Operating Systems (CSL 3030)

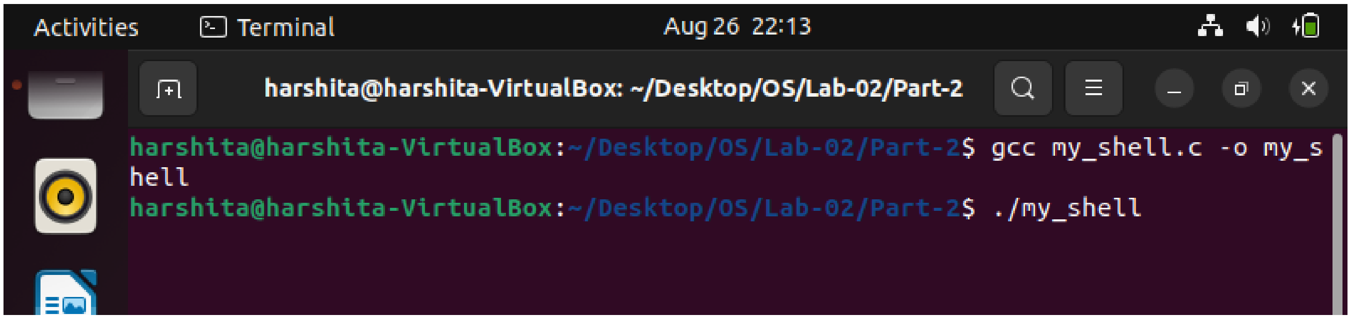
Readme file

Lab Assignment	01
Roll No.	B20CS018
Name	Harshita Gupta

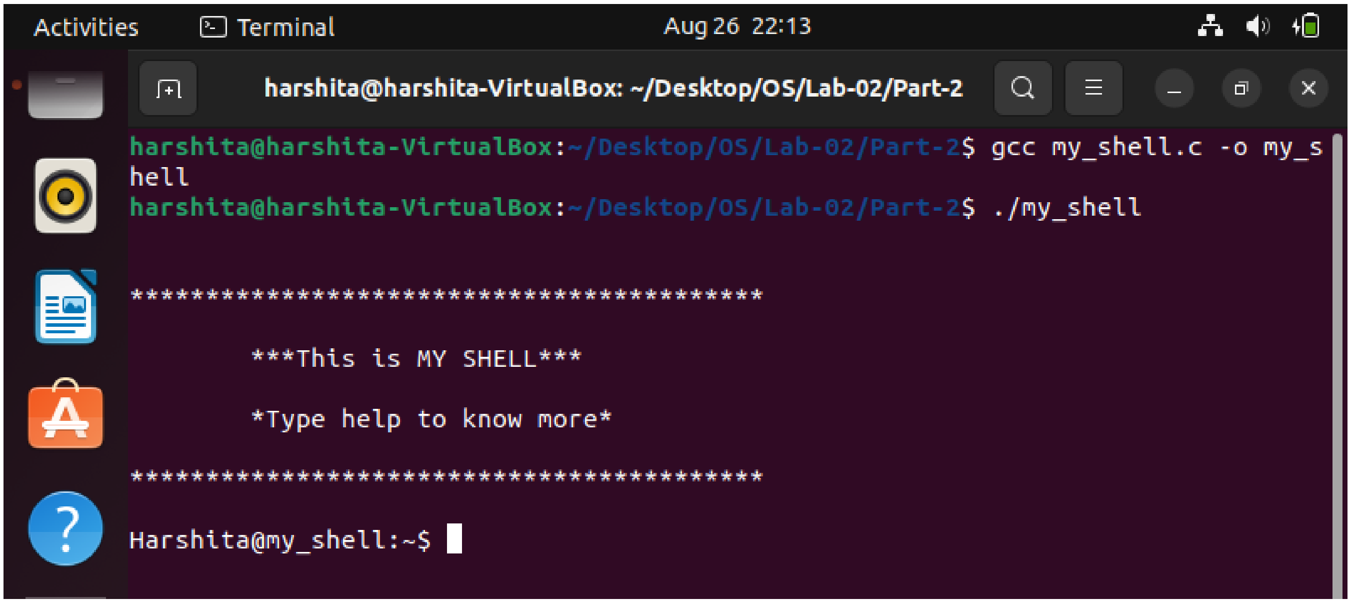
PART II:

Run the C program:

1. Run the C file named “my_shell.c” in the submitted zip file by following the commands as shown below.
2. The shell display a prompt and you will be asked to enter the command.



