Experiment [7] : [Shell programming,process and scheduling]

Name: Prapti Uniyal Roll No.: 590028360 Date: 21-09-2025

Aim:

To automate repetitive system tasks such as backing up files,monitoring system resources, etc.

Requirements

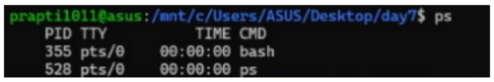
- [Any Linux Distro, any kind of text editor (vs code, vim,nano, etc)].

Theory

Every program running in Linux is a process identified by a unique process ID (PID). Shell programming allows automation of tasks including spawning and controlling processes.

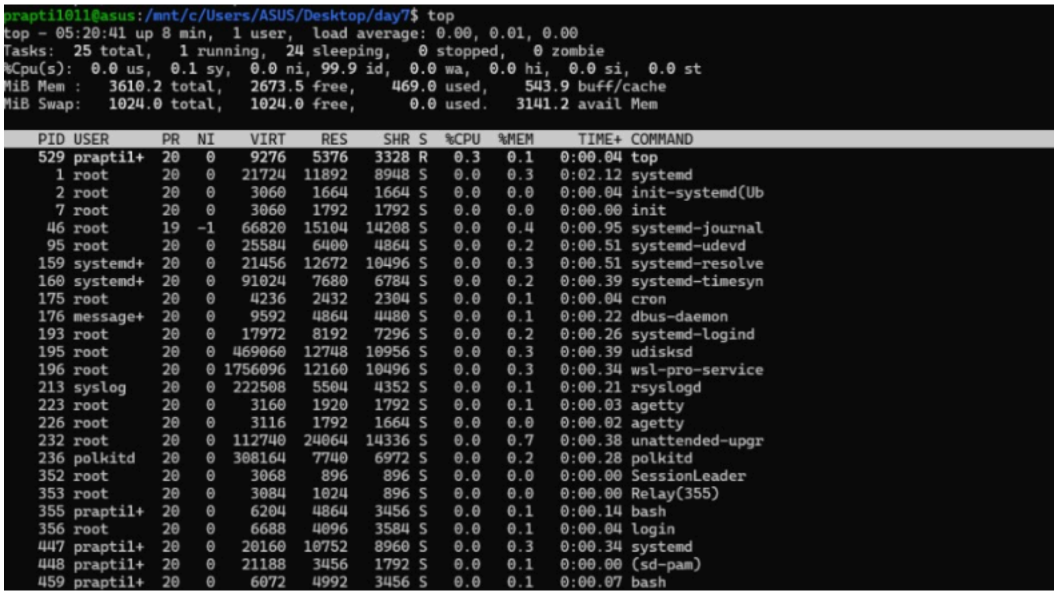
- Commands

1). ps- this command is used to display information about currently running processes. It stands for “process status”.



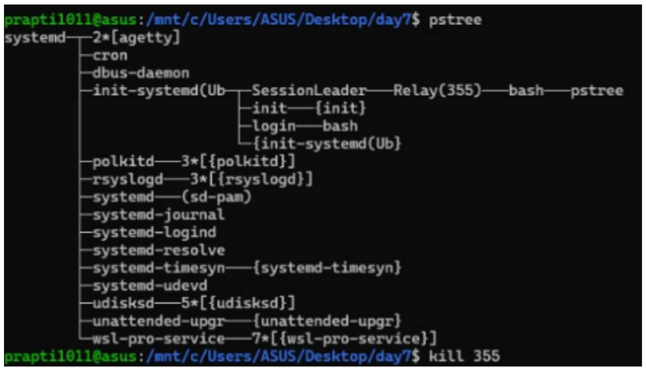
```
prapti1011@asus:/mnt/c/Users/ASUS/Desktop/day7$ ps
    PID TTY          TIME CMD
    355 pts/0    00:00:00 bash
    528 pts/0    00:00:00 ps
```

