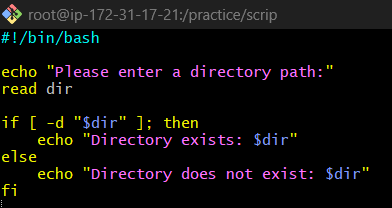
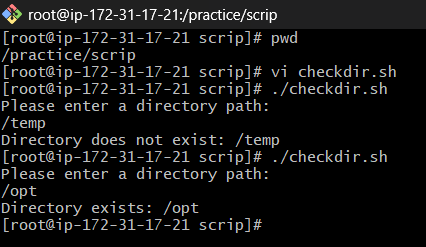
BASH SCRIPTING TASKS + CRONTAB ENTRIES

* Use absolute path to run the script from anywhere else, to run a script using the absolute path, you must specify the full path from the root ***“ / ”*** directory - ***bash /home/User123/practice/scrip/check\_dir.sh***
* A shebang **(#!)** is the first line in a script that tells the system which interpreter (like Bash, Python, etc.) should be used to run the file. **/bin/bash** is the absolute path to the Bash interpreter.
* To execute/run the script ,we need to change the permission of the file to execute it. use **chmod +x <file\_name>** to give execute permission to all the users or **chmod 755 <file\_name>**
* After this we can execute this file using **./filename.bash** or **bash filename.sh**

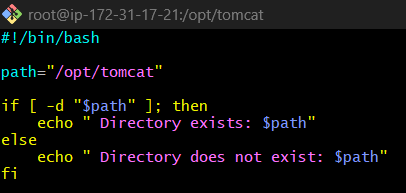
1) Create on Bash script to check if a directory is available or not.

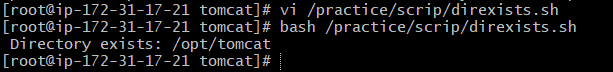
* manual method where it asks the user to enter the directory path and it checks if such directory exists anywhere.



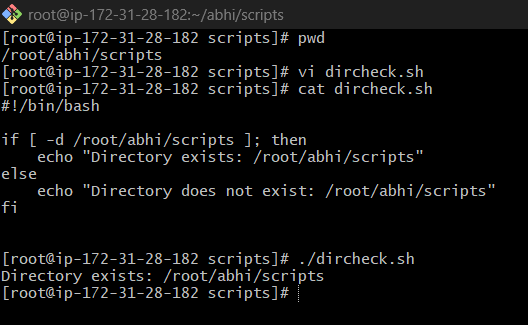


* hardcoded version - doesnt ask for input, useful for Automation,cron jobs, useful in Auto-installation scripts, prevents overwriting or reinstalling

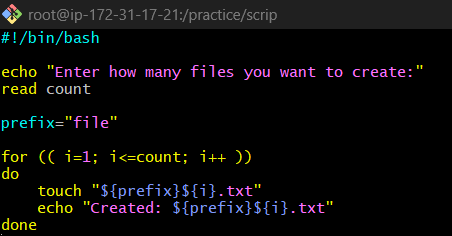


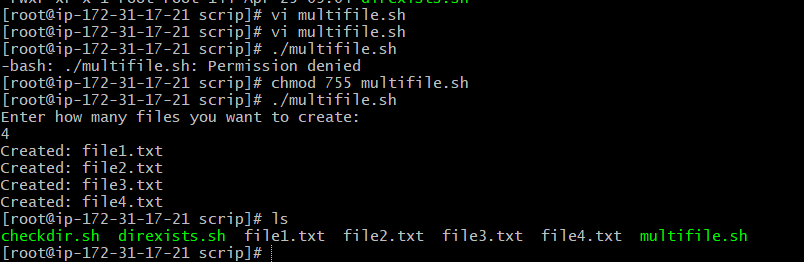


(OR)

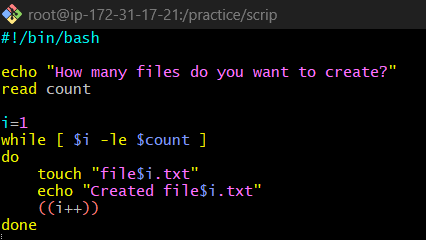


2) Create a bash script which will create multiple files.

