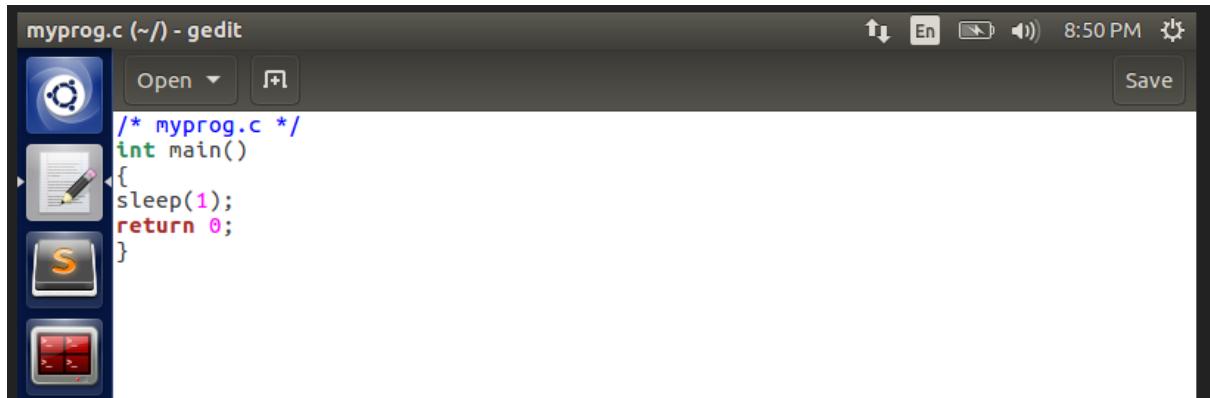
```
#include <stdio.h>
void sleep (int s)
{
/* If this is invoked by a privileged program,
you can do damages here! */
printf("I am not sleeping!\n");
}
```

- The above program is compiled using the following commands, as shown below. `LD_PRELOAD` variable is set after this.



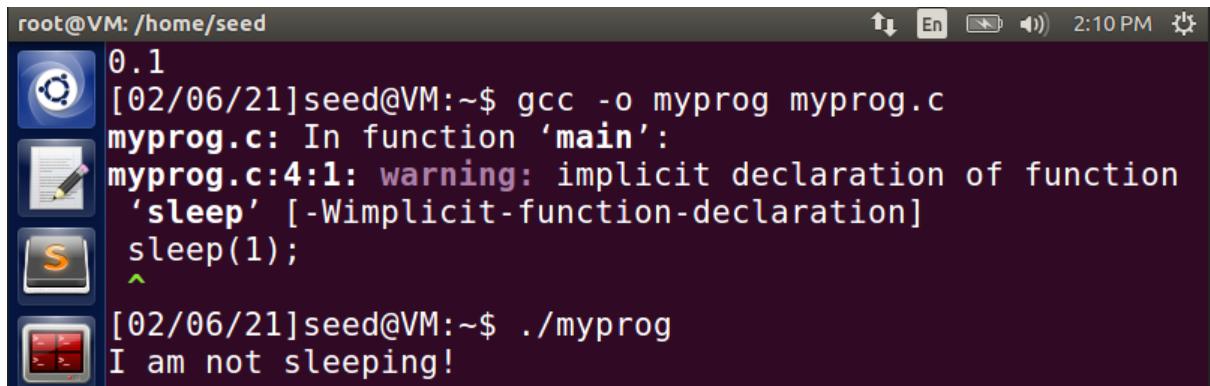
```
[02/06/21]seed@VM:~$ gcc -fPIC -g -c mylib.c
[02/06/21]seed@VM:~$ gcc -shared -o libmylib.so.1.0.1 mylib.o -lc
[02/06/21]seed@VM:~$ export LD_PRELOAD=./libmylib.so.1.0.1
```