2). top- it is a powerful utility for real-time system monitoring and data execution processes.



```
prapti1011@asus:/mnt/c/Users/ASUS/Desktop/day7$ top
top - 05:20:41 up 8 min,  1 user,  load average: 0.00, 0.01, 0.00
Tasks:  25 total,   1 running,  24 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.0 us,  0.1 sy,  0.0 ni, 99.9 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :   3610.2 total,   2673.5 free,    469.0 used,    543.9 buff/cache
MiB Swap:   1024.0 total,   1024.0 free,      0.0 used.   3141.2 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
    529 praptil+  20   0    9276   5376   3328 R   0.3   0.1   0:00.04 top
      1 root      20   0   21724  11892   8948 S   0.0   0.3   0:02.12 systemd
      2 root      20   0    3060   1664   1664 S   0.0   0.0   0:00.04 init-systemd(Ub
      7 root      20   0    3060   1792   1792 S   0.0   0.0   0:00.00 init
     46 root      19  -1   66820  15104  14208 S   0.0   0.4   0:00.95 systemd-journal
     95 root      20   0   25584   6400   4864 S   0.0   0.2   0:00.51 systemd-udevd
    159 systemd+  20   0   21456  12672  10496 S   0.0   0.3   0:00.51 systemd-resolve
    160 systemd+  20   0   91024   7680   6784 S   0.0   0.2   0:00.39 systemd-timesyn
    175 root      20   0    4236   2432   2304 S   0.0   0.1   0:00.04 cron
    176 message+  20   0    9592   4864   4480 S   0.0   0.1   0:00.22 dbus-daemon
    193 root      20   0   17972   8192   7296 S   0.0   0.2   0:00.26 systemd-logind
    195 root      20   0  469060  12748  10956 S   0.0   0.3   0:00.39 udisksd
    196 root      20   0 1756096  12160  10496 S   0.0   0.3   0:00.34 wsl-pro-service
    213 syslog    20   0  222508   5504   4352 S   0.0   0.1   0:00.21 rsyslogd
    223 root      20   0    3160   1920   1792 S   0.0   0.1   0:00.03 agetty
    226 root      20   0    3116   1792   1664 S   0.0   0.0   0:00.02 agetty
    232 root      20   0  112740  24064  14336 S   0.0   0.7   0:00.38 unattended-upgr
    236 polkitd   20   0  308164   7740   6972 S   0.0   0.2   0:00.28 polkitd
    352 root      20   0    3068    896    896 S   0.0   0.0   0:00.00 SessionLeader
    353 root      20   0    3084   1024    896 S   0.0   0.0   0:00.00 Relay(355)
    355 praptil+  20   0    6204   4864   3456 S   0.0   0.1   0:00.14 bash
    356 root      20   0    6688   4096   3584 S   0.0   0.1   0:00.04 login
    447 praptil+  20   0   20160  10752   8960 S   0.0   0.3   0:00.34 systemd
    448 praptil+  20   0   21188   3456   1792 S   0.0   0.1   0:00.00 (sd-pam)
    459 praptil+  20   0    6072   4992   3456 S   0.0   0.1   0:00.07 bash
```

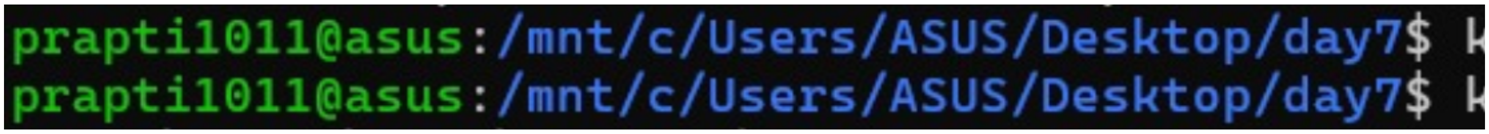
3). pstree- it is used to display a tree like structure of running processes,showing their parent-child relationship.



```
prapti1011@asus:/mnt/c/Users/ASUS/Desktop/day7$ pstree
systemd─┬─2*[agetty]
        ├─cron
        ├─dbus-daemon
        ├─init-systemd(Ub─┬─SessionLeader───Relay(355)───bash───pstree
        │                 ├─init───{init}
        │                 ├─login───bash
        │                 └─{init-systemd(Ub}
        ├─polkitd───3*[{polkitd}]
        ├─rsyslogd───3*[{rsyslogd}]
        ├─systemd───(sd-pam)
        ├─systemd-journal
        ├─systemd-logind
        ├─systemd-resolve
        ├─systemd-timesyn───{systemd-timesyn}
        ├─systemd-udevd
        ├─udisksd───5*[{udisksd}]
        ├─unattended-upgr───{unattended-upgr}
        └─wsl-pro-service───7*[{wsl-pro-service}]
prapti1011@asus:/mnt/c/Users/ASUS/Desktop/day7$ kill 355
```