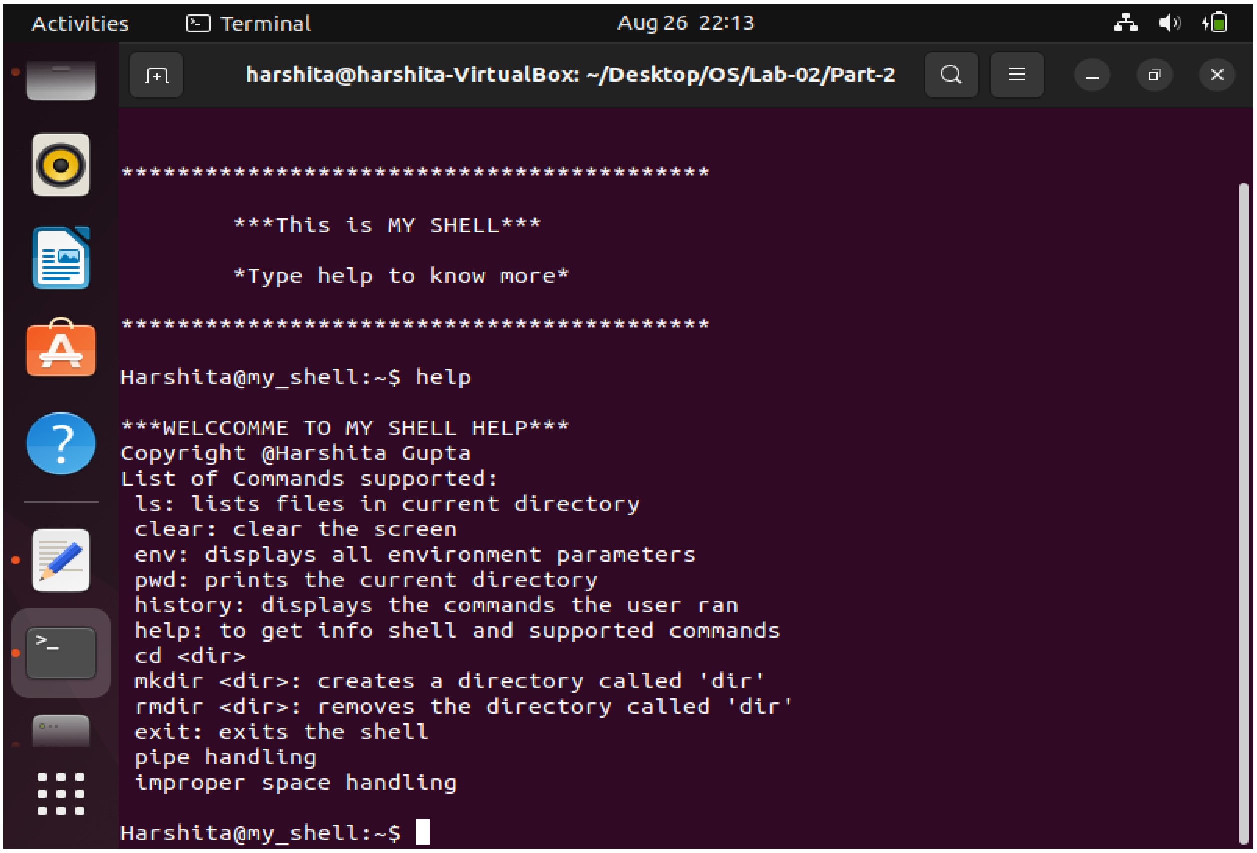
```
harshita@harshita-VirtualBox:~/Desktop/OS/Lab-02/Part-2$ gcc my_shell.c -o my_s
hell
harshita@harshita-VirtualBox:~/Desktop/OS/Lab-02/Part-2$ ./my_shell
```



```
harshita@harshita-VirtualBox:~/Desktop/OS/Lab-02/Part-2$ gcc my_shell.c -o my_s
hell
harshita@harshita-VirtualBox:~/Desktop/OS/Lab-02/Part-2$ ./my_shell


******************************************

        ***This is MY SHELL***

        *Type help to know more*

******************************************

Harshita@my_shell:~$ 
```

Help command:

1. After running the program in the Console it displays "Type help to know more".
2. Help shows the list of the commands the shell supports and its info.



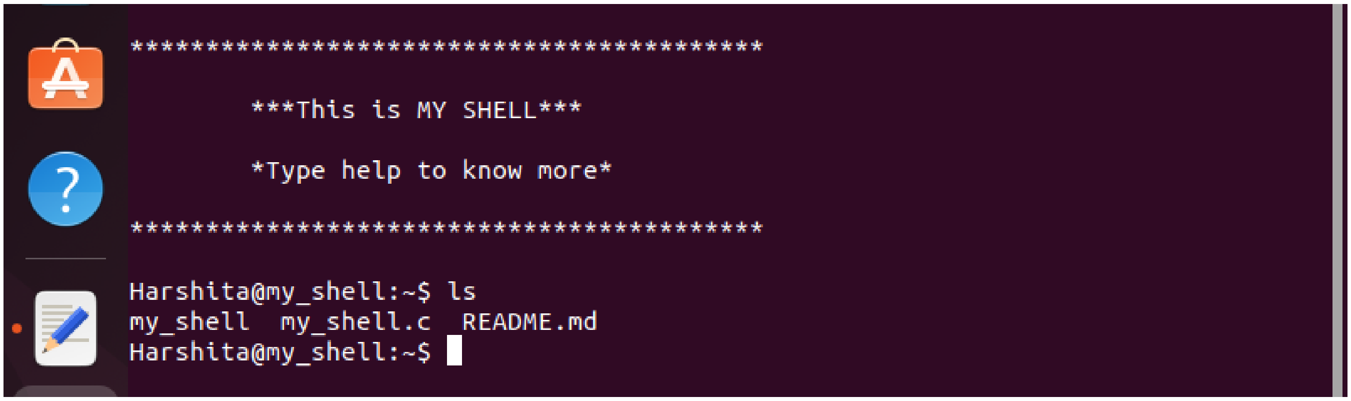
```
******************************************

        ***This is MY SHELL***

        *Type help to know more*

******************************************

Harshita@my_shell:~$ help

***WELCCOMME TO MY SHELL HELP***
Copyright @Harshita Gupta
List of Commands supported:
 ls: lists files in current directory
 clear: clear the screen
 env: displays all environment parameters
 pwd: prints the current directory
 history: displays the commands the user ran
 help: to get info shell and supported commands
 cd <dir>
 mkdir <dir>: creates a directory called 'dir'
 rmdir <dir>: removes the directory called 'dir'
 exit: exits the shell
 pipe handling
 improper space handling

Harshita@my_shell:~$ 
```

(NOTE: there are 2 types of commands internal and external,

Internal commands are - clear, env, pwd, history, help, cd, mkdir, rmdir, and exit

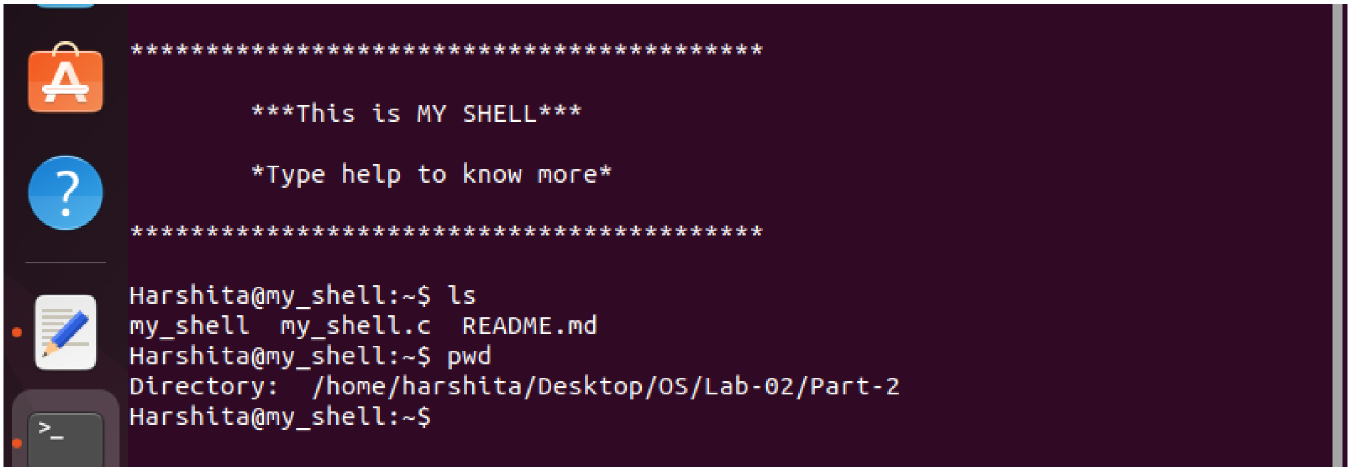
External commands are - ls and cat)

ls command: list all files and directories in the given directory.