- The given program is saved as myprog.c



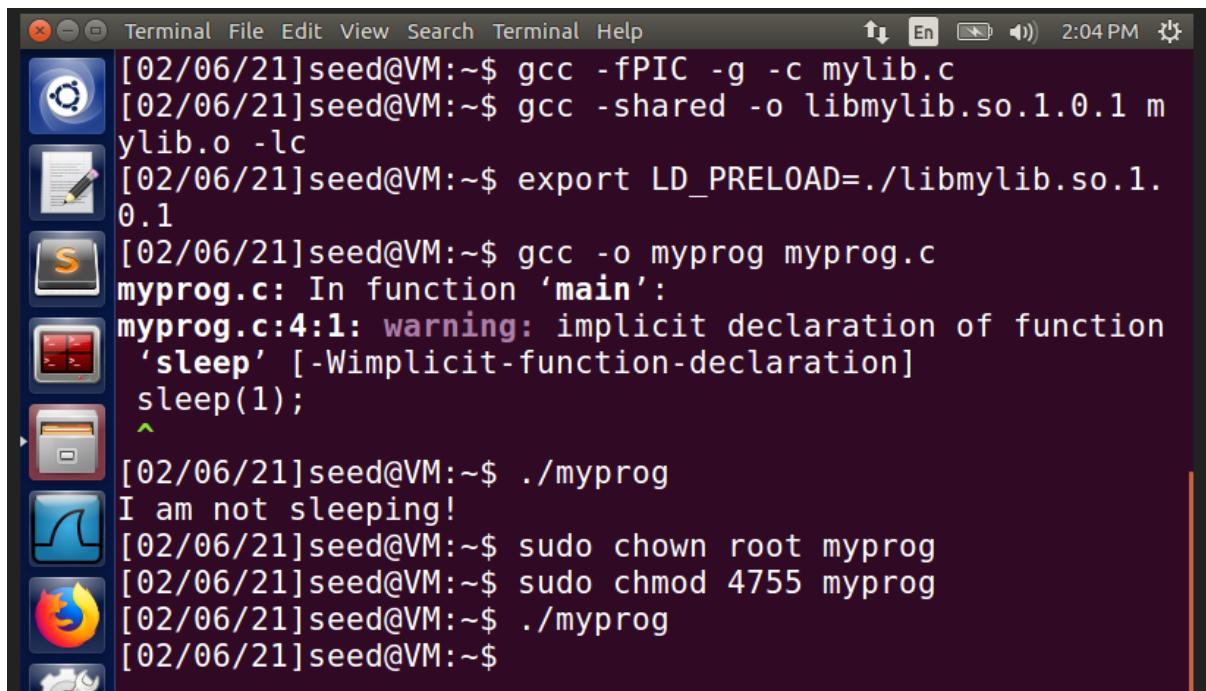
```
myprog.c (~/) - gedit
Open Save
/* myprog.c */
int main()
{
sleep(1);
return 0;
}
```

- The above program is run as a normal user and the output is seen below:



```
root@VM: /home/seed
0.1
[02/06/21]seed@VM:~$ gcc -o myprog myprog.c
myprog.c: In function 'main':
myprog.c:4:1: warning: implicit declaration of function
'sleep' [-Wimplicit-function-declaration]
sleep(1);
^
[02/06/21]seed@VM:~$ ./myprog
I am not sleeping!
```

