Assessment-2
Title: Bash Shell Basics

NAME-SHREYA DOKANIA

Task 1: File and Directory Manipulation

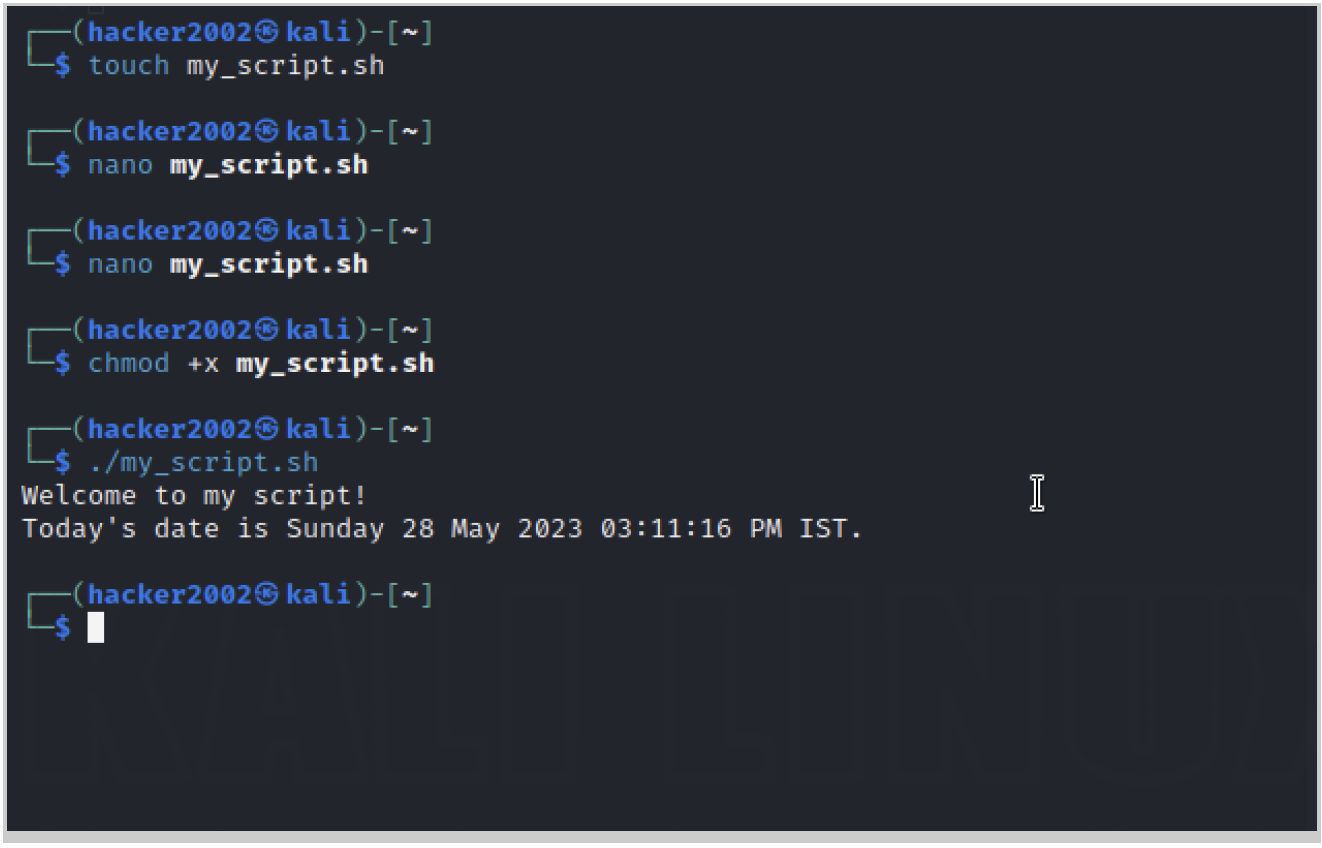
1. Create a directory called "my_directory".
2. Navigate into the "my_directory".
3. Create an empty file called "my_file.txt".
4. List all the files and directories in the current directory.
5. Rename "my_file.txt" to "new_file.txt".
6. Display the content of "new_file.txt" using a pager tool of your choice.
7. Append the text "Hello, World!" to "new_file.txt".
8. Create a new directory called "backup" within "my_directory".
9. Move "new_file.txt" to the "backup" directory.
10. Verify that "new_file.txt" is now located in the "backup" directory.
11. Delete the "backup" directory and all its contents.

```
┌──(hacker2002㉿kali)-[~]
└─$ mkdir mydir

┌──(hacker2002㉿kali)-[~]
└─$ cd mydir

┌──(hacker2002㉿kali)-[~/mydir]
└─$ touch my_file.txt

┌──(hacker2002㉿kali)-[~/mydir]
└─$ ls
my_file.txt

┌──(hacker2002㉿kali)-[~/mydir]
└─$ mv my_file.txt new_file.txt

┌──(hacker2002㉿kali)-[~/mydir]
└─$ cat new_file.txt

┌──(hacker2002㉿kali)-[~/mydir]
└─$ echo 'hello world' >>new_file.txt

┌──(hacker2002㉿kali)-[~/mydir]
└─$ cat new_file.txt
hello world

┌──(hacker2002㉿kali)-[~/mydir]
└─$ mkdir backup

┌──(hacker2002㉿kali)-[~/mydir]
└─$ cd backup
```

```
┌──(hacker2002㉿kali)-[~/mydir]
└─$ mv new_file.txt backup/

┌──(hacker2002㉿kali)-[~/mydir]
└─$ cd backup

┌──(hacker2002㉿kali)-[~/mydir/backup]
└─$ ls
new_file.txt

┌──(hacker2002㉿kali)-[~/mydir/backup]
└─$ cd ..

┌──(hacker2002㉿kali)-[~/mydir]
└─$ rm -r backup

┌──(hacker2002㉿kali)-[~/mydir]
└─$ ls

┌──(hacker2002㉿kali)-[~/mydir]
└─$
```