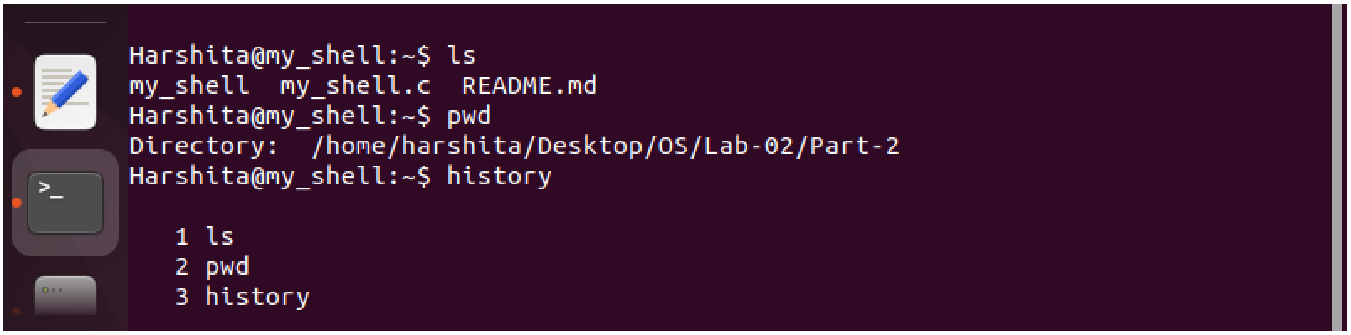
```
******************************************

        ***This is MY SHELL***

        *Type help to know more*

******************************************

Harshita@my_shell:~$ ls
my_shell  my_shell.c  README.md
Harshita@my_shell:~$ 
```

pwd command: print the current directory.



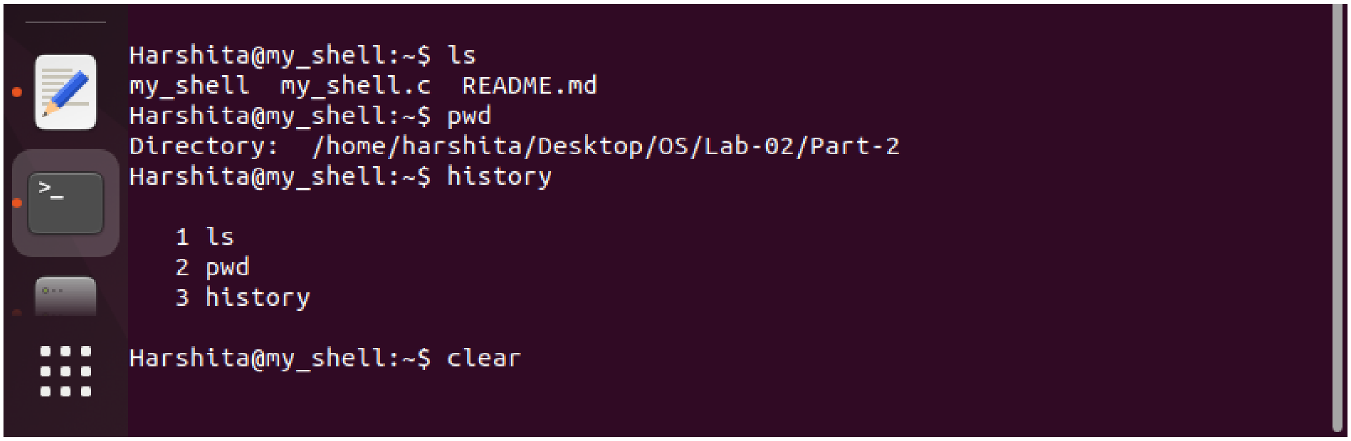
```
****************************************

       ***This is MY SHELL***

       *Type help to know more*

****************************************

Harshita@my_shell:~$ ls
my_shell  my_shell.c  README.md
Harshita@my_shell:~$ pwd
Directory:  /home/harshita/Desktop/OS/Lab-02/Part-2
Harshita@my_shell:~$
```

history command: display the commands the user ran.

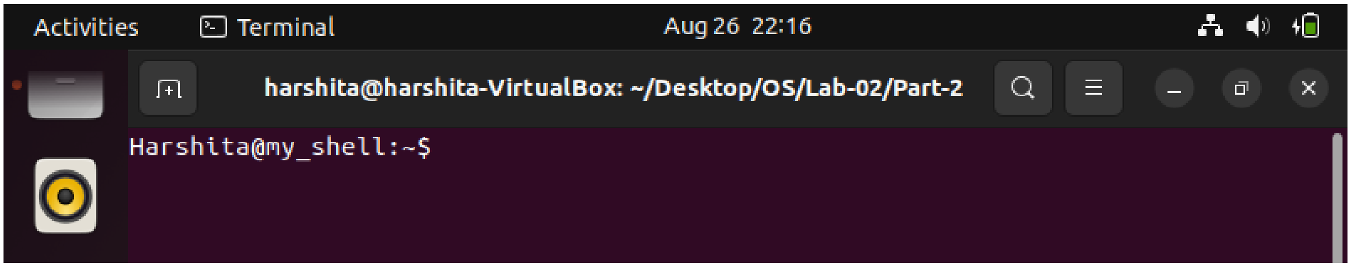


```
Harshita@my_shell:~$ ls
my_shell  my_shell.c  README.md
Harshita@my_shell:~$ pwd
Directory:  /home/harshita/Desktop/OS/Lab-02/Part-2
Harshita@my_shell:~$ history

   1 ls
   2 pwd
   3 history
```

Clear command: clear the console.