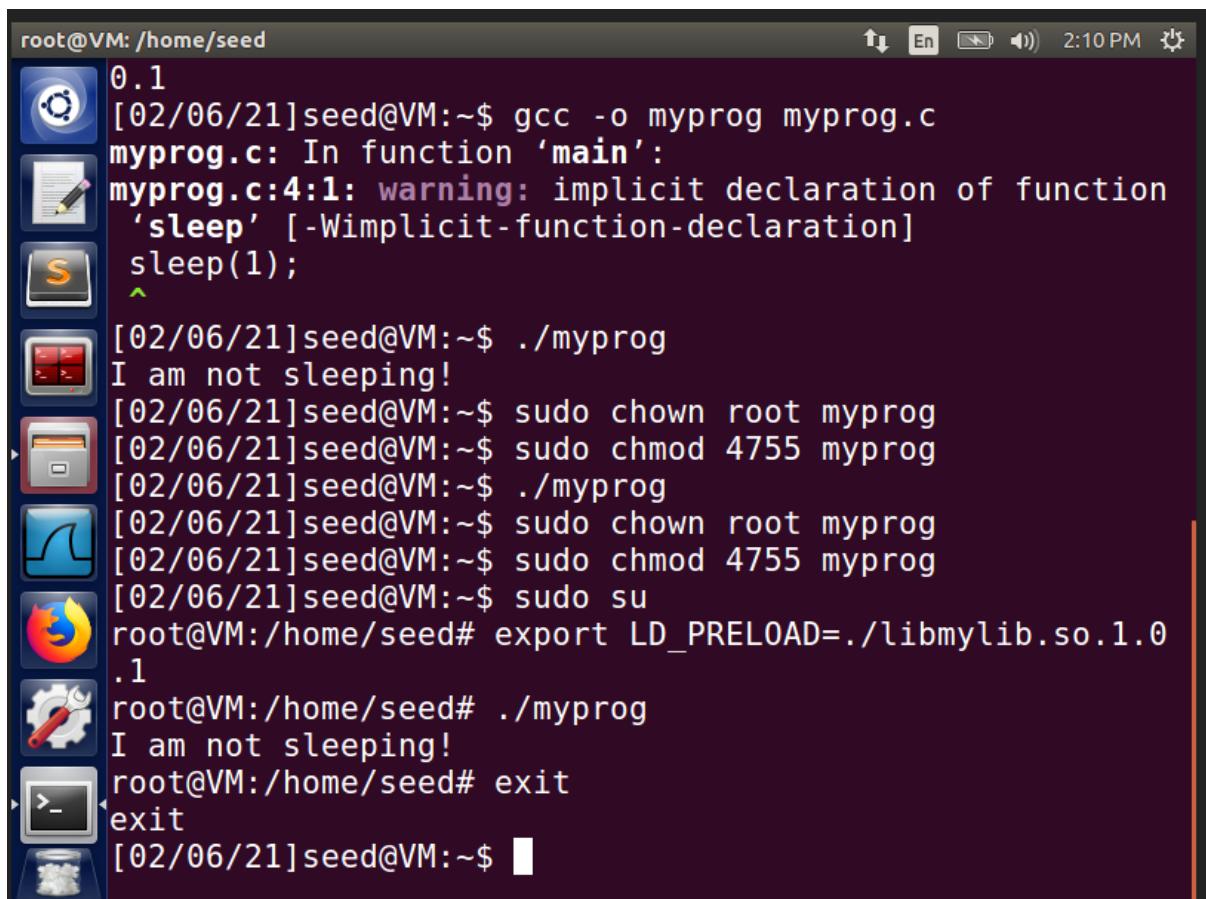
- Here, we can see that the myprog.c is run as a normal user and the printf present in mylib.c “I am not sleeping” is displayed. This shows that the sleep() function present in myprog.c is overridden by the function in mylib, which was earlier compiled and pre-loaded.
- Now, we give root permission to the myprog function and try running the program again. We can observe that on giving root permission, the program performs the root task (task defined in the myprog.c) and the program goes to sleep, which is what is defined in the program.



A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window. The terminal window has a dark background and contains the following command-line session:

```
[02/06/21]seed@VM:~$ gcc -fPIC -g -c mylib.c
[02/06/21]seed@VM:~$ gcc -shared -o libmylib.so.1.0.1 mylib.o -lc
[02/06/21]seed@VM:~$ export LD_PRELOAD=./libmylib.so.1.0.1
[02/06/21]seed@VM:~$ gcc -o myprog myprog.c
myprog.c: In function 'main':
myprog.c:4:1: warning: implicit declaration of function
'sleep' [-Wimplicit-function-declaration]
sleep(1);
^
[02/06/21]seed@VM:~$ ./myprog
I am not sleeping!
[02/06/21]seed@VM:~$ sudo chown root myprog
[02/06/21]seed@VM:~$ sudo chmod 4755 myprog
[02/06/21]seed@VM:~$ ./myprog
[02/06/21]seed@VM:~$
```

- Now, we again give root permission to myprog. This time, before trying to run the program, we export the LD_PRELOAD variable and then try running the program again.



