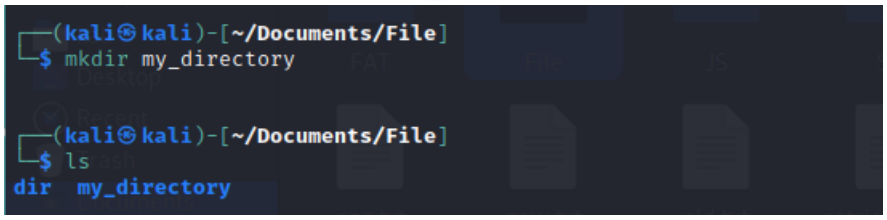
Winston Varghese Thomas
20BCY10014
VIT Bhopal
winston.varghese2020@vitbhopal.ac.in

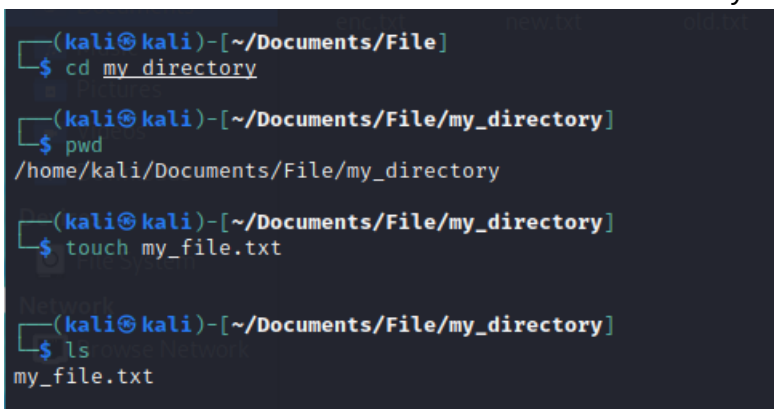
Assignment -2: Bash Shell Basics

Task 1: File and Directory Manipulation

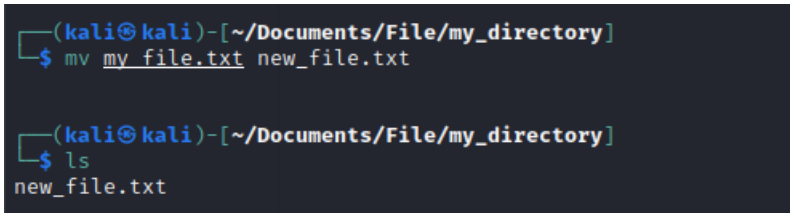
1. Create a directory called "my_directory".



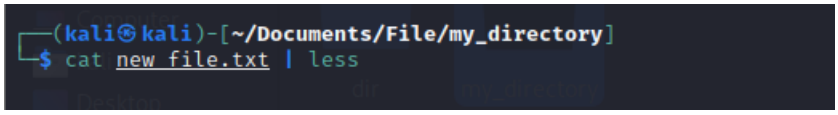
2. Navigate into the "my_directory".
3. Create an empty file called "my_file.txt".
4. List all the files and directories in the current directory.