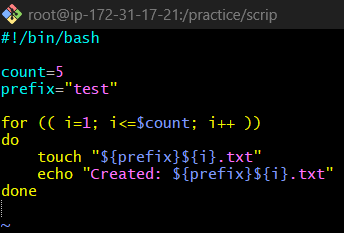


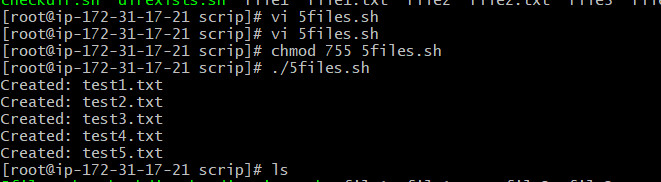


another way

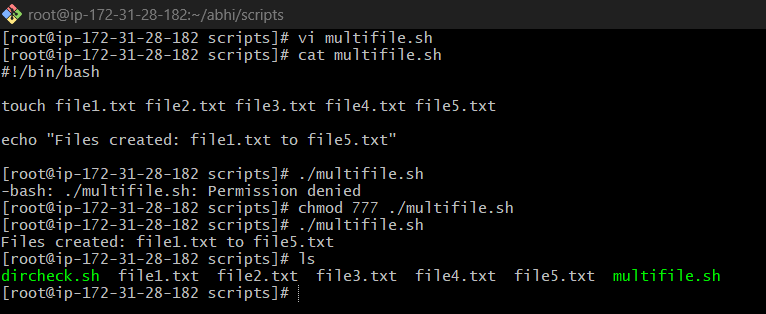


hardcoded-

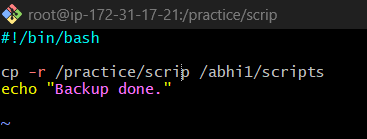


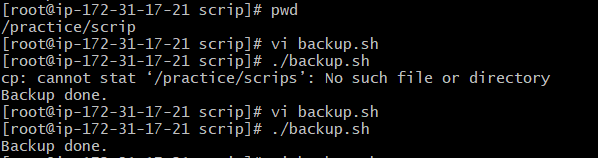


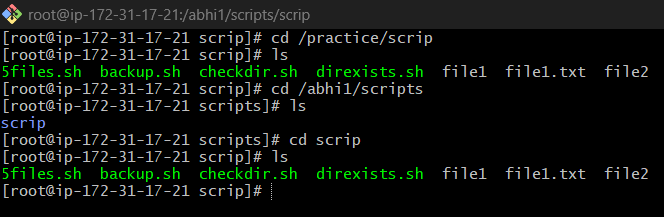
or



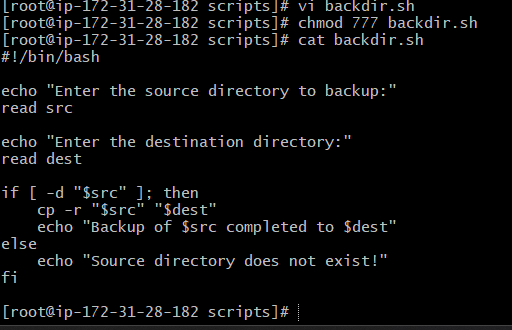
3) Create a bash script to take backup of a directory.