A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window with root privileges. The terminal window has a dark background and contains the following command-line session:

```
root@VM: /home/seed
0.1
[02/06/21]seed@VM:~$ gcc -o myprog myprog.c
myprog.c: In function 'main':
myprog.c:4:1: warning: implicit declaration of function
'sleep' [-Wimplicit-function-declaration]
sleep(1);
^
[02/06/21]seed@VM:~$ ./myprog
I am not sleeping!
[02/06/21]seed@VM:~$ sudo chown root myprog
[02/06/21]seed@VM:~$ sudo chmod 4755 myprog
[02/06/21]seed@VM:~$ ./myprog
[02/06/21]seed@VM:~$ sudo chown root myprog
[02/06/21]seed@VM:~$ sudo chmod 4755 myprog
[02/06/21]seed@VM:~$ sudo su
root@VM:/home/seed# export LD_PRELOAD=./libmylib.so.1.0.1
root@VM:/home/seed# ./myprog
I am not sleeping!
root@VM:/home/seed# exit
exit
[02/06/21]seed@VM:~$
```

- In this case, the general sleep() function is overridden with the mylib.c printf() function, which is loaded through the libmylib.so.1.0.1

- Now, a new user **user1** is created and make myprog a Set-UID seed program as shown below:



```
user01@VM: /home/seed
[02/06/21]seed@VM:~$ sudo useradd -d /usr/user01 -m user01
[02/06/21]seed@VM:~$ sudo chown seed myprog
[02/06/21]seed@VM:~$ sudo chgrp seed myprog
[02/06/21]seed@VM:~$ sudo su user1
user1@VM:/home/seed$ exit
exit
```

- LD_PRELOAD variable is exported into that user account and myprog is run as below:



```
user01@VM: /home/seed
[02/06/21]seed@VM:~$ sudo useradd -d /usr/user01 -m user01
[02/06/21]seed@VM:~$ sudo chown seed myprog
[02/06/21]seed@VM:~$ sudo chgrp seed myprog
[02/06/21]seed@VM:~$ sudo su user1
user1@VM:/home/seed$ exit
exit
[02/06/21]seed@VM:~$ sudo su user01
user01@VM:/home/seed$ printenv LD_PRELOAD
user01@VM:/home/seed$ export user01 LD_PRELOAD=./libmylib.so.1.0.1
```

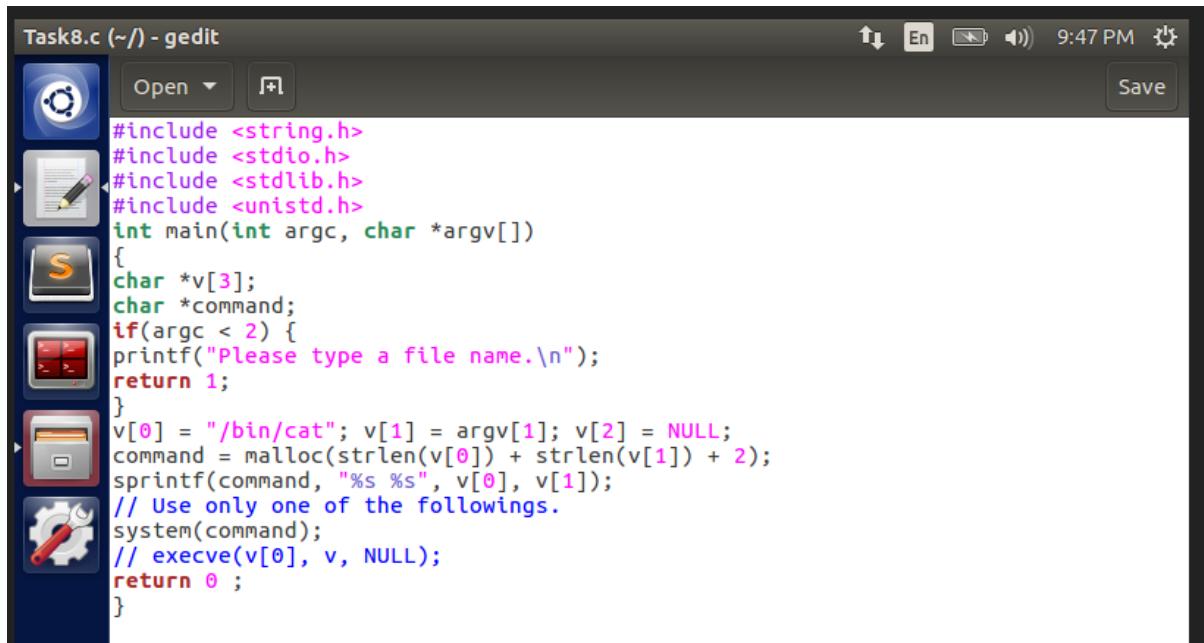


```
user01@VM: /home/seed
[02/06/21]seed@VM:~$ sudo useradd -d /usr/user01 -m user01
[02/06/21]seed@VM:~$ sudo chown seed myprog
[02/06/21]seed@VM:~$ sudo chgrp seed myprog
[02/06/21]seed@VM:~$ sudo su user1
user1@VM:/home/seed$ exit
exit
[02/06/21]seed@VM:~$ sudo su user01
user01@VM:/home/seed$ printenv LD_PRELOAD
user01@VM:/home/seed$ export user01 LD_PRELOAD=./libmylib.so.1.0.1
user01@VM:/home/seed$ exit
exit
[02/06/21]seed@VM:~$ printenv LD_PRELOAD
./libmylib.so.1.0.1
[02/06/21]seed@VM:~$ ./myprog
I am not sleeping!
[02/06/21]seed@VM:~$ sudo su user01
user01@VM:/home/seed$ ./myprog
user01@VM:/home/seed$
```