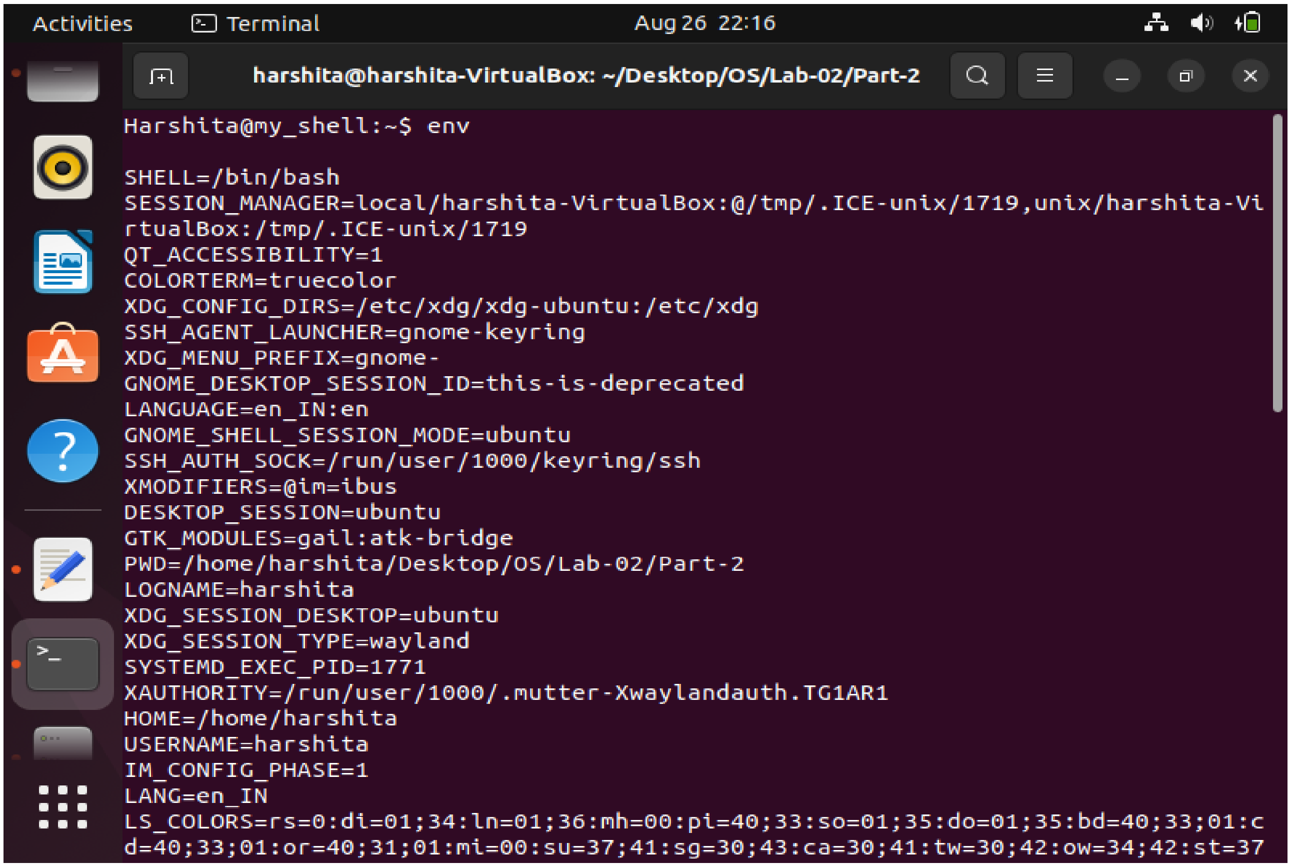


```
Harshita@my_shell:~$ ls
my_shell  my_shell.c  README.md
Harshita@my_shell:~$ pwd
Directory:  /home/harshita/Desktop/OS/Lab-02/Part-2
Harshita@my_shell:~$ history

   1 ls
   2 pwd
   3 history

Harshita@my_shell:~$ clear
```

