Assignment 1: Running Linux Commands

Part 1: 40 Linux Commands

1. pwd – pwd is used to display the location of the current working directory.
2. mkdir – mkdir is used to create a new directory under any directory.
3. rmdir - rmdir command is used to delete a directory.
4. ls - ls command is used to display a list of content of a directory.
5. Cd - cd command is used to change the current directory.
6. touch - The touch command is used to create empty files
7. cat - cat command can be used to create a file, display content of the file, copy the content of one file to another file
8. rm - rm command is used to remove a file.
9. cp - command is used to copy a file or directory.
10. mv - command is used to move a file or a directory form one location to another location.
11. rename - rename command is used to rename files.
12. head - command is used to display the content of a file. It displays the first 10 lines of a file.
13. tail - tail displays the last ten lines of the file content.
14. tac - tac displays the file content in reverse order (from the last line).
15. more - more command displays screenful output at a time.
16. id - command is used to display the user ID (UID) and group ID (GID).
17. useradd - command is used to add or remove a user on a Linux server.
18. passwd - passwd is used to create and change the password for a user.
19. ps - Display currently running processes.
20. top – Show real-time process monitoring.
21. htop – Interactive process monitoring (if installed).
22. df -h – Show disk space usage.
23. du -sh - Show directory size.
24. free -h - show free memory.
25. uptime – Show system uptime.
26. who – List logged-in users.
27. uname -a – Show system information.
28. ifconfig – Display network interfaces
29. ip a – Show IP addresses of network interfaces.
30. netstat -tulnp – Show active network connections.
31. ss -tulnp – Show open ports and listening services.
32 . curl – Fetch data from a URL.
33 . clear - it is useful to remove clutter from the terminal.

34. find – it is used to Search for files
35. nano - Open a file in the Nano text editor
36. echo – Print text on the terminal or write to a file
37. whoami – Show the current user
38. realpath - it is used to display the absolute path of a file or directory.
39. ping google.com - Tests network connectivity
40. netstat - Shows network statistics

Run all commands on Linux.

```
601  clear
602  pwd
603  mkdir test_dir
604  rmdir test_dir
605  ls
606  cd Documents
607  touch example.txt
608  cat example.txt
609  rm example.txt
610  touch file1.txt
611  cp file1.txt file2.txt
612  mv file1.txt ~/Documents/
613  mv file2.txt renamed_file.txt
614  head test.txt
615  tail test.txt
616  tac test.txt
617  more test.txt
618  id
619  sudo useradd newuser
620  ps
621  top
622  htop
623  df -h
624  du -sh
625  free -h
626  uptime
627  who
628  uname -a
629  sudo ifconfig
630  ip a
631  sudo netstat -tulnp
632  sudo ss -tulnp
633  curl https://www.google.com
634  clear
635  find . -name "file1.txt"
636  nano testfile.txt
637  echo "Hello, World!" > testfile.txt
638  whoami
639  realpath testfile.txt
640  ping google.com
641  netstat
642  history
643  echo "User: $(whoami), Hostname: $(hostname)" && history | tail -n 40
```

Part 2: Shell Scripts

1.Print a given number in reverse order and sum of individual digits

Code:

```
#!/bin/bash
echo "Enter a number:"
read number
reverse=0
sum=0
while [ $number -gt 0 ]
do
    digit=$((number % 10))
    sum=$((sum + digit))
    reverse=$((reverse * 10 + digit))
    number=$((number / 10))
done
echo "Reversed number: $reverse"
echo "Sum of digits: $sum"
```

```
(base) tetarwal005@hp:~/c_program$ nano reverse_sum.sh
(base) tetarwal005@hp:~/c_program$ chmod +x reverse_sum.sh
(base) tetarwal005@hp:~/c_program$ ./reverse_sum.sh
Enter a number:
571
Reversed number: 175
Sum of digits: 13
(base) tetarwal005@hp:~/c_program$
```

Explanation:

- The script prompts the user to enter a number.
- It then initializes two variables reverse to store the reversed number and sum to store the sum of digits.
- A while loop extracts the last digit of the number using number%10 adds the digit to sum, and reverses the number by multiplying the current reverse by 10 and adding the digit.