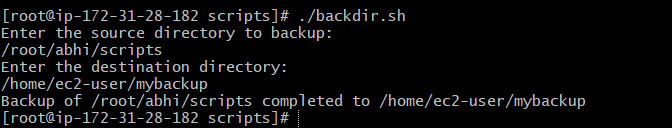


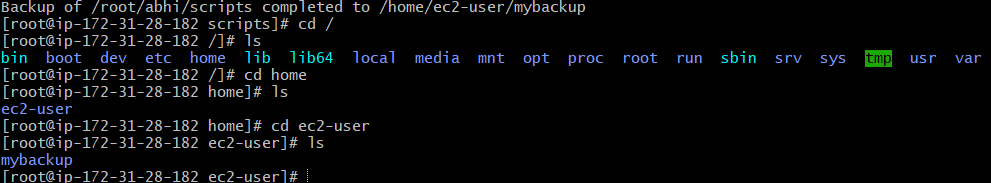




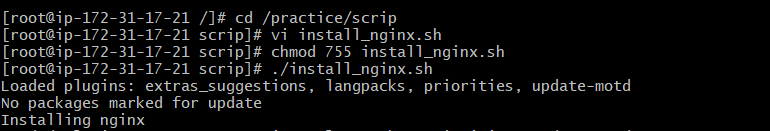
Another way

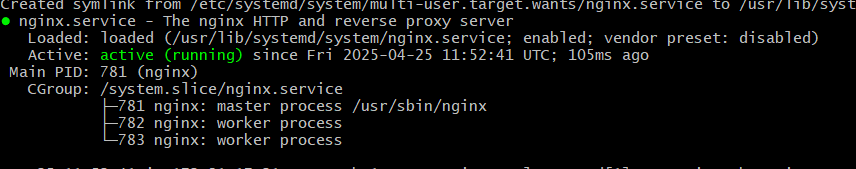


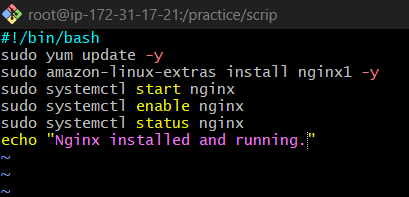




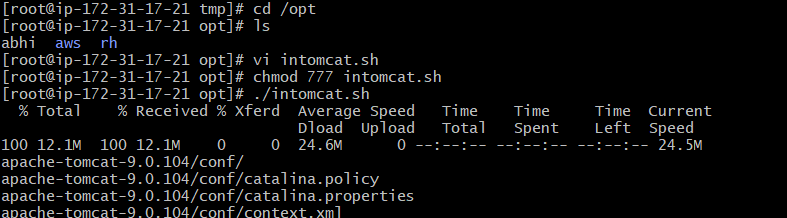
4) Create a bash script to install nginx in ec2 server.

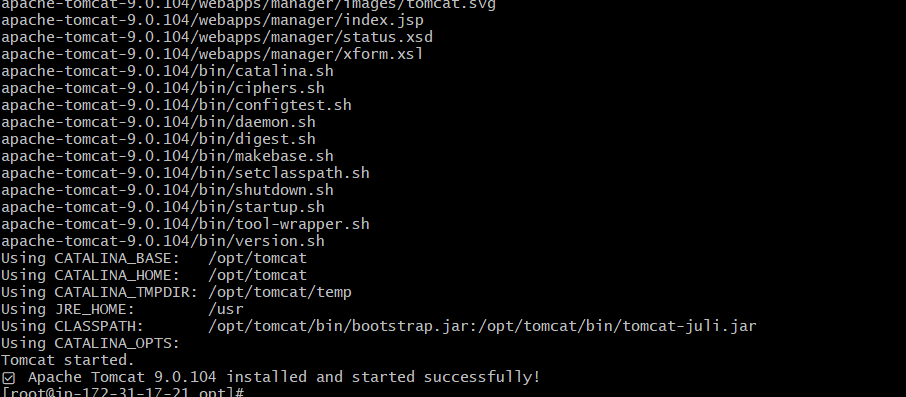


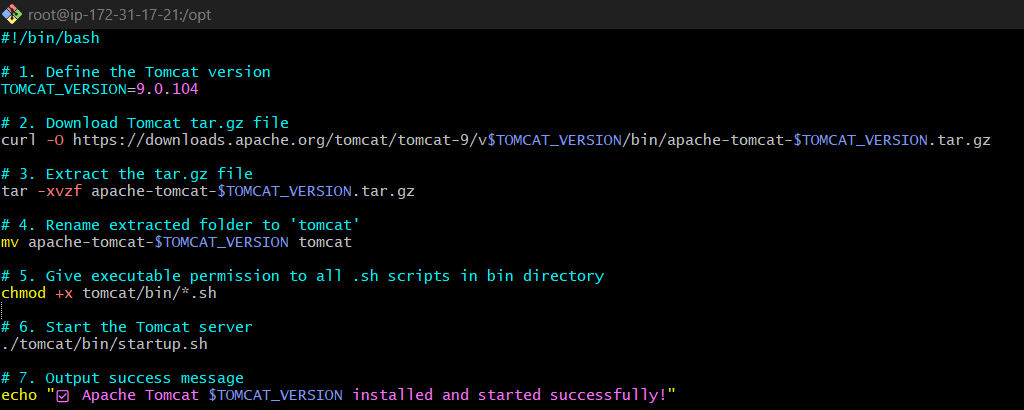




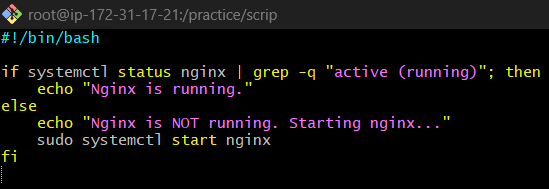
5) Create a bash script to install ApacheTomcat in ec2 server.