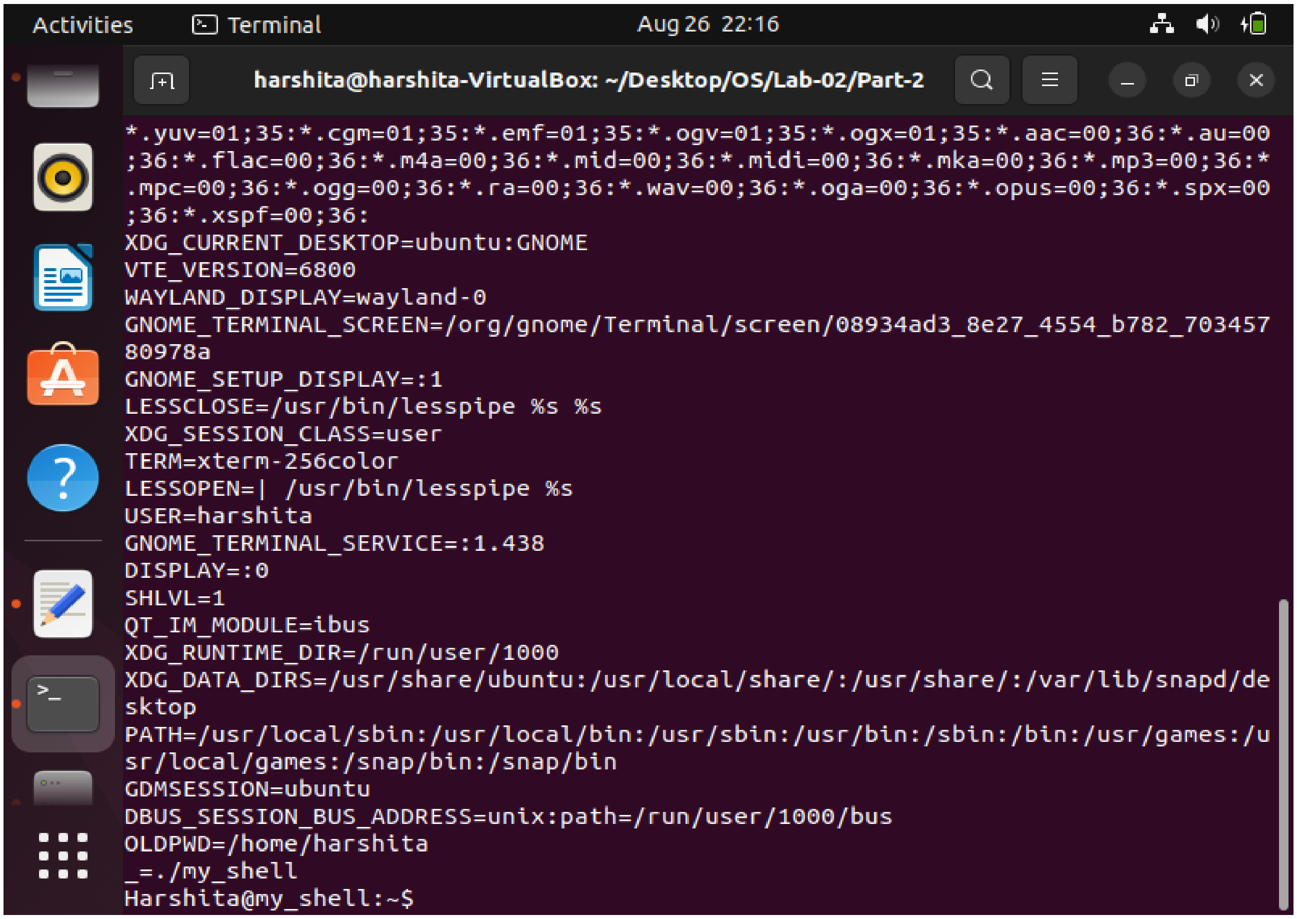


```
Harshita@my_shell:~$
```

env command: display all environment variables.

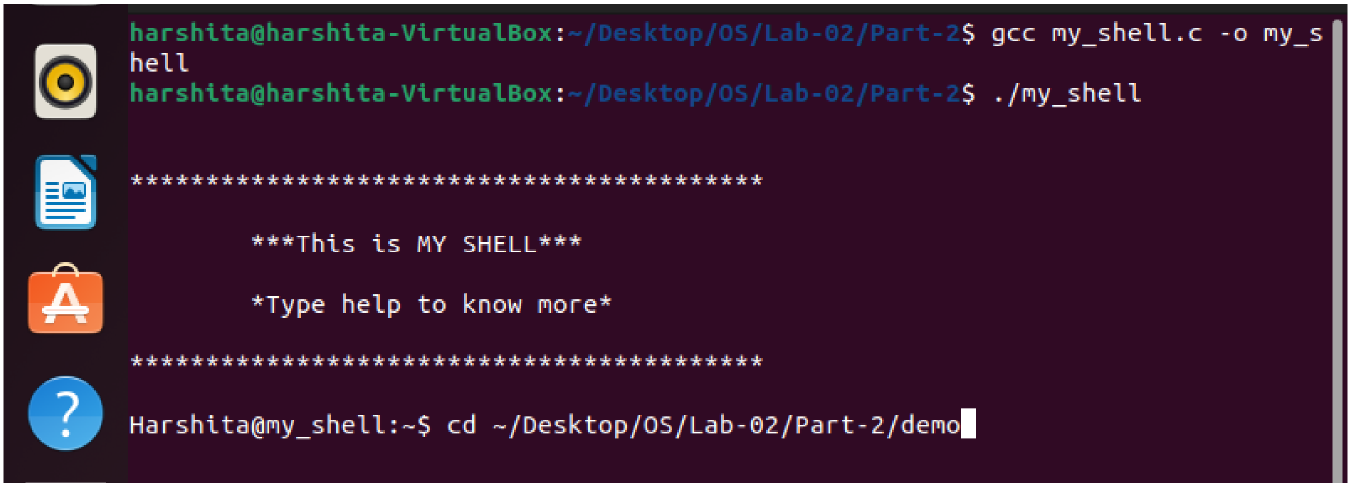


```
Harshita@my_shell:~$ env

SHELL=/bin/bash
SESSION_MANAGER=local/harshita-VirtualBox:@/tmp/.ICE-unix/1719,unix/harshita-Vi
rtualBox:/tmp/.ICE-unix/1719
QT_ACCESSIBILITY=1
COLORTERM=truecolor
XDG_CONFIG_DIRS=/etc/xdg/xdg-ubuntu:/etc/xdg
SSH_AGENT_LAUNCHER=gnome-keyring
XDG_MENU_PREFIX=gnome-
GNOME_DESKTOP_SESSION_ID=this-is-deprecated
LANGUAGE=en_IN:en
GNOME_SHELL_SESSION_MODE=ubuntu
SSH_AUTH_SOCK=/run/user/1000/keyring/ssh
XMODIFIERS=@im=ibus
DESKTOP_SESSION=ubuntu
GTK_MODULES=gail:atk-bridge
PWD=/home/harshita/Desktop/OS/Lab-02/Part-2
LOGNAME=harshita
XDG_SESSION_DESKTOP=ubuntu
XDG_SESSION_TYPE=wayland
SYSTEMD_EXEC_PID=1771
XAUTHORITY=/run/user/1000/.mutter-Xwaylandauth.TG1AR1
HOME=/home/harshita
USERNAME=harshita
IM_CONFIG_PHASE=1
LANG=en_IN
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:c
d=40;33;01:or=40;31;01:mi=00:su=37;41:sg=30;43:ca=30;41:tw=30;42:ow=34;42:st=37
```