- After the loop, it prints the reversed number and the sum of digits.

2. Accept one integer argument and print its multiplication table

```
(base) tetarwal005@hp:~/c_program$ nano multiplication_table.sh
(base) tetarwal005@hp:~/c_program$ chmod +x multiplication_table.sh
(base) tetarwal005@hp:~/c_program$ ./multiplication_table.sh 5
Multiplication Table for 5:
5 x 1 = 5
5 x 2 = 10
5 x 3 = 15
5 x 4 = 20
5 x 5 = 25
5 x 6 = 30
5 x 7 = 35
5 x 8 = 40
5 x 9 = 45
5 x 10 = 50
(base) tetarwal005@hp:~/c_program$
```

Code:

```
#!/bin/bash
if [ $# -eq 0 ]; then
    echo "Please provide a number as an argument."
    exit 1
fi
number=$1
echo "Multiplication Table for $number:"
for i in {1..10}
do
    result=$((number * i))
    echo "$number x $i = $result"
done
```

Explanation:

- The script checks if the user provided an argument. If not, it prints a message and exits.

- The argument passed is assigned to the variable number.
- A for loop iterates from 1 to 10 and prints the multiplication of the given number with each iteration.

Assignment 2: Running a simple C program on linux

```
(base) tetarwal005@hp:~/c_program$ ls
a.out  hello  hello.c
(base) tetarwal005@hp:~/c_program$ gcc hello.c -o hello
(base) tetarwal005@hp:~/c_program$ ./hello
hello, Linux World!
(base) tetarwal005@hp:~/c_program$
```

Code:

```c
#include <stdio.h>
int main()
{  printf("hello, Linux World!\n");
    return 0;
}
```

Explanation:

- The correct command to compile a C program is gcc hello.c -o hello.
- The -o option is used to specify the output file name instead of using the default a.out.
- Running ./hello executes the compiled program and prints Hello, World!.
- The command gcc hello.c - hello.o is incorrect due to the unnecessary use of - and hello.o.
- The - character tells gcc to read from standard input, which is not needed in this case.

Assignment 3: Shell scripts - Research on Shell scripts and solve these questions.

1: Script to Print All .txt and .c Files

```
(base) tetarwal005@hp:~/c_program$ chmod +x check_files.sh
(base) tetarwal005@hp:~/c_program$ ./check_files.sh
Existing .txt files:
Existing .c files:
hello.c
(base) tetarwal005@hp:~/c_program$
```

code:

```
txt_files=$(ls *.txt 2>/dev/null)
c_files=$(ls *.c 2>/dev/null)

# If no .txt or .c files are found, create dummy files
if [ -z "$txt_files" ] && [ -z "$c_files" ]; then
    echo "No .txt or .c files found. Creating dummy files..."
    touch dummy1.txt dummy2.txt dummy1.c dummy2.c
    echo "Dummy files created: dummy1.txt, dummy2.txt, dummy1.c, dummy2.c"
else
    echo "Existing .txt files:"
    ls *.txt 2>/dev/null
    echo "Existing .c files:"
    ls *.c 2>/dev/null
fi
```

Explanation:

- The script checks for .txt and .c files in the current directory.
- If no such files are found, it creates dummy1.txt, dummy2.txt, dummy1.c, and dummy2.c.
-
- If files already exist, it lists them instead of creating new ones.
- The script needs to be saved, given execute permissions, and then run.

2: Script to Add Two Numbers

```
(base) tetarwal005@hp:~/c_program$ nano add_two_numbers.sh
(base) tetarwal005@hp:~/c_program$ chmod +x add_two_numbers.sh
(base) tetarwal005@hp:~/c_program$ ./add_two_numbers.sh
Enter the first number:
10
Enter the second number:
11
The sum of 10 and 11 is: 21
(base) tetarwal005@hp:~/c_program$
```

Code:

```
echo "Enter the first number:"
read num1
echo "Enter the second number:"
read num2
sum=$((num1 + num2))
echo "The sum of $num1 and $num2 is: $sum"
```

Explanation:

- The script prompts the user to enter two numbers and stores them in variables.
- It calculates the sum using arithmetic expansion and stores the result.
- The script then displays the sum in a formatted message.
- It must be given execute permissions before running using chmod +x.
- When executed, the user inputs numbers, and the script prints their sum.