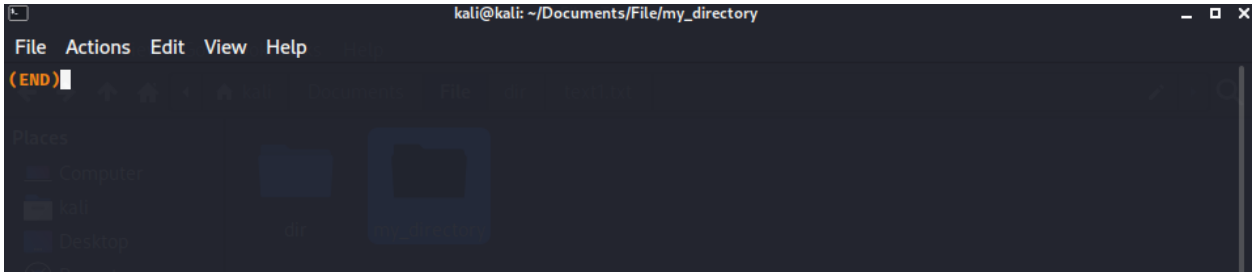


5. Rename "my_file.txt" to "new_file.txt".



6. Display the content of "new_file.txt" using a pager tool of your choice.





7. Append the text "Hello, World!" to "new_file.txt".

```
┌──(kali㉿kali)-[~/Documents/File/my_directory]
└─$ echo "Hello, World!" >> new_file.txt
```

```
┌──(kali㉿kali)-[~/Documents/File/my_directory]
└─$ cat new_file.txt
"Hello, World!"
```

8. Create a new directory called "backup" within "my_directory".

```
┌──(kali㉿kali)-[~/Documents/File/my_directory]
└─$ mkdir backup

┌──(kali㉿kali)-[~/Documents/File/my_directory]
└─$ ls
backup  new_file.txt
```

9. Move "new_file.txt" to the "backup" directory.
10. Verify that "new_file.txt" is now located in the "backup" directory.

```
┌──(kali㉿kali)-[~/Documents/File/my_directory]
└─$ mv new_file.txt backup/


┌──(kali㉿kali)-[~/Documents/File/my_directory]
└─$ cd backup

┌──(kali㉿kali)-[~/Documents/File/my_directory/backup]
└─$ ls
new_file.txt
```

11. Delete the "backup" directory and all its contents.

```
┌──(kali㉿kali)-[~/Documents/File/my_directory/backup]
└─$ cd ..

┌──(kali㉿kali)-[~/Documents/File/my_directory]
└─$ rm -r backup

┌──(kali㉿kali)-[~/Documents/File/my_directory]
└─$ ls
```

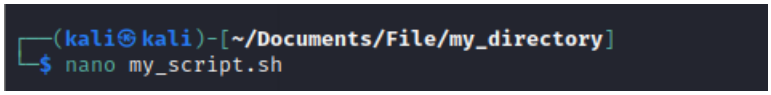
Task 2: Permissions and Scripting

- Create a new file called "my_script.sh".

- Edit "my_script.sh" using a text editor of your choice and add the following lines:

bash
#!/bin/bash
echo "Welcome to my script!"
echo "Today's date is $(date)."
Save and exit the file.

- Make "my_script.sh" executable.
- Run "my_script.sh" and verify that the output matches the expected result.

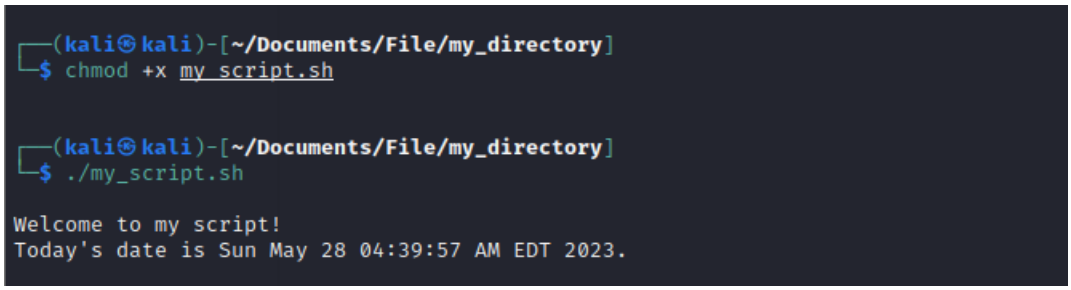


STEPS

Save and exit the file in the nano editor:
Press Ctrl+O to save the file.
Press Enter to confirm the filename.
Press Ctrl+X to exit the nano editor.