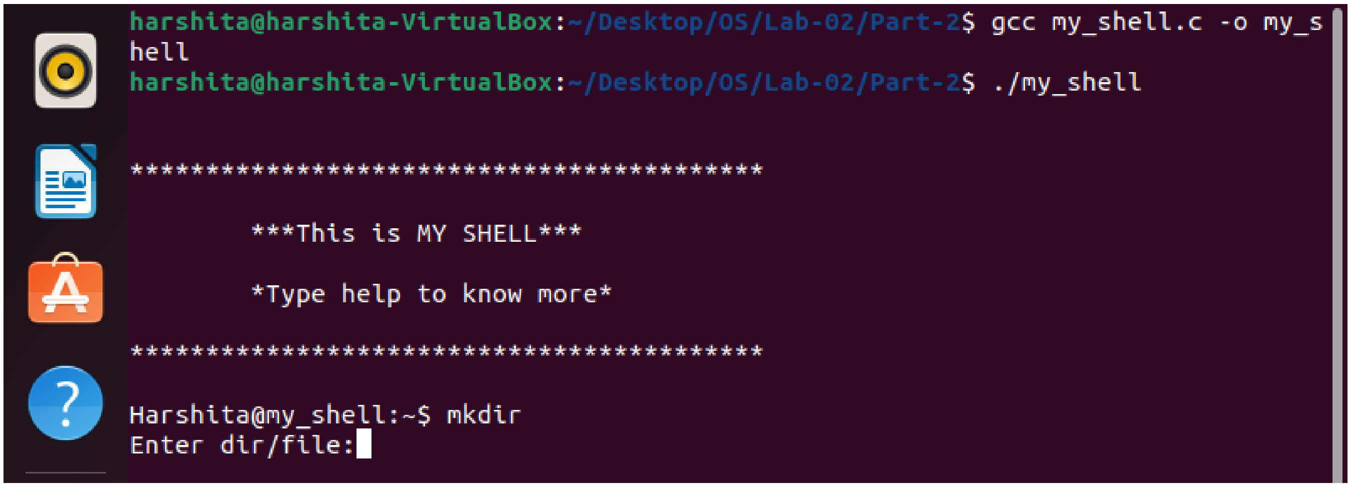
```
*.yuv=01;35:*.cgm=01;35:*.emf=01;35:*.ogv=01;35:*.ogx=01;35:*.aac=00;36:*.au=00
;36:*.flac=00;36:*.m4a=00;36:*.mid=00;36:*.midi=00;36:*.mka=00;36:*.mp3=00;36:*
.mpc=00;36:*.ogg=00;36:*.ra=00;36:*.wav=00;36:*.oga=00;36:*.opus=00;36:*.spx=00
;36:*.xspf=00;36:
XDG_CURRENT_DESKTOP=ubuntu:GNOME
VTE_VERSION=6800
WAYLAND_DISPLAY=wayland-0
GNOME_TERMINAL_SCREEN=/org/gnome/Terminal/screen/08934ad3_8e27_4554_b782_703457
80978a
GNOME_SETUP_DISPLAY=:1
LESSCLOSE=/usr/bin/lesspipe %s %s
XDG_SESSION_CLASS=user
TERM=xterm-256color
LESSOPEN=| /usr/bin/lesspipe %s
USER=harshita
GNOME_TERMINAL_SERVICE=:1.438
DISPLAY=:0
SHLVL=1
QT_IM_MODULE=ibus
XDG_RUNTIME_DIR=/run/user/1000
XDG_DATA_DIRS=/usr/share/ubuntu:/usr/local/share/:/usr/share/:/var/lib/snapd/de
sktop
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/u
sr/local/games:/snap/bin:/snap/bin
GDMSESSION=ubuntu
DBUS_SESSION_BUS_ADDRESS=unix:path=/run/user/1000/bus
OLDPWD=/home/harshita
_=./my_shell
Harshita@my_shell:~$
```

cd command: open up a directory.



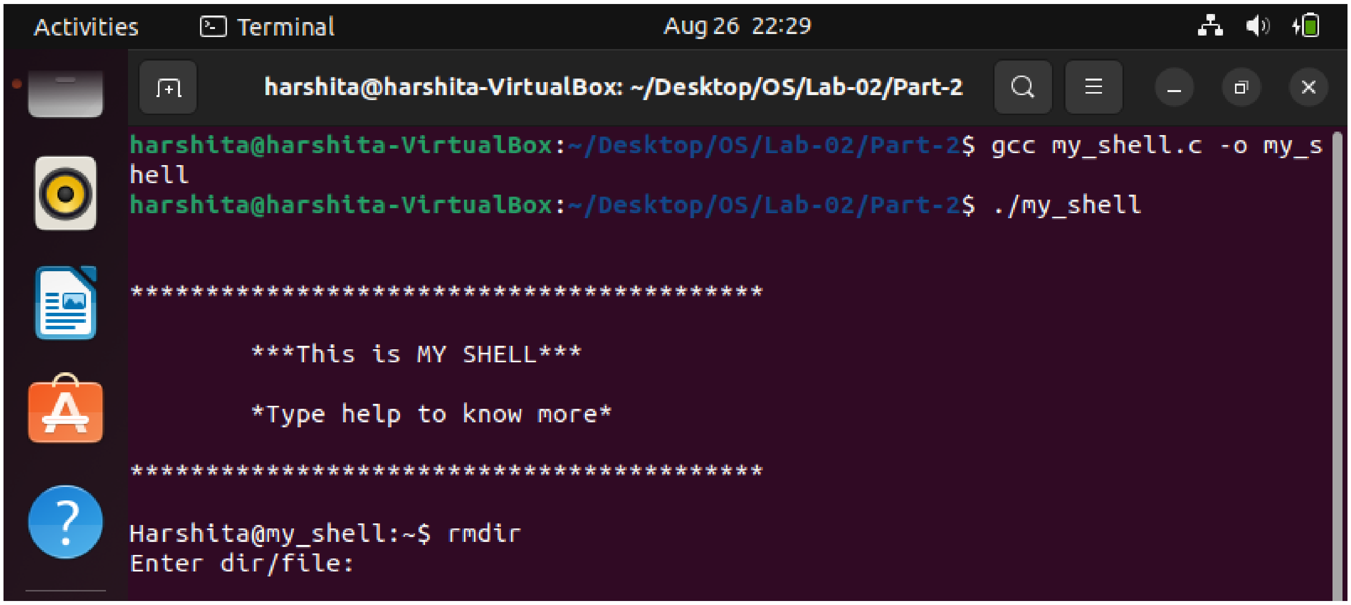
```
harshita@harshita-VirtualBox:~/Desktop/OS/Lab-02/Part-2$ gcc my_shell.c -o my_s
hell
harshita@harshita-VirtualBox:~/Desktop/OS/Lab-02/Part-2$ ./my_shell


******************************************

        ***This is MY SHELL***

        *Type help to know more*

******************************************

Harshita@my_shell:~$ cd ~/Desktop/OS/Lab-02/Part-2/demo
```

mkdir command: creates a directory.