- Killing process

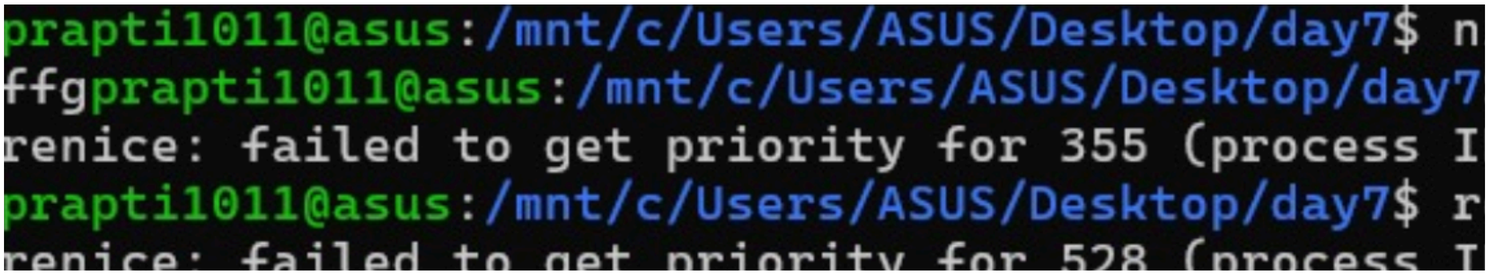
“killing a process” refers to the act of terminating a running program or application.This is typically done when the process becomes unresponsive,consumes excessive resources.



```
prapti1011@asus:/mnt/c/Users/ASUS/Desktop/day7$ k
prapti1011@asus:/mnt/c/Users/ASUS/Desktop/day7$ k
```

- Process Prioritization

It refers to managing the “nice value” of a process,which influences how the linux kernel allocates CPU time among competing processes.

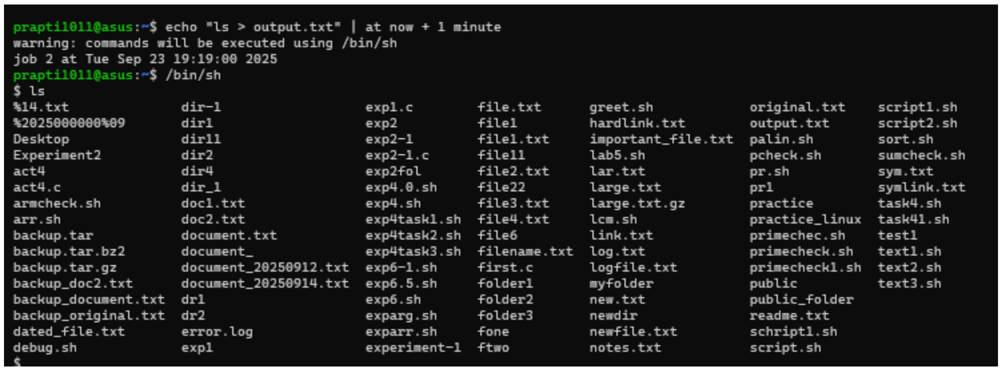


```
prapti1011@asus:/mnt/c/Users/ASUS/Desktop/day7$ n
ffgprapti1011@asus:/mnt/c/Users/ASUS/Desktop/day7
renice: failed to get priority for 355 (process I
prapti1011@asus:/mnt/c/Users/ASUS/Desktop/day7$ r
renice: failed to get priority for 528 (process I
```

- Scheduling processes

It uses the command “at”,this command in linux provides a method for scheduling commands or scripts to be executed once at a specific time in the future.