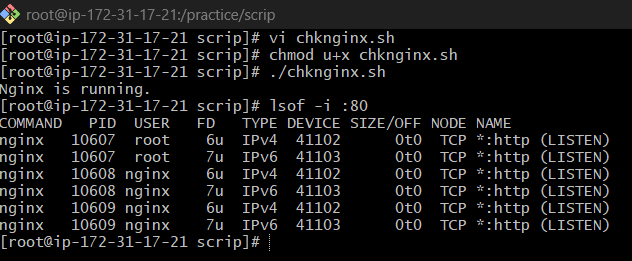




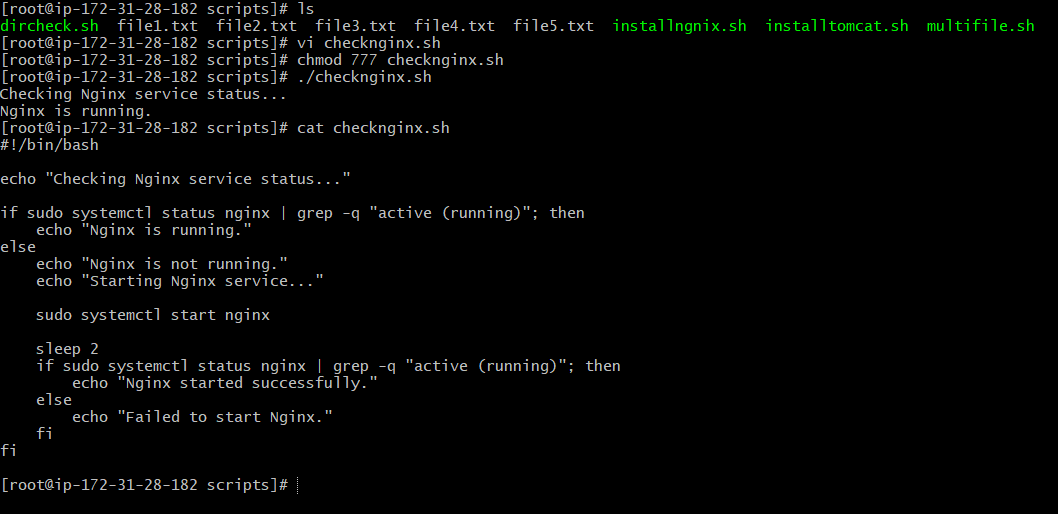


6) Create a bash script to check list if nginx service is running or not,if not running then script should start the service.

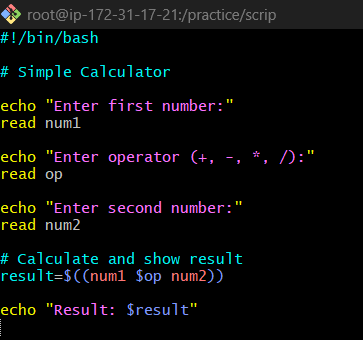


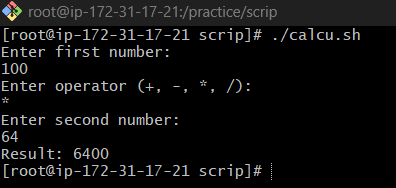


similar, slight difference-

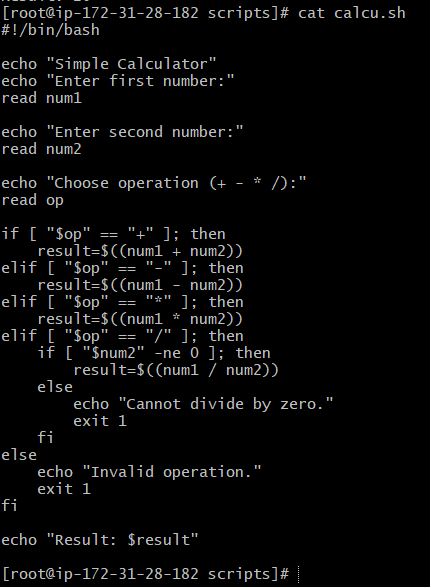
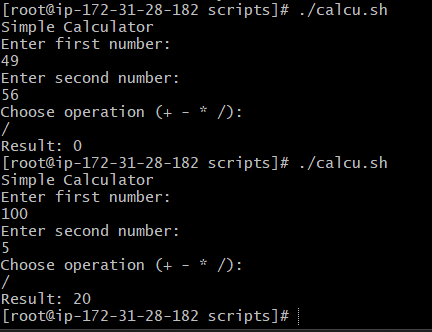


7) Create a bash script for calculator.