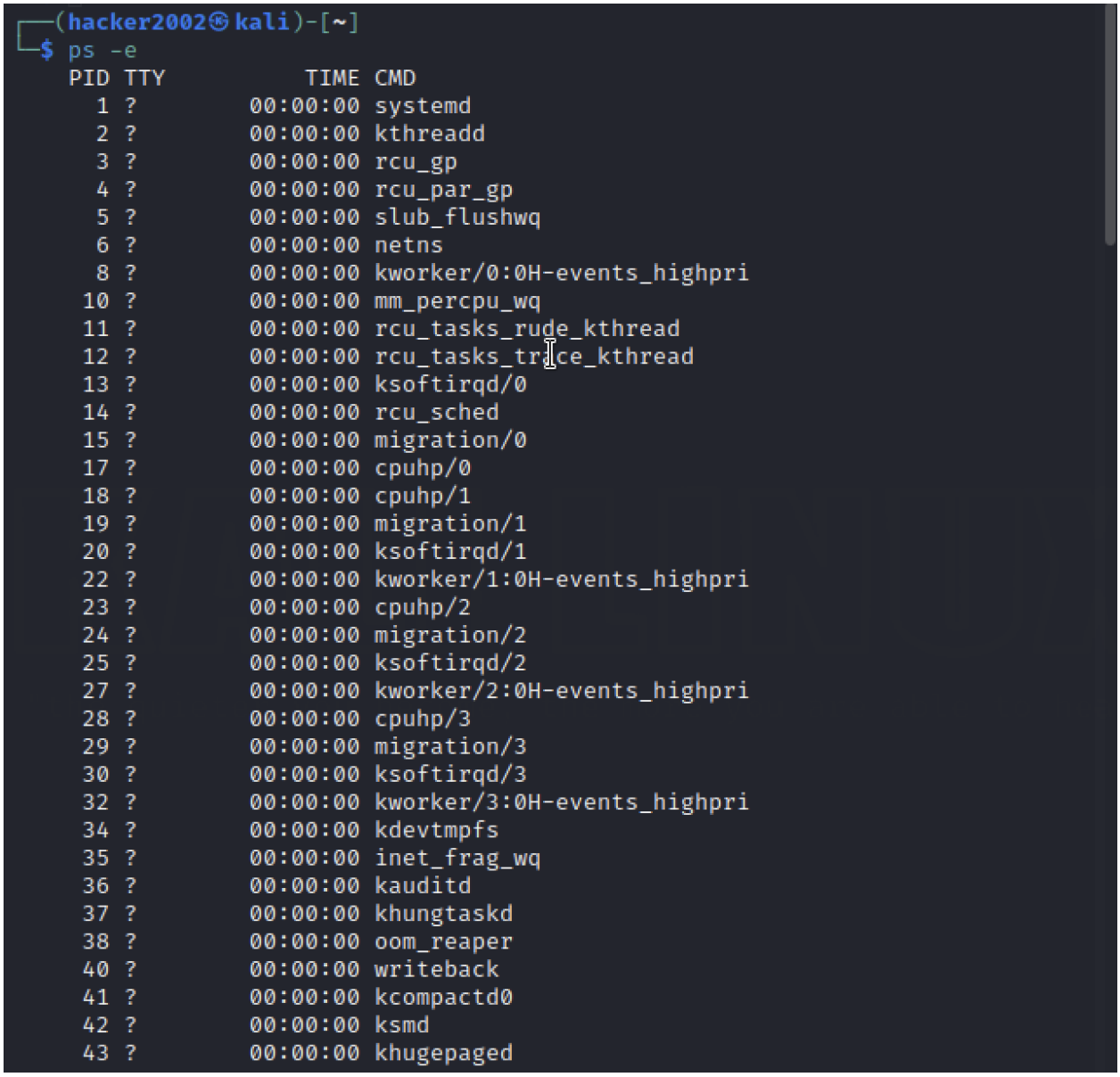
Task 2: Permissions and Scripting

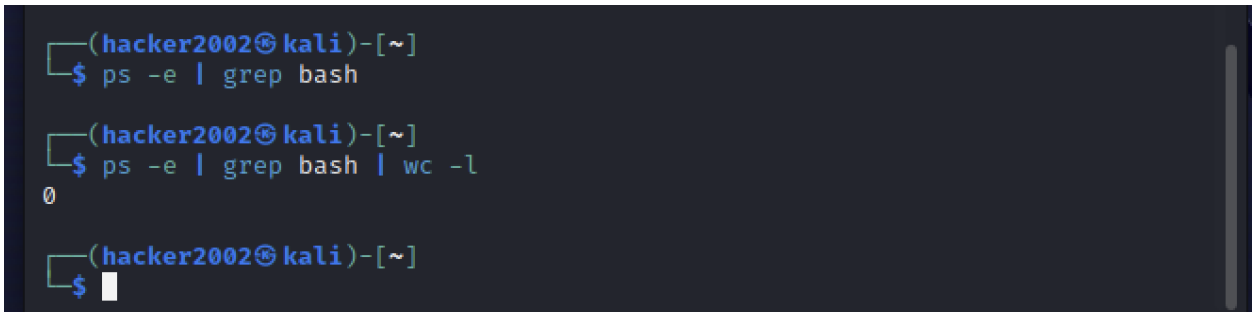
- Create a new file called "my_script.sh".
- Edit "my_script.sh" using a text editor of your choice and add the following lines:
  - bash
  - #!/bin/bash
  - echo "Welcome to my script!"
  - echo "Today's date is $(date)."
  - Save and exit the file.
- Make "my_script.sh" executable.
- Run "my_script.sh" and verify that the output matches the expected result.



Task 3: Command Execution and Pipelines

- List all the processes running on your system using the "ps" command.
- Use the "grep" command to filter the processes list and display only the processes with "bash" in their name.
- Use the "wc" command to count the number of lines in the filtered output.




(hacker2002㉿kali)-[~]
$ ps -e | grep bash
(hacker2002㉿kali)-[~]
$ ps -e | grep bash | wc -l
0
(hacker2002㉿kali)-[~]
$