Linux Assessment-2

1. What is the command to make a Bash script executable? Describe how you would give execution permissions to a script named backup.sh.

***To execute : ./backup.sh***

***To execute permission : chmod +x backup.sh***

1. What command would you use to count the number of lines in a file called data.txt?

***Count size :wc-l data.txt***

1. How would you extract the current date and time in a Bash script?

***current\_date=$(date+”%y%m%d”)***

***current \_time=$(date+”%H%M%S”)***

1. How do you execute a Bash script named script.sh from the terminal?

***Execute bashscript : sh script.sh***

1. What is the purpose of # in a Bash script?

***# purpose is to provide context about the upcoming code.***

1. How do you print "Hello, World!" to the terminal in a Bash script?

***echo” Hello ,world”***

1. What command is used to read user input in a Bash script?

***read***

1. How do you assign the value 10 to a variable named count in a Bash script?

***count=10***

1. What does the command chmod 755 script.sh do to the permissions of script.sh?

c***hmod 755=rwxr-xr-x***

1. What does the command chmod 644 file.txt accomplish?

***chmod644=rw-r--r--***

1. What does the command chmod u+x,g-w script.sh do?

***chmod u+x=add execute permission to username(--x------)***

***chmod g-w=remove write permission to group(rwxr-xrwx)***

1. How can you remove execute permissions from a script named script.sh for all users?

***Chmod -x script.sh***

13.How can you change the permissions of all scripts in the current directory to be executable by the user and group but not by others?

***Execute permission for user and group= Chmod ug+x***

***Not execute permission for others =chmod o-x***

1. How would you check if a file exists using an if statement in Bash?

***file=”practice.txt”***

***if [ -e “$ file”];then***

***echo “File exists $file”***

***else***

***echo”File not exists $file”***

**fi**

1. If a user tries to execute a script but receives a "Permission denied" error, what steps would you take to troubleshoot this?

***ls-l***

***chmod +x filename***

1. Write a Bash script that accepts two arguments (file and directory) and checks if the given file exists. If it does, move it to the provided directory.

***file = $1***

***directory =$2***

***if [ -e “ $1”];then***

***echo “The given file exists $1”***

***mv “file$1” “directory$2”***

***else***

***echo”The given file not exit”***

***fi***

1. How can you append output to a file instead of overwriting it?

***redirect operator(>>)***

1. How do you create a directory named myfolder in a Bash script, ensuring that it doesn't throw an error if the directory already exists?

***mkdir -p myfolder***

1. Write a Bash command that checks if a directory exists, and if not, creates it.

i***f [ -d ”$Directory ”];then***

***echo “The Directory exists”***

***else***

***echo”The Directory not exists”***

***mkdir Directory***

***fi***

1. How would you check if a variable named x is equal to 5 in a Bash script?

***echo-n ”enter the value of X:”***

***read X***

***if [ $X -eq5 ];then***

***echo “X is equal to 5”***

***else***

***echo “X is not equal to 5”***

fi

1. Create a script that accepts a list of filenames as arguments and checks if each file exists. If a file does not exist, it should create an empty file with that name.
2. Write a shell script that takes a directory as input and performs the following operations:

* Lists all files and directories in the given directory.

***Mkdir example***

***cd example***

***touch script.txt***

***ls-l example***

* Counts and displays the number of files and directories separately.

***ls | wc-l example***

1. Write a shell script that creates a new file named example.txt in the current directory, write the text "Hello, World!" to the file, and saves it.

***Mkdir example***

***cd example***

***pwd***

***touch example.txt***

***vim example.txt***

***echo “hello world”***

***:wq!***

***Chmod +x example.txt***

***./example.txt***

1. Write a shell script that copies the example.txt file to a new file named example\_backup.txt and then renames example.txt to example\_old.txt.

***Cp example.txt backup.txt***

***mv example.txt example old.txt***

1. Write a shell script that reads a date in the format YYYY-MM-DD from user input and converts it to the format DD/MM/YYYY.

***$ date -d “2024-10-16”+’%d/%m/%y’***

1. Write a shell script that prompts the user to enter a filename, then displays the content of that file if it exists.

***Touch filename.txt***

***Read “$filename”***

***if [ -e “$filename” ];then***

***echo ”The content is: $filename”***

***else***

***echo ”filename not exsits”***

***fi***

1. Write a Bash script that automates the process of backing up a directory to a specified location. The script should create a timestamped archive file.

1. Write a script that automates the cleanup of temporary files in a specified directory. The script should remove files older than a specified number of days.
2. Create a Bash script that accepts a username and a password as arguments. The script should create a new user with the provided username and set the password to the provided value.

***USERNAME=$1***

***PASSWORD=$2***

***sudo user\_add -m”$USERNAME”***

***sudo chpasswd”$PASSWORD”***

1. Write a Bash script that reads a list of usernames from a file and creates user accounts on a Linux system for each username. Use a simple default password for all the new users.
2. Write a Bash script that lists all currently running processes and filters the output to display only processes belonging to a specific user. The username should be provided as an argument to the script.

***USERNAME=$1***

***ps-u “$USERNAME”***

1. Write a Bash script that uses the top command to display the top 5 memory-consuming processes on the system and saves the output to a file named top\_processes.txt.

***Vim processes.txt***

t***op processes.txt***

***ps aux –sort=-%mem***

***:wq!***

1. Create a Bash script that lists all the files in the /home/ubuntu directory and displays their sizes, sorting the output by size in descending order.

l***s /home/ubuntu directory***

***du-h ”ubuntu directory”/\* | sort-hr***

1. Create a script that outputs the total number of processes currently running on the system, along with the total number of users logged in.

***total \_processes=$(ps aux|wc -1)***

***total\_users=$( who | wc-l)***