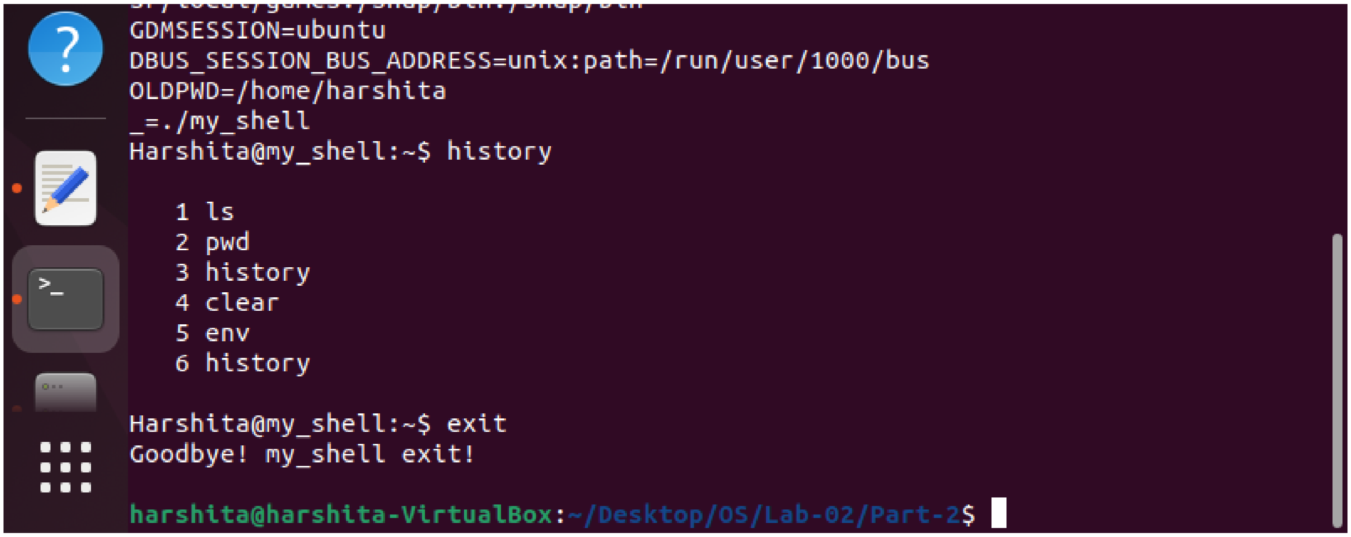
```
harshita@harshita-VirtualBox:~/Desktop/OS/Lab-02/Part-2$ gcc my_shell.c -o my_s
hell
harshita@harshita-VirtualBox:~/Desktop/OS/Lab-02/Part-2$ ./my_shell


******************************************

        ***This is MY SHELL***

        *Type help to know more*

******************************************

Harshita@my_shell:~$ mkdir
Enter dir/file:
```

rmdir command: removes a directory.



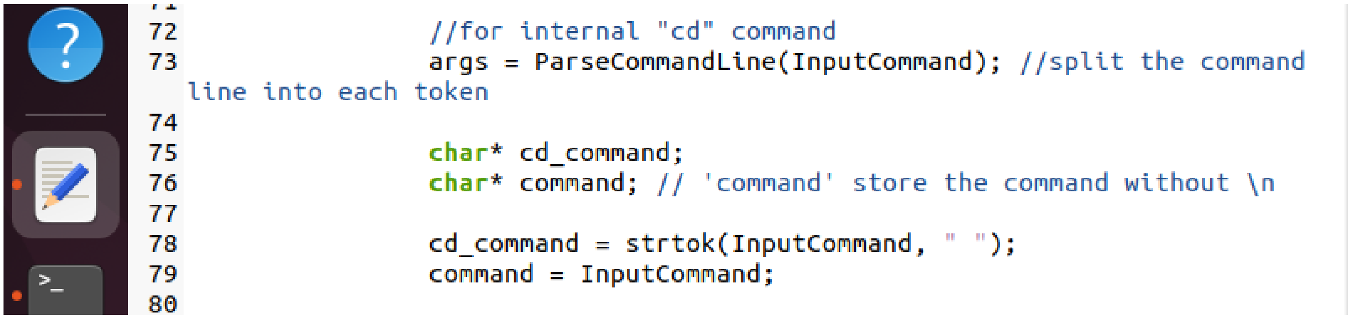
```
harshita@harshita-VirtualBox:~/Desktop/OS/Lab-02/Part-2$ gcc my_shell.c -o my_s
hell
harshita@harshita-VirtualBox:~/Desktop/OS/Lab-02/Part-2$ ./my_shell


******************************************

        ***This is MY SHELL***

        *Type help to know more*

******************************************

Harshita@my_shell:~$ rmdir
Enter dir/file:
```

exit command: It is used to exit the program shell.



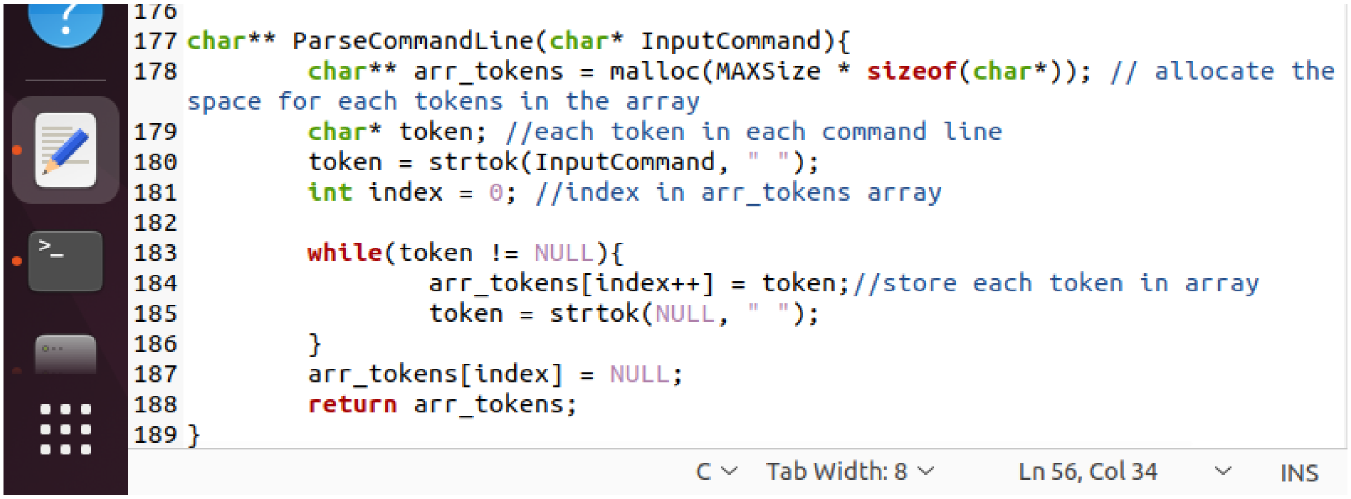
```
GDMSESSION=ubuntu
DBUS_SESSION_BUS_ADDRESS=unix:path=/run/user/1000/bus
OLDPWD=/home/harshita
_=./my_shell
Harshita@my_shell:~$ history

   1 ls
   2 pwd
   3 history
   4 clear
   5 env
   6 history

Harshita@my_shell:~$ exit
Goodbye! my_shell exit!

harshita@harshita-VirtualBox:~/Desktop/OS/Lab-02/Part-2$ 
```