- It is observed from the above images that if the LD_PRELOAD variable is exported into the user01 account, then running the myprog will override the general sleep() function and execute the commands we have defined in mylib. But when we do not export the LD_PRELOAD variable, the general sleep() function is invoked.
- This is because, only when the user account and the LD_PRELOAD exporting user account are the same, only then our defined mylib function will work or else default functions will only work.

Task 8: Invoking External Programs Using system() versus execve()

- For this task, the given program is stored as Task8.c

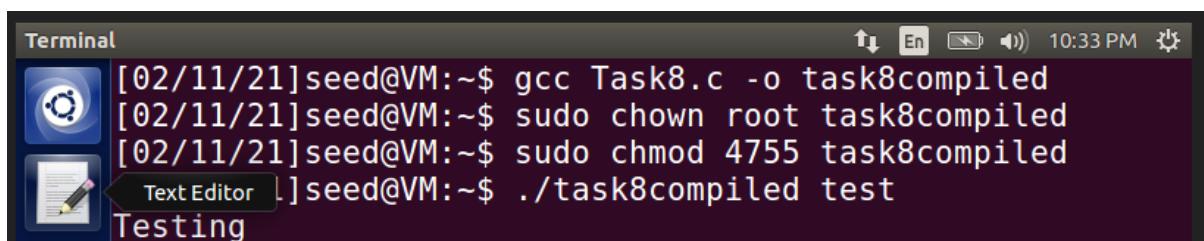


```

Task8.c (~/) - gedit
Open Save
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
int main(int argc, char *argv[])
{
    char *v[3];
    char *command;
    if(argc < 2) {
        printf("Please type a file name.\n");
        return 1;
    }
    v[0] = "/bin/cat"; v[1] = argv[1]; v[2] = NULL;
    command = malloc(strlen(v[0]) + strlen(v[1]) + 2);
    sprintf(command, "%s %s", v[0], v[1]);
    // Use only one of the followings.
    system(command);
    // execve(v[0], v, NULL);
    return 0 ;
}