Syntax- at [OPTIONS] TIME [DATE]



```
prapti1011@asus:~$ echo "ls > output.txt" | at now + 1 minute
warning: commands will be executed using /bin/sh
job 2 at Tue Sep 23 19:19:00 2025
prapti1011@asus:~$ /bin/sh
$ ls
%14.txt              dir-1                  exp1.c        file.txt      greet.sh            original.txt    script1.sh
%2025000000%09       dir1                   exp2          file1         hardlink.txt        output.txt      script2.sh
Desktop              dir11                  exp2-1        file1.txt     important_file.txt  palin.sh        sort.sh
Experiment2          dir2                   exp2-1.c      file11        lab5.sh             pcheck.sh       sumcheck.sh
act4                 dir4                   exp2fol       file2.txt     lar.txt             pr.sh           sym.txt
act4.c               dir_1                  exp4.0.sh     file22        large.txt           pr1             symlink.txt
armcheck.sh          doc1.txt               exp4.sh       file3.txt     large.txt.gz        practice        task4.sh
arr.sh               doc2.txt               exp4task1.sh  file4.txt     lcm.sh              practice_linux  task41.sh
backup.tar           document.txt           exp4task2.sh  file6         link.txt            primechec.sh    test1
backup.tar.bz2       document_              exp4task3.sh  filename.txt  log.txt             primecheck.sh   text1.sh
backup.tar.gz        document_20250912.txt  exp6-1.sh     first.c       logfile.txt         primecheck1.sh  text2.sh
backup_doc2.txt      document_20250914.txt  exp6.5.sh     folder1       myfolder            public          text3.sh
backup_document.txt  dr1                    exp6.sh       folder2       new.txt             public_folder
backup_original.txt  dr2                    exparg.sh     folder3       newdir              readme.txt
dated_file.txt       error.log              exparr.sh     fone          newfile.txt         schript1.sh
debug.sh             exp1                   experiment-1  ftwo          notes.txt           script.sh
$
```

Procedure & Observations

Task [1]: [Check the existence of a file]

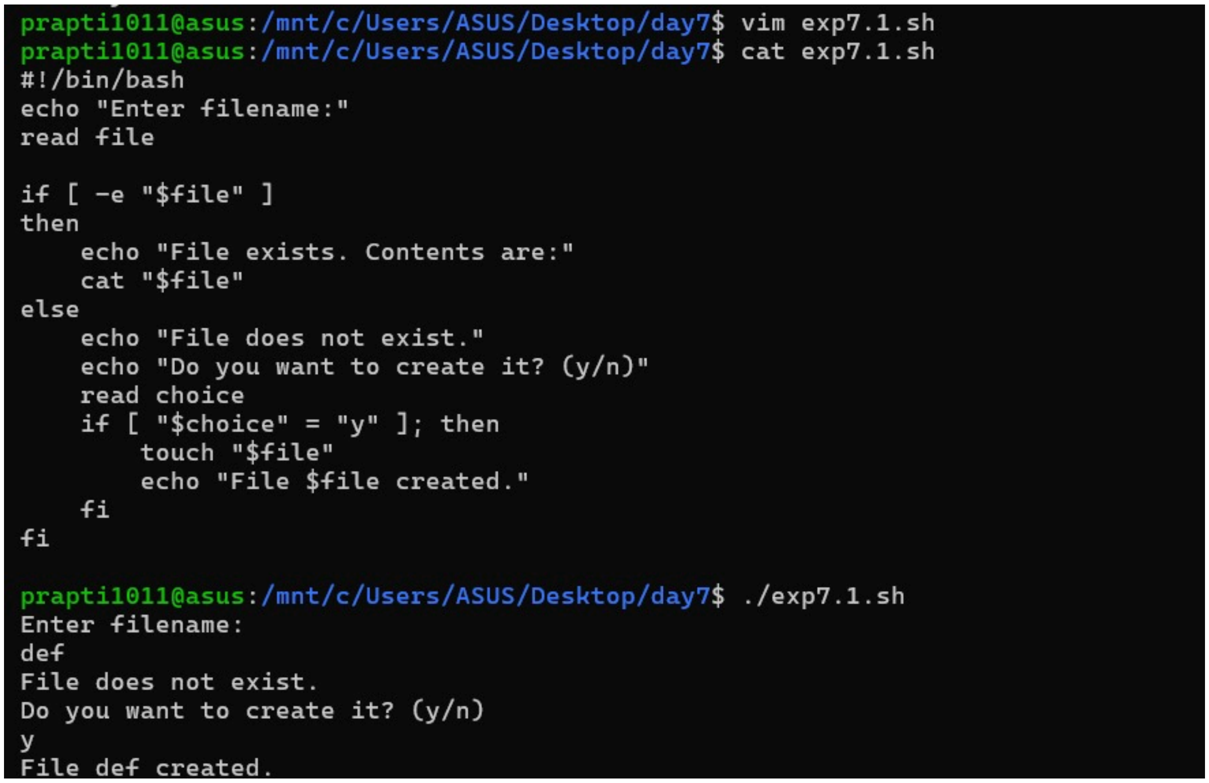
Task Statement:

Wap a shell program to check if the given file exists or not.