Make "my_script.sh" executable using the chmod command:

This command grants execute permission to the file.
Run "my_script.sh" using the ./ notation to execute it:



Task 3: Command Execution and Pipelines

- List all the processes running on your system using the "ps" command.

```
┌──(kali㉿kali)-[~/Documents/File/my_directory]
└─$ ps
    PID TTY          TIME CMD
   1602 pts/0    00:00:00 zsh
   1668 pts/0    00:00:00 less
   1709 pts/0    00:00:00 less
   1870 pts/0    00:00:00 ps

┌──(kali㉿kali)-[~/Documents/File/my_directory]
└─$ ps aux
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root           1  0.1  0.5 165892 10948 ?        Ss   04:12   0:01 /sbin/init splash
root           2  0.0  0.0      0     0 ?        S    04:12   0:00 [kthreadd]
root           3  0.0  0.0      0     0 ?        I<   04:12   0:00 [rcu_gp]
root           4  0.0  0.0      0     0 ?        I<   04:12   0:00 [rcu_par_gp]
root           5  0.0  0.0      0     0 ?        I    04:12   0:00 [kworker/0:0-events_freezable_power_]
root           6  0.0  0.0      0     0 ?        I<   04:12   0:00 [kworker/0:0H-events_highpri]
root           9  0.0  0.0      0     0 ?        I<   04:12   0:00 [mm_percpu_wq]
root          10  0.0  0.0      0     0 ?        S    04:12   0:00 [rcu_tasks_rude_]
root          11  0.0  0.0      0     0 ?        S    04:12   0:00 [rcu_tasks_trace]
root          12  0.0  0.0      0     0 ?        S    04:12   0:00 [ksoftirqd/0]
root          13  0.0  0.0      0     0 ?        I    04:12   0:00 [rcu_sched]
root          14  0.0  0.0      0     0 ?        S    04:12   0:00 [migration/0]
root          15  0.0  0.0      0     0 ?        S    04:12   0:00 [cpuhp/0]
root          16  0.0  0.0      0     0 ?        S    04:12   0:00 [cpuhp/1]
root          17  0.0  0.0      0     0 ?        S    04:12   0:00 [migration/1]
root          18  0.0  0.0      0     0 ?        S    04:12   0:00 [ksoftirqd/1]
root          20  0.0  0.0      0     0 ?        I<   04:12   0:00 [kworker/1:0H-events_highpri]
root          22  0.0  0.0      0     0 ?        I    04:12   0:00 [kworker/u4:1-flush-8:0]
root          23  0.0  0.0      0     0 ?        S    04:12   0:00 [kdevtmpfs]
root          24  0.0  0.0      0     0 ?        I<   04:12   0:00 [netns]
root          25  0.0  0.0      0     0 ?        S    04:12   0:00 [kauditd]
root          26  0.0  0.0      0     0 ?        S    04:12   0:00 [khungtaskd]
root          27  0.0  0.0      0     0 ?        S    04:12   0:00 [oom_reaper]
root          28  0.0  0.0      0     0 ?        I<   04:12   0:00 [writeback]
root          29  0.0  0.0      0     0 ?        S    04:12   0:00 [kcompactd0]
root          30  0.0  0.0      0     0 ?        SN   04:12   0:00 [ksmd]
root          31  0.0  0.0      0     0 ?        SN   04:12   0:00 [khugepaged]
```

- Use the "grep" command to filter the processes list and display only the processes with "bash" in their name.

```
┌──(kali㉿kali)-[~/Documents/File/my_directory]
└─$ ps aux | grep bash

kali        1883  0.0  0.1   6480  2240 pts/0    S+   04:43   0:00 grep --color=auto bash
```

- Use the "wc" command to count the number of lines in the filtered output.

```
┌──(kali㉿kali)-[~/Documents/File/my_directory]
└─$ ps aux | grep bash

kali        1883  0.0  0.1   6480  2240 pts/0    S+   04:43   0:00 grep --color=auto bash

┌──(kali㉿kali)-[~/Documents/File/my_directory]
└─$ ps aux | grep bash | wc -l

1
```