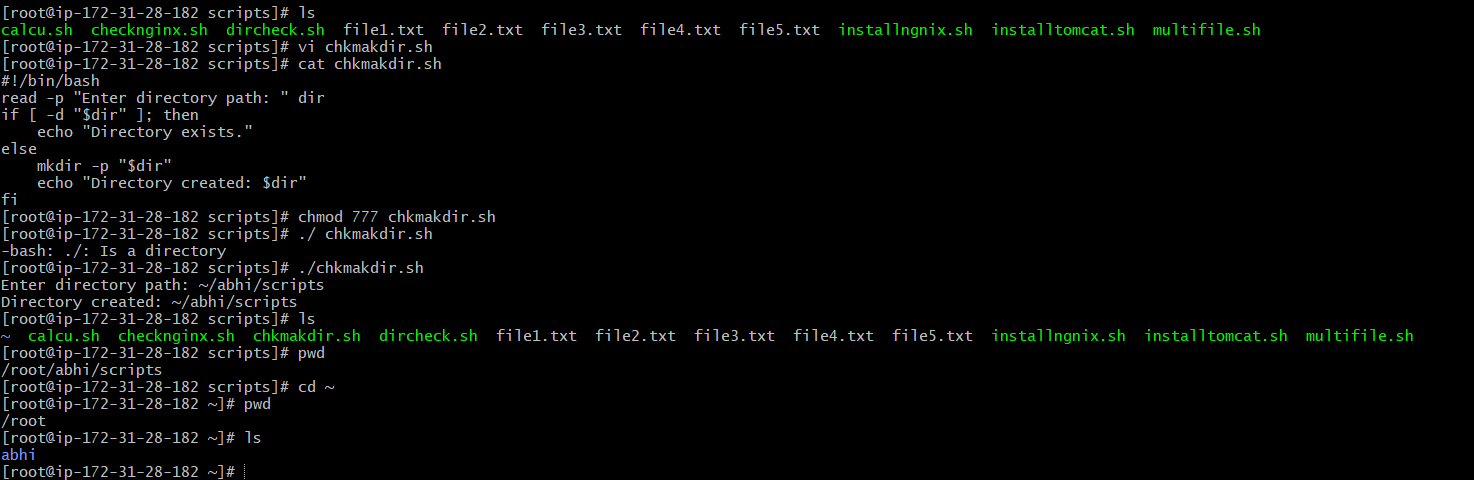




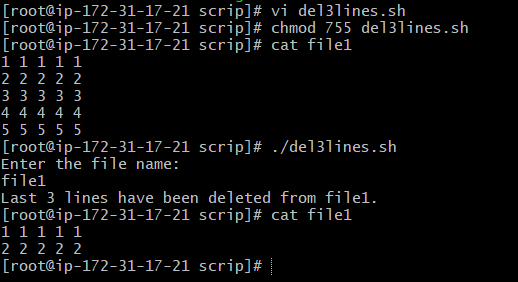
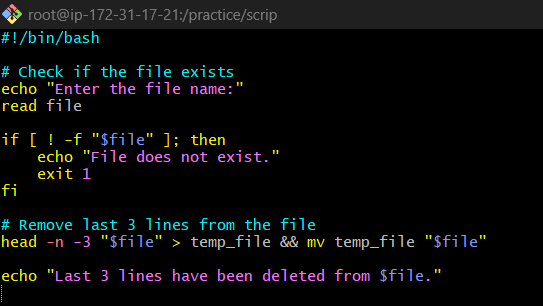
another way



8) Create a bash script to check if directory is avaialble or not,if not then create a directory.

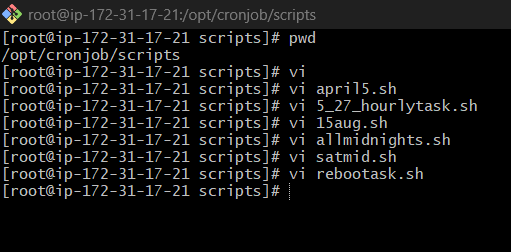


9) Create bash script to delete last 3 lines for a file.

