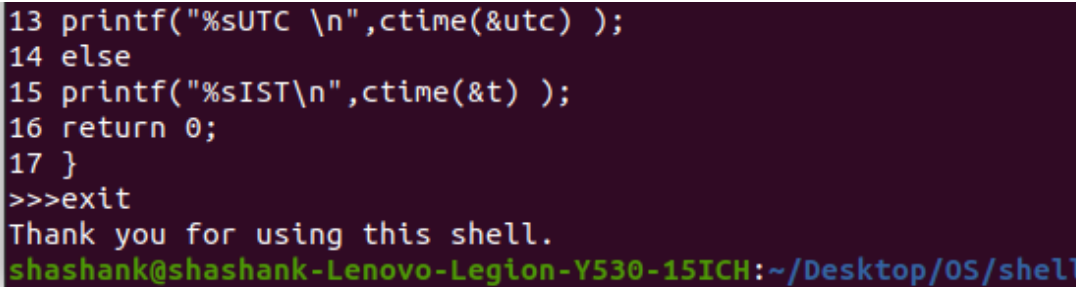
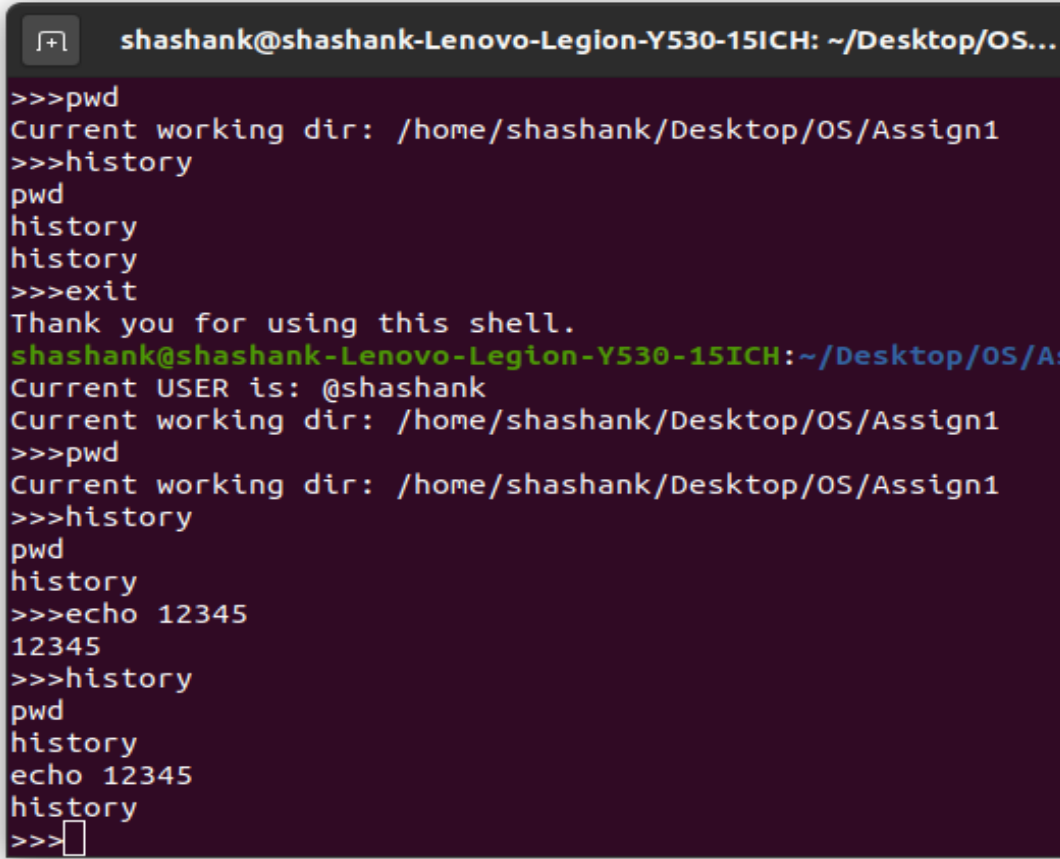