Improper space handling:



```
			//for internal "cd" command
			args = ParseCommandLine(InputCommand); //split the command line into each token

			char* cd_command;
			char* command; // 'command' store the command without \n

			cd_command = strtok(InputCommand, " ");
			command = InputCommand;

```

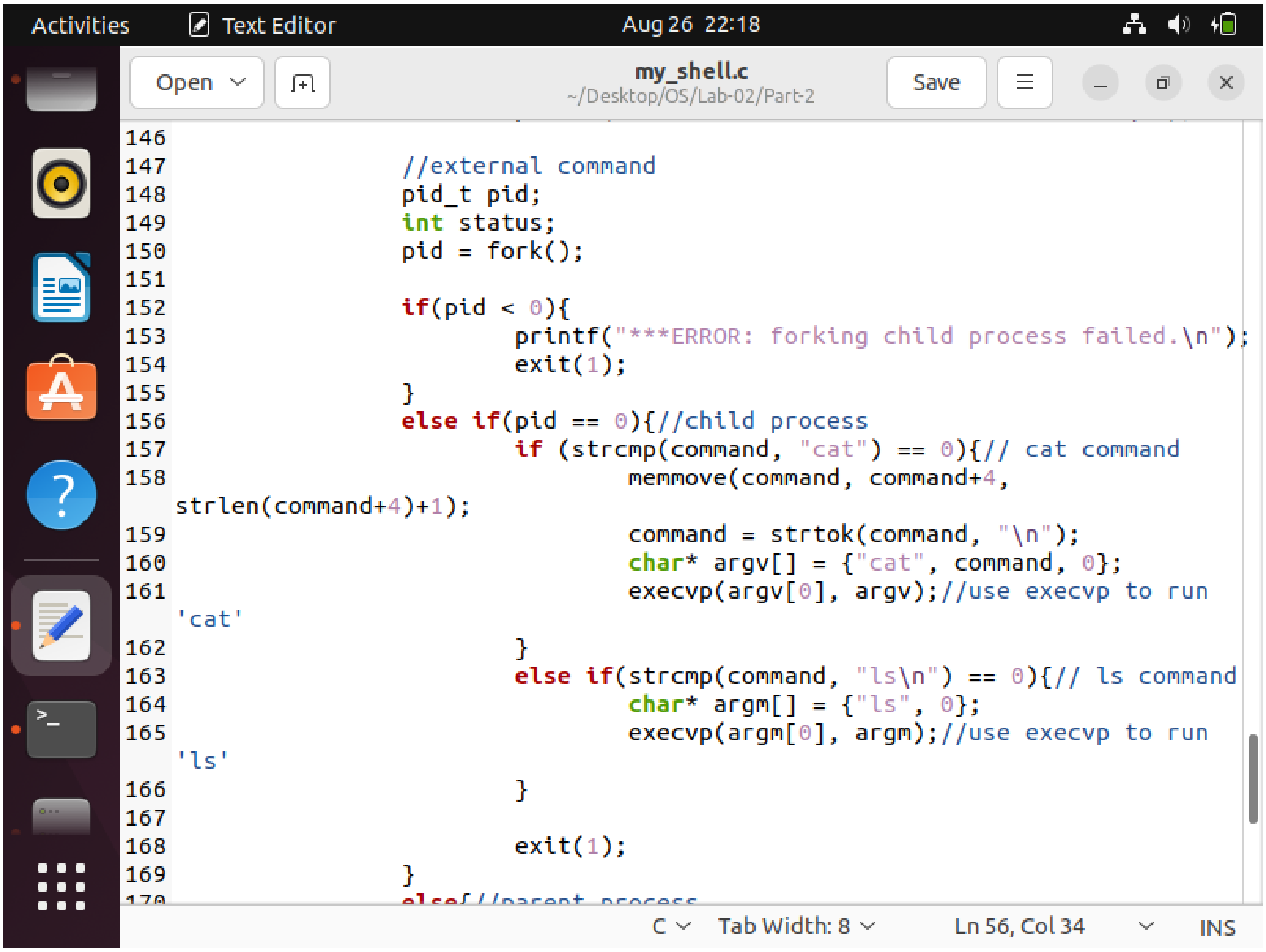
Parse the line into the program name and an array of parameters:



```

char** ParseCommandLine(char* InputCommand){
	char** arr_tokens = malloc(MAXSize * sizeof(char*)); // allocate the space for each tokens in the array
	char* token; //each token in each command line
	token = strtok(InputCommand, " ");
	int index = 0; //index in arr_tokens array

	while(token != NULL){
		arr_tokens[index++] = token;//store each token in array
		token = strtok(NULL, " ");
	}
	arr_tokens[index] = NULL;
	return arr_tokens;
}
```

C ∨ Tab Width: 8 ∨ Ln 56, Col 34 ∨ INS

Use the fork() system call to spawn a new child process and once the child finishes, the shell repeats the loop by jumping to 1::



```

                //external command
                pid_t pid;
                int status;
                pid = fork();

                if(pid < 0){
                        printf("***ERROR: forking child process failed.\n");
                        exit(1);
                }
                else if(pid == 0){//child process
                        if (strcmp(command, "cat") == 0){// cat command
                                memmove(command, command+4, strlen(command+4)+1);
                                command = strtok(command, "\n");
                                char* argv[] = {"cat", command, 0};
                                execvp(argv[0], argv);//use execvp to run 'cat'
                        }
                        else if(strcmp(command, "ls\n") == 0){// ls command
                                char* argm[] = {"ls", 0};
                                execvp(argm[0], argm);//use execvp to run 'ls'
                        }

                        exit(1);
                }
                else{//parent process
```