```

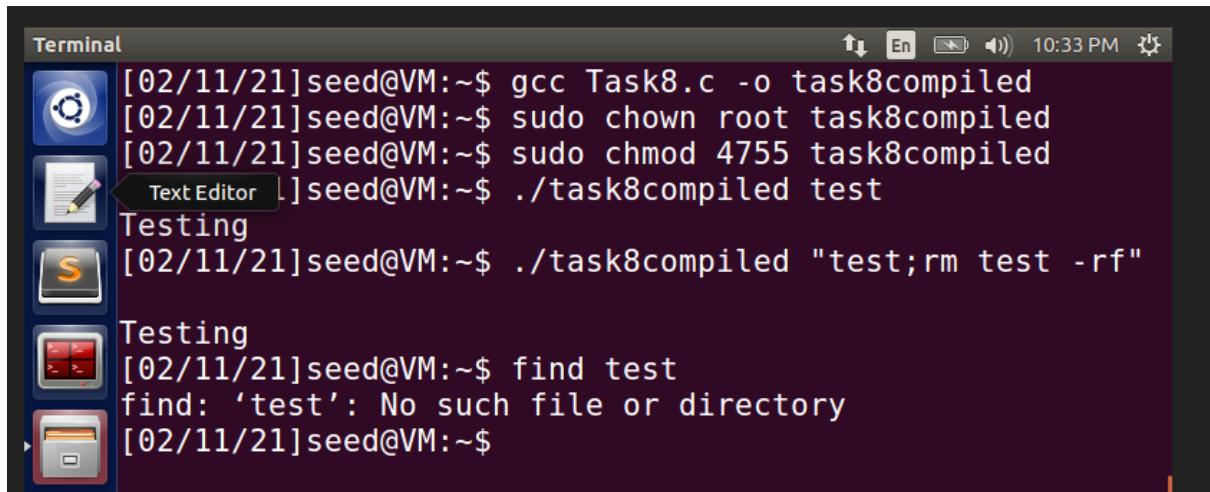
- The Task8.c is compiled and output is stored into task8compiled. Root access privileges are given to task8compiled using **chown** and **chmod** commands. A test document is created and run with the task8compiled. The test file is read and its contents are displayed.



```

Terminal
[02/11/21]seed@VM:~$ gcc Task8.c -o task8compiled
[02/11/21]seed@VM:~$ sudo chown root task8compiled
[02/11/21]seed@VM:~$ sudo chmod 4755 task8compiled
[02/11/21]seed@VM:~$ ./task8compiled test
Testing

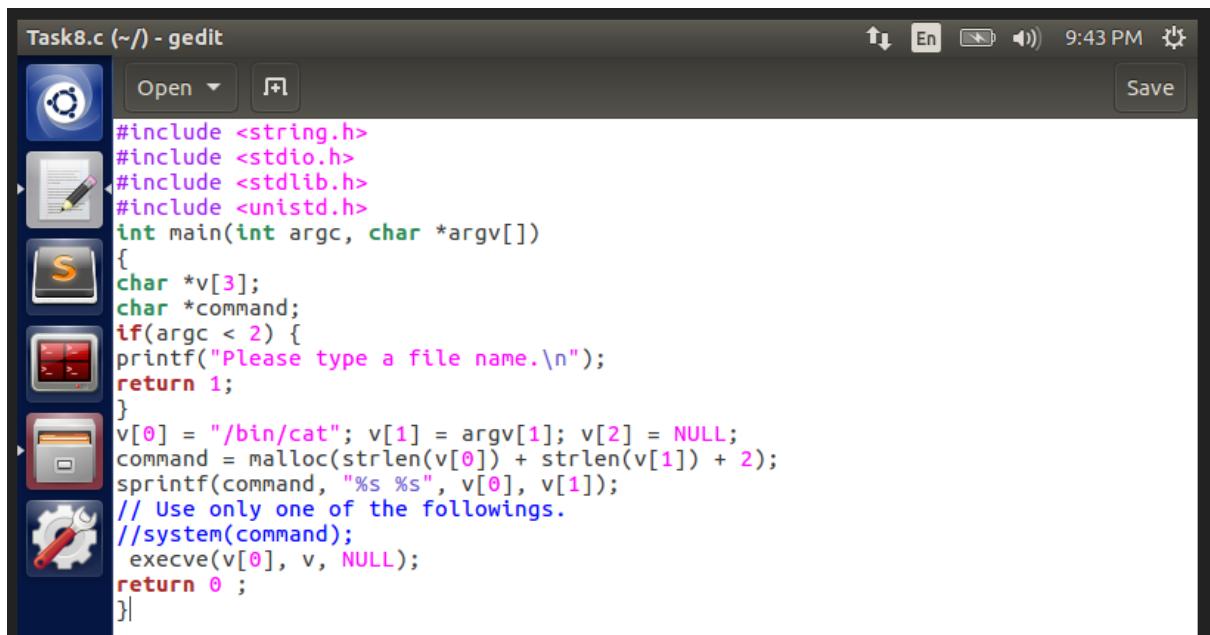
```



A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window titled "Terminal". The terminal window has a dark background and displays the following command-line session:

```
[02/11/21]seed@VM:~$ gcc Task8.c -o task8compiled
[02/11/21]seed@VM:~$ sudo chown root task8compiled
[02/11/21]seed@VM:~$ sudo chmod 4755 task8compiled
[02/11/21]seed@VM:~$ ./task8compiled test
Testing
[02/11/21]seed@VM:~$ ./task8compiled "test;rm test -rf"
Testing
[02/11/21]seed@VM:~$ find test
find: 'test': No such file or directory
[02/11/21]seed@VM:~$
```