Command(s):

```
#!/bin/bash
echo "Enter filename: "
read file

if [ -e "$file" ]
then
    echo "File exists. Contents are:"
    cat "$file"
else
    echo "File does not exist."
    echo "Do you want to create it? (y/n)"
    read choice
    if [ "$choice" = "y" ]; then
        touch "$file"
        echo "File $file created."
    fi
fi
```

Output:



```
prapti1011@asus:/mnt/c/Users/ASUS/Desktop/day7$ vim exp7.1.sh
prapti1011@asus:/mnt/c/Users/ASUS/Desktop/day7$ cat exp7.1.sh
#!/bin/bash
echo "Enter filename:"
read file

if [ -e "$file" ]
then
    echo "File exists. Contents are:"
    cat "$file"
else
    echo "File does not exist."
    echo "Do you want to create it? (y/n)"
    read choice
    if [ "$choice" = "y" ]; then
        touch "$file"
        echo "File $file created."
    fi
fi

prapti1011@asus:/mnt/c/Users/ASUS/Desktop/day7$ ./exp7.1.sh
Enter filename:
def
File does not exist.
Do you want to create it? (y/n)
y
File def created.
```

Task [2]: [Print numbers from 1 to 10]

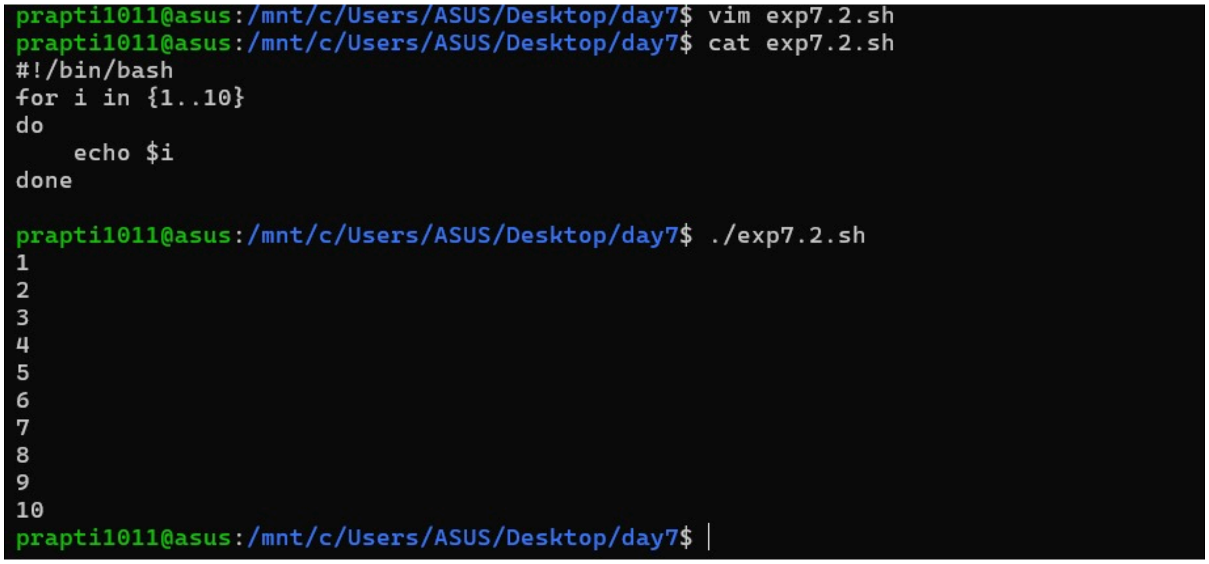
Task Statement:

Wap a shell program to print numbers from 1 to 10.