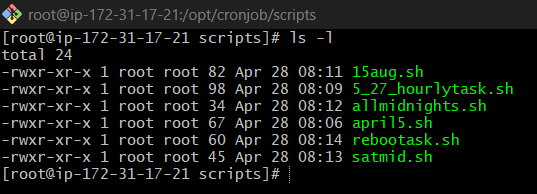


**crontab entries practice**

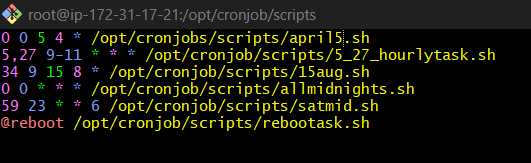
first make all the scripts



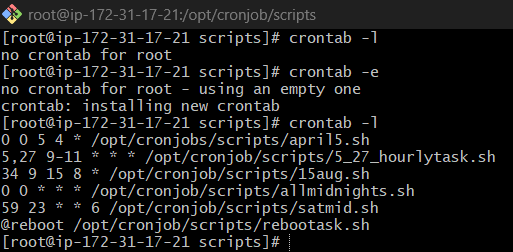
change the permissions



crontab -e and assign each script to the time you need them to run at



crontab -l to see the list of cronjobs

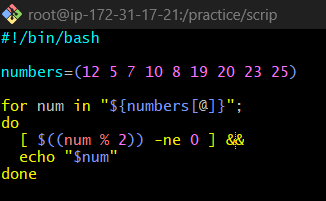


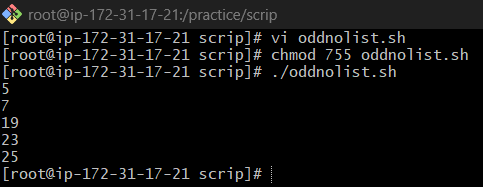
**PART 2 OF TASKS**

1) Bash script to print odd numbers from the list.

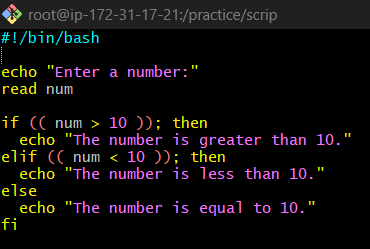
(12,5,7,10,8,19,20,23,25)

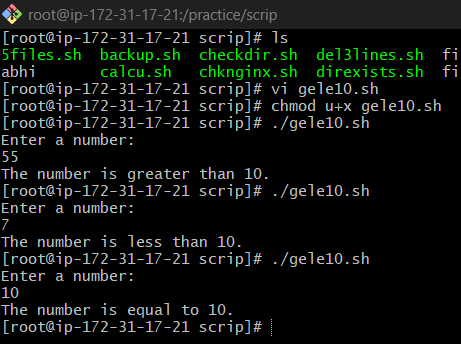
* we use {numbers[@]} if we want each element individually to be printed .
* we can also use {numbers[\*]}, if we want them in a single string with spaces between them.



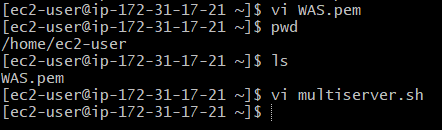


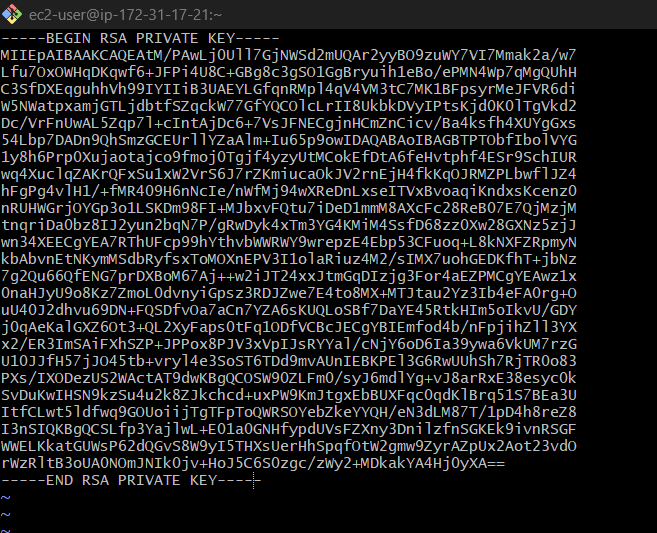
2) Bash script to take a input from user and check if it is greater than or less than 10