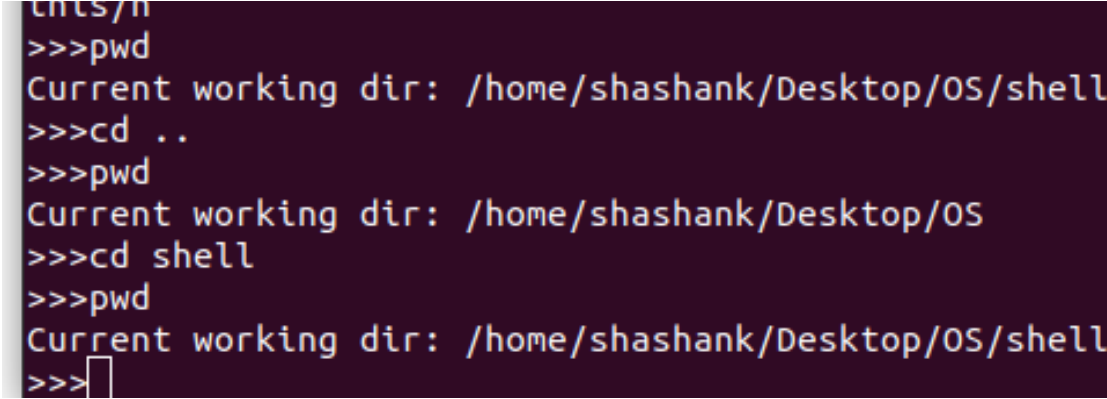
Task 2 Description Write Up

Command s	Options Implement ed	Syntax	C Functions used	Error handling
pwd	-P (default)	>>>pwd >>>pwd -P	It is implemented using getcwd() in C.	<p>If getcwd function somehow encounters error and returns a NULL value it is caught as an exception and is handled using an error message to the use</p> <pre>>>>^C shashank@shashank-Lenovo-Legion-Y530-15ICH:~/Desktop/OS/shell Current USER is: @shashank Current working dir: /home/shashank/Desktop/OS/shell >>>echo hello world hello world >>>echo -n Hello world Hello world>>>echo -E Hello Hello >>>echo -E this/n this/n >>>pwd Current working dir: /home/shashank/Desktop/OS/shell >>>cd .. >>>pwd Current working dir: /home/shashank/Desktop/OS >>>cd shell >>>pwd Current working dir: /home/shashank/Desktop/OS/shell >>></pre>

ls	-i -a	>>>ls >>>ls [path](optional)	It uses readdir() of the C library dirent.h and accesses	<p>This is an external command, it forks a child process, if the child process is not created properly, it throws “fork error” and prints an error message in the shell, the user however can resume his working. If it is unable to open a directory the responsible error message is printed. Also if the path entered is invalid, it is intimated to the user through an error message.</p> <pre>shashank@shashank-Lenovo-Legion-Y530-15ICH:~/Desktop/0 Current USER is: @shashank Current working dir: /home/shashank/Desktop/OS/shell >>>ls shell time . History.txt pwd.c ..</pre> <pre>realmkdir.c >>>ls -i 6427114 shell 6425348 time 6426558 . 6426577 History.txt 6425067 pwd.c 5506070 .. 6425068 pwd 6425349 time.c</pre>
----	----------	--	--	---

exit	No flags	>>>exit	exit() // this commands terminates the shell // All the open files are closed first and then the program exits with exit status 0.	It terminates the program. 
------	----------	---------	---	---

history	Default -c	>>>history >>>history -c	<pre>fopen(); read(); // This command opens a History.txt file which is maintained by the shell, in which all the commands entered by the user are stored in Chronological order. //It stores all commands since the beginning of the execution of the shell. // -c flag in history command is an instruction to clear the history, it overwrites History.txt and clears its previous contents.</pre>	<p>The error can be due to forced deletion of History.txt during the runtime which is handled by an error message. Errors me further include inability of reading History.txt file which is handled by the shell itself by printing an error message.</p>  <p>The screenshot shows a terminal window with the title 'shashank@shashank-Lenovo-Legion-Y530-15ICH: ~/Desktop/OS...'. The terminal output is as follows: >>>pwd Current working dir: /home/shashank/Desktop/OS/Assign1 >>>history pwd history history >>>exit Thank you for using this shell. shashank@shashank-Lenovo-Legion-Y530-15ICH:~/Desktop/OS/A Current USER is: @shashank Current working dir: /home/shashank/Desktop/OS/Assign1 >>>pwd Current working dir: /home/shashank/Desktop/OS/Assign1 >>>history pwd history >>>echo 12345 12345 >>>history pwd history echo 12345 history >>></p>
---------	---------------	-----------------------------	---	---