Command(s):

```
#!/bin/bash
for i in {1..10}
do
    echo $i
done
```

Output:



```
prapti1011@asus:/mnt/c/Users/ASUS/Desktop/day7$ vim exp7.2.sh
prapti1011@asus:/mnt/c/Users/ASUS/Desktop/day7$ cat exp7.2.sh
#!/bin/bash
for i in {1..10}
do
    echo $i
done

prapti1011@asus:/mnt/c/Users/ASUS/Desktop/day7$ ./exp7.2.sh
1
2
3
4
5
6
7
8
9
10
prapti1011@asus:/mnt/c/Users/ASUS/Desktop/day7$ |
```

Task [3]: [Count lines,words and characters]

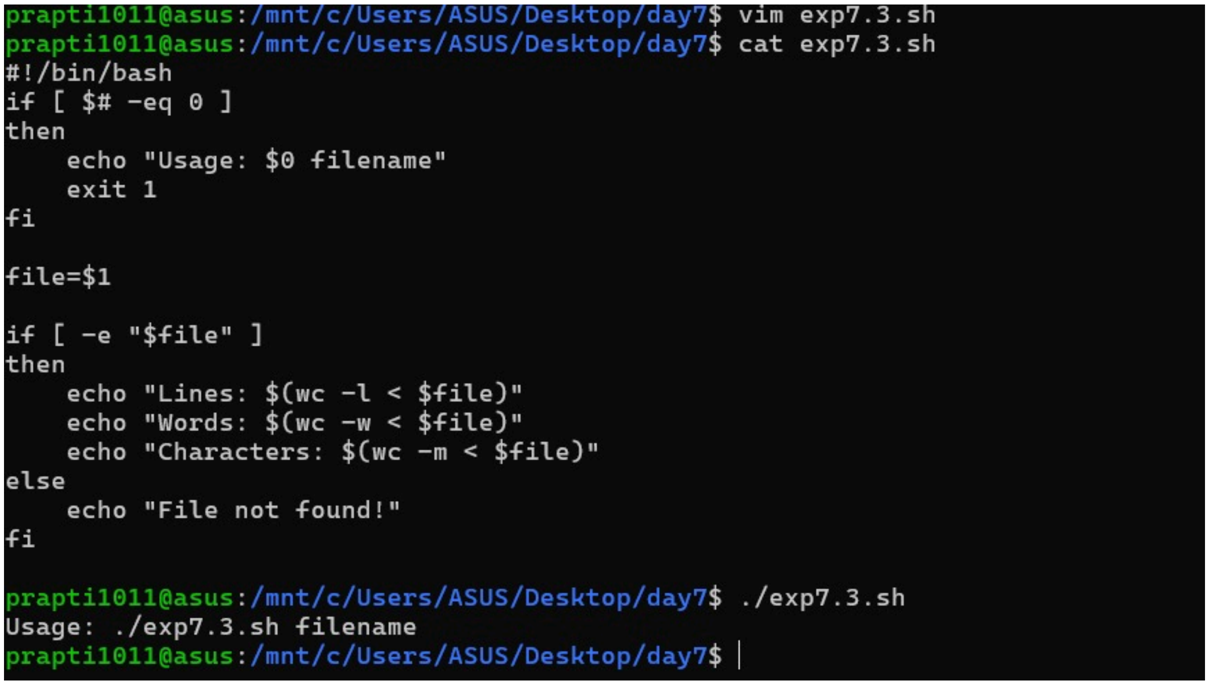
Task Statement:

Wap a shell program to count lines,words and characters.

Command(s):

```
#!/bin/bash
if [ $# -eq 0 ]
then
    echo "Usage: $0 filename"
    exit 1
fi

file=$1

if [ -e "$file" ]
then
    echo "Lines: $(wc -l < $file)"
    echo "Words: $(wc -w < $file)"
    echo "Characters: $(wc -m < $file)"
else
    echo "File not found!"
fi
```

Output:



```
prapti1011@asus:/mnt/c/Users/ASUS/Desktop/day7$ vim exp7.3.sh
prapti1011@asus:/mnt/c/Users/ASUS/Desktop/day7$ cat exp7.3.sh
#!/bin/bash
if [ $# -eq 0 ]
then
    echo "Usage: $0 filename"
    exit 1
fi

file=$1

if [ -e "$file" ]
then
    echo "Lines: $(wc -l < $file)"
    echo "Words: $(wc -w < $file)"
    echo "Characters: $(wc -m < $file)"
else
    echo "File not found!"
fi

prapti1011@asus:/mnt/c/Users/ASUS/Desktop/day7$ ./exp7.3.sh
Usage: ./exp7.3.sh filename
prapti1011@asus:/mnt/c/Users/ASUS/Desktop/day7$ |
```

Task [4]: [Find factorial using function]