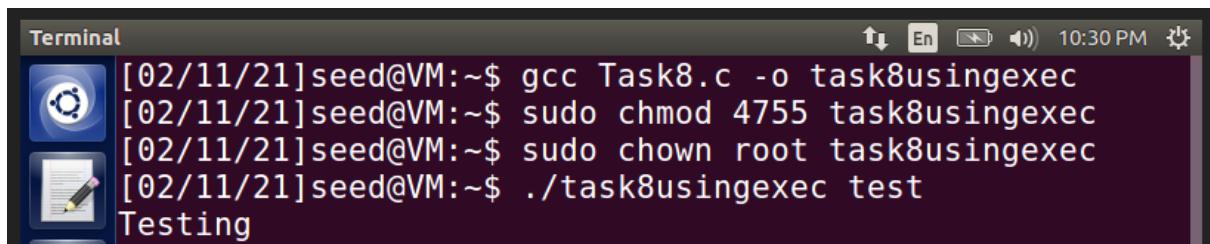
- Here, if we add ;rm test -rf after the test document, it deletes everything from the root directory, in this case, it deletes the test file. That is the reason why it says “No such file or directory” if we try to find the test file.
- Now, we comment out the system() command and uncomment the execv() function in Task8.c file.



A screenshot of a Linux desktop environment, likely Ubuntu, showing a text editor window titled "Task8.c (~/) - gedit". The text editor window has a dark background and displays the C code for Task8.c:

```
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
int main(int argc, char *argv[])
{
    char *v[3];
    char *command;
    if(argc < 2) {
        printf("Please type a file name.\n");
        return 1;
    }
    v[0] = "/bin/cat"; v[1] = argv[1]; v[2] = NULL;
    command = malloc(strlen(v[0]) + strlen(v[1]) + 2);
    sprintf(command, "%s %s", v[0], v[1]);
    // Use only one of the followings.
    //system(command);
    execve(v[0], v, NULL);
    return 0 ;
}
```

- The Task8.c is compiled again and output is stored into task8usingexec. Root access privileges are given to task8usingexec using **chown** and **chmod** commands. A test document is created and run with the task8usingexec. The test file is read and its contents are displayed.



A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window titled "Terminal". The terminal window has a dark background and displays the following command-line session:

```
[02/11/21]seed@VM:~$ gcc Task8.c -o task8usingexec
[02/11/21]seed@VM:~$ sudo chmod 4755 task8usingexec
[02/11/21]seed@VM:~$ sudo chown root task8usingexec
[02/11/21]seed@VM:~$ ./task8usingexec test
Testing
```

- Here, if we add ;rm test -rf after the test document, unlike system() command which deletes everything, in the case of execv() command, it does not delete anything.

```
[02/11/21]seed@VM:~$ gcc Task8.c -o task8usingexec
[02/11/21]seed@VM:~$ sudo chmod 4755 task8usingexec
[02/11/21]seed@VM:~$ sudo chown root task8usingexec
[02/11/21]seed@VM:~$ ./task8usingexec test
Testing
[02/11/21]seed@VM:~$ ./task8usingexec "test;rm test -rf"
/bin/cat: 'test;rm test -rf': No such file or directory
[02/11/21]seed@VM:~$
```

- Rather, it treats the whole of “**test;rm test -rf**” as a single string and tries to search for that string in the directory which will result in the above output. This is because, **execv()** function does not have the root access privileges unlike **system()** function.

Task 9: Capability Leaking

- For this task, the given program is stored as Task9.c

```
#include <stdlib.h>
#include <fcntl.h>
void main()
{
    int fd;
    /* Assume that /etc/zzz is an important system file,
     * and it is owned by root with permission 0644.
     * Before running this program, you should creat
     * the file /etc/zzz first. */
    fd = open("/etc/zzz", O_RDWR | O_APPEND);
    if (fd == -1) {
        printf("Cannot open /etc/zzz\n");
        exit(0);
    }
    /* Simulate the tasks conducted by the program */
    sleep(1);
    /* After the task, the root privileges are no longer needed,
     * it's time to relinquish the root privileges permanently. */
    setuid(getuid()); /* getuid() returns the real uid */
    if (fork()) { /* In the parent process */
        close(fd);
        exit(0);
    } else { /* in the child process */
        /* Now, assume that the child process is compromised, malicious
         * attackers have injected the following statements
         * into this process */
        write(fd, "Malicious Data\n", 15);
        close(fd);
    }
}
```