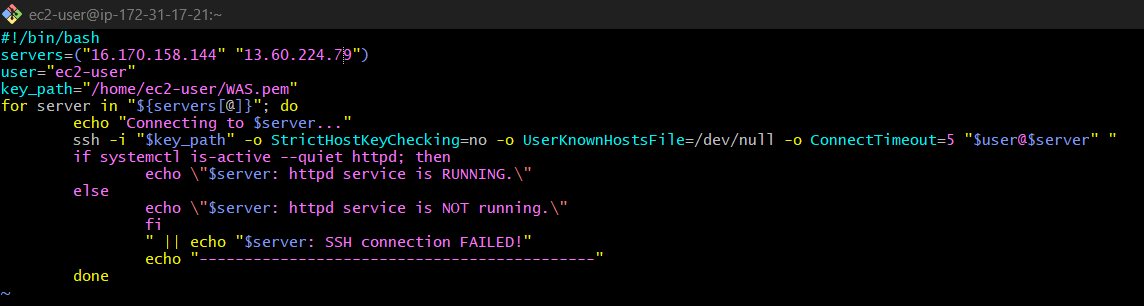


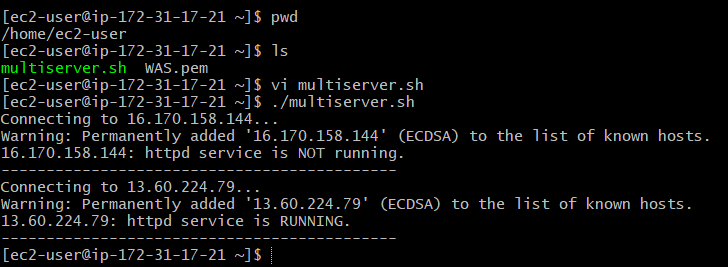


3) Bash script to login to multiple servers and check if httpd service is running or not

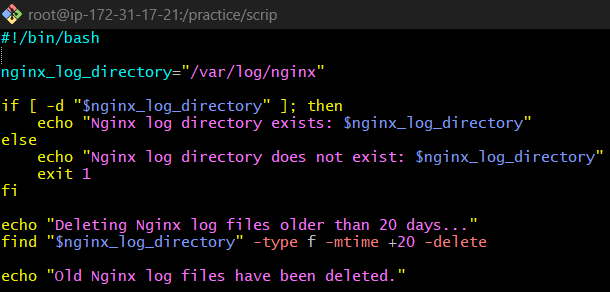


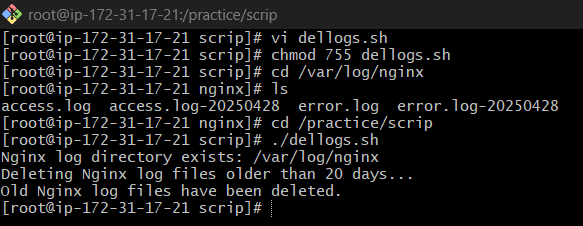




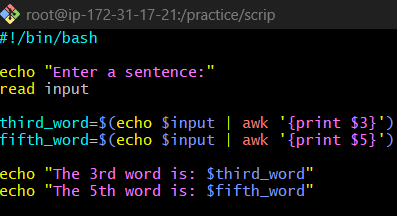


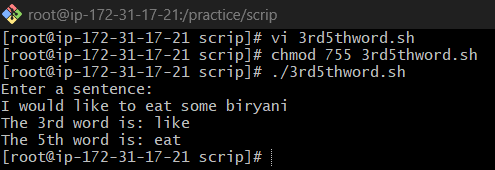
4) Bash script to check the log files from a path and delete files older than 20 days.





5) Bash script to print 3rd word and 5th word from the given input of user





echo $input outputs the sentence that was read from the user.

The output is then piped (|) to the awk command.

awk '{print $3}':

awk splits the sentence into fields (words) by default using spaces.

{print $3} tells awk to print the third word from the input.

The $( ... ) syntax captures the output of the command and assigns it to the variable third\_word.

Similarly, fifth\_word=$(echo $input | awk '{print $5}') extracts the 5th word from the sentence.

6) Bash script to print numbers between 1 to 100 and then seperate the odd numbers and even numbers