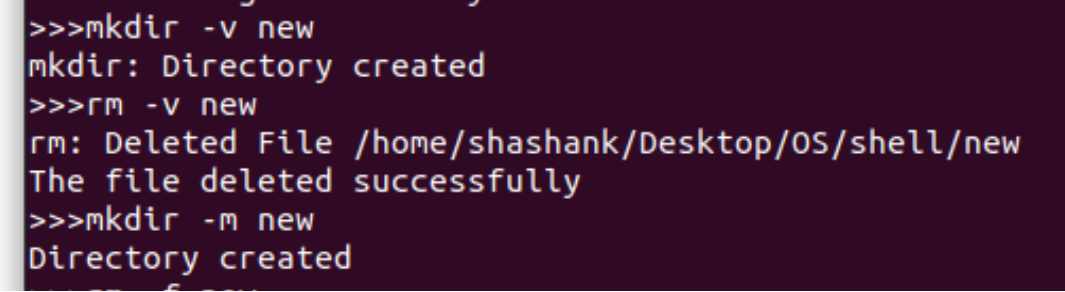
cd	-P (default) ~ .. .	>>>cd [destination] >>>cd ~ >>>cd ..	It uses chdir() from the C library unistd.h. This command is used to seek to a directory. This command take relative path of the destination as argument and takes the working directory of the shell to that directory. It returns NULL if no directory is found.	Cd command returns NULL if the argument is not a valid directory, the shell prints "Directory not found!! " message in case no directory is found on specified path. The change in directory can be seen using pwd command. 
----	------------------------------	---	---	---

cat	-n -i default	>>> cat -[flags](option al) args[File] >>>cat file.txt >>>cat -n file.txt >>>cat -i -n file.txt	It uses fopen(), fscanf() and printf() functions to open, parse and print the contents of the file.	<p>This is an external command, it forks a child process, if child process is not created properly, it throws “fork error” and prints an error message in the shell, user however can resume his working.</p> <p>The child process uses execl command to execute cat (Process Executable) calling it with specified path given by the shell and giving file names and flags as arguments.</p> <p>If the file doesn't exists in given relative path, cat command throws unable to find file error, the message is printed in the shell and exec process exits with status 1. However the shell is unaffected and user is able to provide next command even after the error message.</p>
-----	---------------------	--	---	--

```
}  
>>>cat -n time.c  
Here 2  
1 #include <stdio.h>  
2 #include <dirent.h>  
3 #include<string.h>  
4 #include <time.h>  
5 int main(int argc, char *argv[])  
6 {  
7     time_t t;
```

date	-u --debug (default)	>>>date >>>date -u >>>date --debug	It is implemented using the C functions like time(), gmtime(), mktime(), ctime() that are included in the header time.h	<p>This is an external command, it forks a child process, if child process is not created properly, it throws “fork error” and prints an error message in the shell, user however can resume his working.</p> <p>Date command detects for typos in spellings and also if invalid flags are entered. It prints a message when it detects any flag other than the two stated.</p> <pre>realmkdir.c >>>date Mon Sep 28 22:27:55 2020 IST >>>date -u Mon Sep 28 16:58:03 2020 UTC >>>date -debug Mon Sep 28 22:28:11 2020 IST >>>date -y Invalid Flag for Date -y >>></pre>
------	----------------------------	---	---	--

rm	<ul style="list-style-type: none">-f-v-d(default)	>>>rm -[flags](optional) files/directories	It is implemented using the C function remove().	<p>This is an external command, it forks a child process, if child process is not created properly, it throws “fork error” and prints an error message in the shell, user however can resume his working.</p> <p>The complete path of the file to be deleted is to be passed to the remove function of the program. The remove function checks if file exists and deletes it if it does otherwise returns a non zero value;</p> <p>That non zero return value is handled by the shell and the responsive error message is then printed.</p> <pre>Invalid Flag for Date -y >>>mkdir -v new mkdir: Directory created >>>rm -v new rm: Deleted File /home/shashank/Desktop/OS/shell The file deleted successfully >>></pre> <pre>>>>mkdir -m new Directory created >>>rm -f new >>></pre>
----	---	--	--	--

mkdir	-m -v	>>>mkdir -[flags](optional) [Name] >>>mkdir newD >>>mkdir -v newD >>>mkdir -m newD	This command uses C library function mkdir() to create a directory in the given path.	<p>This is an external command, it forks a child process, if child process is not created properly, it throws “fork error” and prints an error message in the shell, user however can resume his working.</p> <p>If the directory with same name already exists or if the folder needs to be authorised for creating directories then the command throws error and the user is intimated by printed messages on the shell.</p>  <pre>>>>mkdir -v new mkdir: Directory created >>>rm -v new rm: Deleted File /home/shashank/Desktop/OS/shell/new The file deleted successfully >>>mkdir -m new Directory created</pre>
-------	----------	---	---	---

echo	-E (default) -n	>>>echo -[flags] [arguments] >>>echo hello world >>>echo -n hello >>>echo -E hello	It uses simple printf() of stdio.h in C.	<p>The scope of errors here only resides in typos in flags and commands which are handled by the shell.</p> <pre>shashank@shashank-Lenovo-Legion-Y530-15ICH:~/Desktop/OS/s Current USER is: @shashank Current working dir: /home/shashank/Desktop/OS/shell >>>echo hello world hello world >>>echo -n Hello world Hello world>>>echo -E Hello Hello >>>echo -E this/n this/n >>>pwd</pre>
------	--------------------	--	---	--