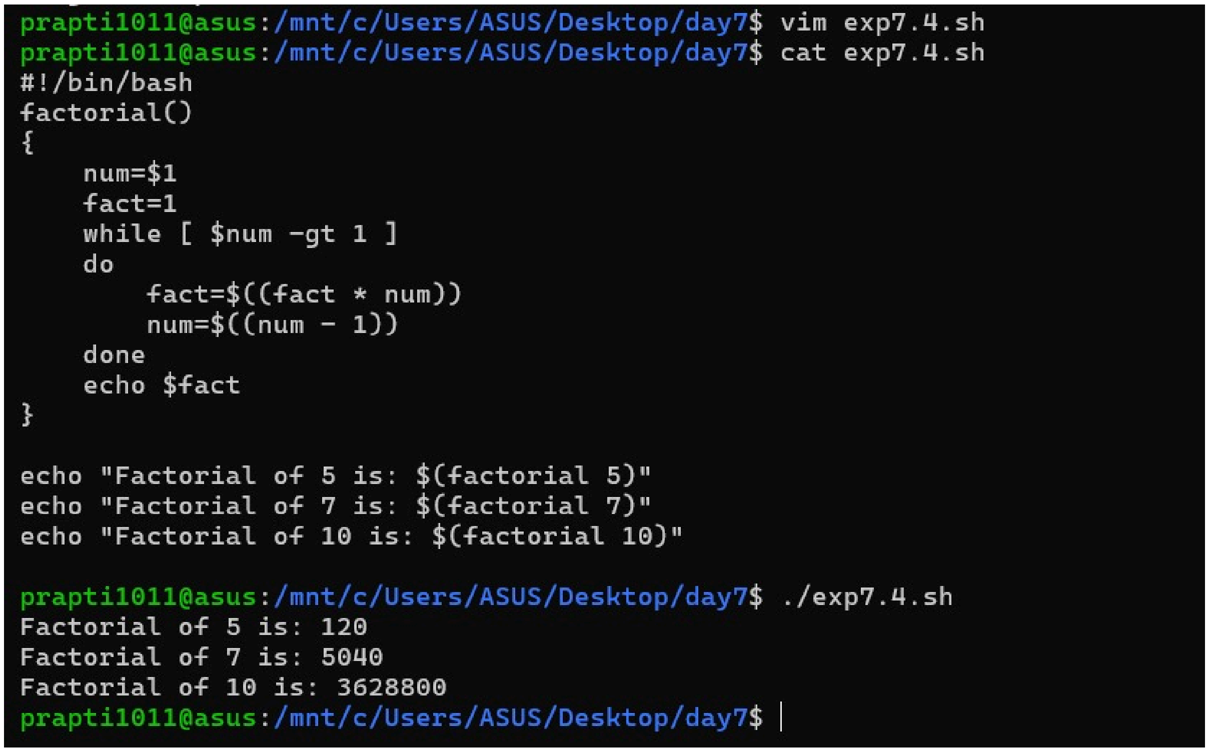
Task Statement:

Wap a shell program to find factorial of a number.

Command(s):

```
#!/bin/bash
factorial() {
    num=$1
    fact=1
    while [ $num -gt 1 ]
    do
        fact=$((fact * num))
        num=$((num - 1))
    done
    echo $fact
}

echo "Factorial of 5 is: $(factorial 5)"
echo "Factorial of 7 is: $(factorial 7)"
echo "Factorial of 10 is: $(factorial 10)"
```

Output:



```
prapti1011@asus:/mnt/c/Users/ASUS/Desktop/day7$ vim exp7.4.sh
prapti1011@asus:/mnt/c/Users/ASUS/Desktop/day7$ cat exp7.4.sh
#!/bin/bash
factorial()
{
    num=$1
    fact=1
    while [ $num -gt 1 ]
    do
        fact=$((fact * num))
        num=$((num - 1))
    done
    echo $fact
}

echo "Factorial of 5 is: $(factorial 5)"
echo "Factorial of 7 is: $(factorial 7)"
echo "Factorial of 10 is: $(factorial 10)"

prapti1011@asus:/mnt/c/Users/ASUS/Desktop/day7$ ./exp7.4.sh
Factorial of 5 is: 120
Factorial of 7 is: 5040
Factorial of 10 is: 3628800
prapti1011@asus:/mnt/c/Users/ASUS/Desktop/day7$
```