3: Installing Docker Using a Shell Script

```
(base) tetarwal005@hp:~/c_program$ nano install_docker.sh
(base) tetarwal005@hp:~/c_program$ chmod +x install_docker.sh
(base) tetarwal005@hp:~/c_program$ ./install_docker.sh
Updating package list...
[sudo] password for tetarwal005:
Get:1 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Hit:2 http://in.archive.ubuntu.com/ubuntu noble InRelease
Get:3 http://in.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Hit:4 https://ftp.postgresql.org/pub/pgadmin/pgadmin4/apt/noble pgadmin4 InRelease
Get:5 https://apt.postgresql.org/pub/repos/apt noble-pgdg InRelease [129 kB]
```

```
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /usr/lib/systemd/system/docker.socket.
Processing triggers for man-db (2.12.0-4build2) ...
Checking Docker version...
Docker installed successfully. Version: Docker version 26.1.3, build 26.1.3-0ubuntu1~24.04.1
(base) tetarwal005@hp:~/c_program$
```

Script :

```
echo "Updating package list..."
sudo apt-get update -y
echo "Installing Docker..."
sudo apt-get install docker.io -y
echo "Checking Docker version..."
docker_version=$(docker --version)
echo "Docker installed successfully. Version: $docker_version"
```

Explanation:

- The script first updates the package list using sudo apt-get update -y to ensure the system has the latest package information.
- It then installs Docker using sudo apt-get install docker.io -y, which fetches and installs the Docker package from the repository.
- After installation, the script verifies the Docker installation by checking its version using docker --version.
- The output displays the installed Docker version to confirm successful installation.
- The script uses sudo to execute commands that require administrative privileges, ensuring smooth installation.

4: Download and Install MySQL Database

```
(base) tetarwal005@hp:~/c_program$ nano install_mysql.sh
(base) tetarwal005@hp:~/c_program$ chmod +x install_mysql.sh
(base) tetarwal005@hp:~/c_program$ ./install_mysql.sh
Updating package list...
Hit:1 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:2 https://ftp.postgresql.org/pub/pgadmin/pgadmin4/apt/noble pgadmin4 InRelease
Hit:3 https://apt.postgresql.org/pub/repos/apt noble-pgdg InRelease
Hit:4 http://in.archive.ubuntu.com/ubuntu noble InRelease
Hit:5 http://in.archive.ubuntu.com/ubuntu noble-updates InRelease
```

```
Starting MySQL service...
Enabling MySQL service to start on boot...
Synchronizing state of mysql.service with SysV service script with /usr/lib/systemd/
Executing: /usr/lib/systemd/systemd-sysv-install enable mysql
Checking MySQL service status...
     Active: active (running) since Thu 2025-01-30 22:39:59 IST; 10s ago
(base) tetarwal005@hp:~/c_program$ mysql --version
mysql  Ver 8.0.41-0ubuntu0.24.04.1 for Linux on x86_64 ((Ubuntu))
(base) tetarwal005@hp:~/c_program$
```

Script:

```
echo "Updating package list..."
sudo apt-get update -y
echo "Installing MySQL..."
sudo apt-get install mysql-server -y
echo "Starting MySQL service..."
sudo systemctl start mysql
echo "Enabling MySQL service to start on boot..."
sudo systemctl enable mysql
echo "Checking MySQL service status..."
sudo systemctl status mysql | grep "Active"
```

Explanation:

- The script first updates the package list using sudo apt-get update -y to ensure the latest package versions are installed.
- It installs the MySQL server using sudo apt-get install mysql-server -y, which downloads and sets up MySQL.

- The script starts the MySQL service with sudo systemctl start mysql and enables it to start on boot using sudo systemctl enable mysql.
- To verify the installation, it checks the MySQL service status using sudo systemctl status mysql | grep "Active" and displays whether it is running.
- After execution, the output confirms MySQL installation, service startup, and active status, ensuring successful setup.