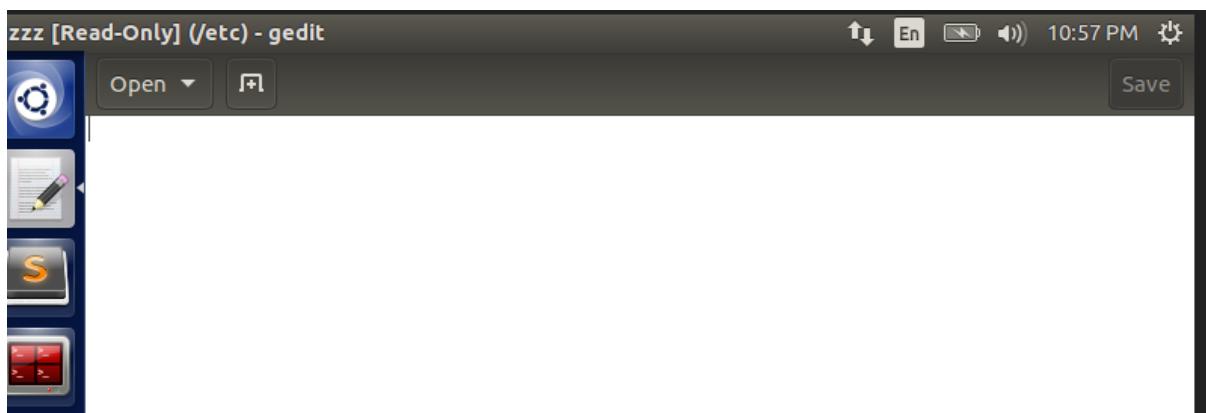
- Root privileges are given to the file task9op and we find that the file zzz is empty at first.

```
root@VM: /etc
'fork' [-Wimplicit-function-declaration]
Search your computer /* In the parent process */

Task9.c:22:1: warning: implicit declaration of function
  'close' [-Wimplicit-function-declaration]
    close (fd);

Task9.c:28:1: warning: implicit declaration of function
  'write' [-Wimplicit-function-declaration]
    write (fd, "Malicious Data\n", 15);

[02/06/21]seed@VM:~$ sudo chown root task9op
[02/06/21]seed@VM:~$ sudo chmod 4755 task9op
[02/06/21]seed@VM:~$ su root
Password:
root@VM:/home/seed# cd /etc
root@VM:/etc# touch zzz
root@VM:/etc# chown root zzz
root@VM:/etc# chmod 644 zzz
root@VM:/etc# ls -lah zzz
-rw-r--r-- 1 root root 0 Feb  6 16:20 zzz
root@VM:/etc#
```



- Now, we run the same program as a normal user with the command `./task9op` and we find that “Malicious Data” is written into the zzz file as seen below.

Text Editor

```
Task9.c:21:5: warning: implicit declaration of function 'fork' [-Wimplicit-function-declaration]
if (fork()) { /* In the parent process */
^
Task9.c:22:1: warning: implicit declaration of function 'close' [-Wimplicit-function-declaration]
close (fd);
^
Task9.c:28:1: warning: implicit declaration of function 'write' [-Wimplicit-function-declaration]
write (fd, "Malicious Data\n", 15);
^
[02/06/21]seed@VM:~$ sudo chown root task9op
[02/06/21]seed@VM:~$ sudo chmod 4755 task9op
[02/06/21]seed@VM:~$ su root
Password:
root@VM:/home/seed# cd /etc
root@VM:/etc# touch zzz
root@VM:/etc# chown root zzz
root@VM:/etc# chmod 644 zzz
root@VM:/etc# ls -lah zzz
-rw-r--r-- 1 root root 0 Feb  6 16:44 zzz
root@VM:/etc# exit
exit
[02/06/21]seed@VM:~$ ./task9op
[02/06/21]seed@VM:~$
```

zzz [Read-Only] (/etc) - gedit

Open ▾

Malicious Data

This is because, when the user has root access, it does not allow them access the malicious file and edit its contents. But when the privileges are downgraded, we are able to launch the attack and write into the file.