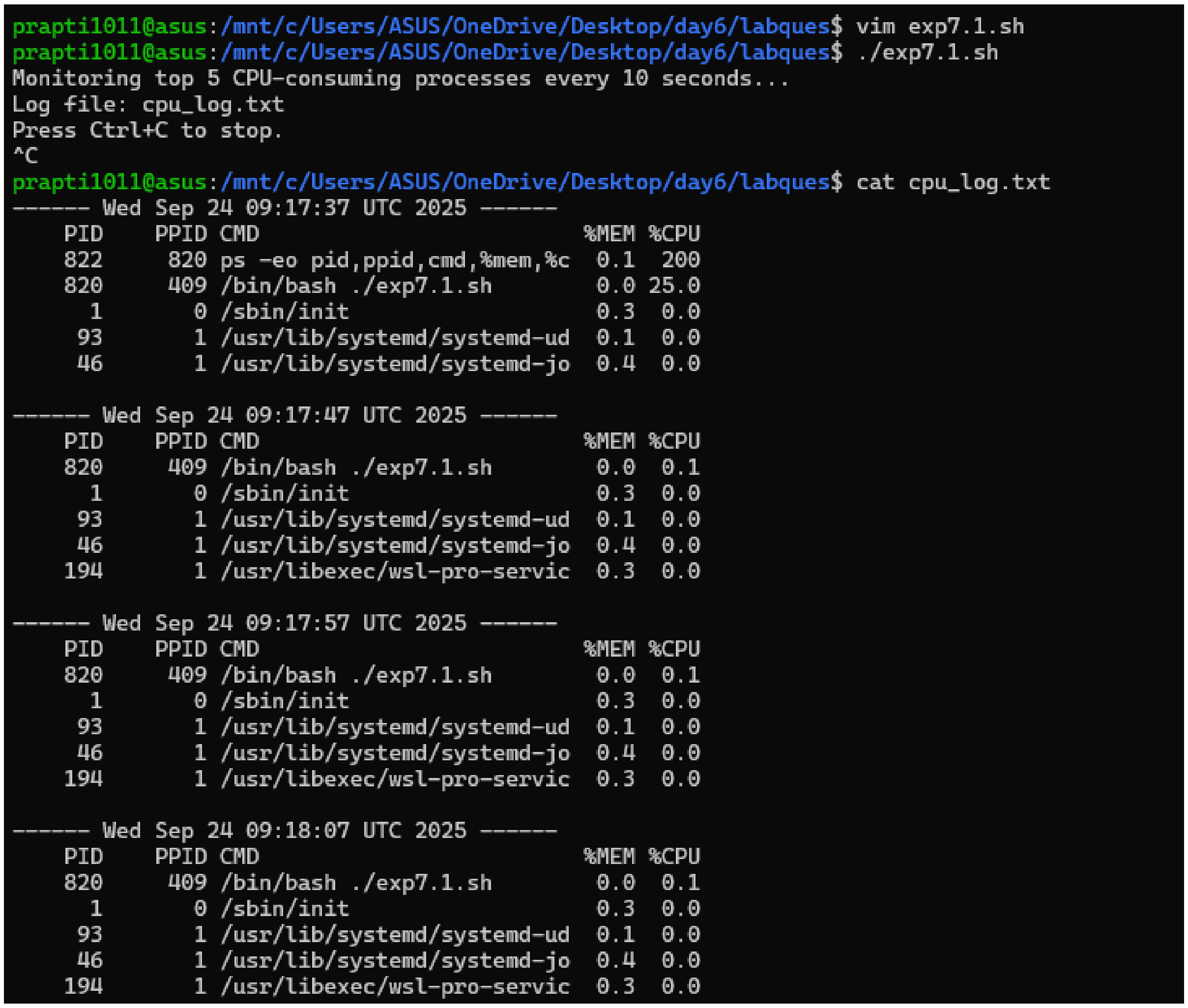
ASSIGNMENT

EXERCISE 1 : [Write a script that monitors the top5 processes consuming the most CPU and logs them into a file every 10 seconds]

Command(s):

```
#!/bin/bash

# Log file name
logfile="cpu_log.txt"

echo "Monitoring top 5 CPU-consuming processes every 10 seconds..."
echo "Log file: $logfile"
echo "Press Ctrl+C to stop."

while true
do
    echo "------ $(date) ------" >> "$logfile"
    ps -eo pid,ppid,cmd,%mem,%cpu --sort=-%cpu | head -n 6 >> "$logfile"
    echo "" >> "$logfile"
    sleep 10
done
```

Output:



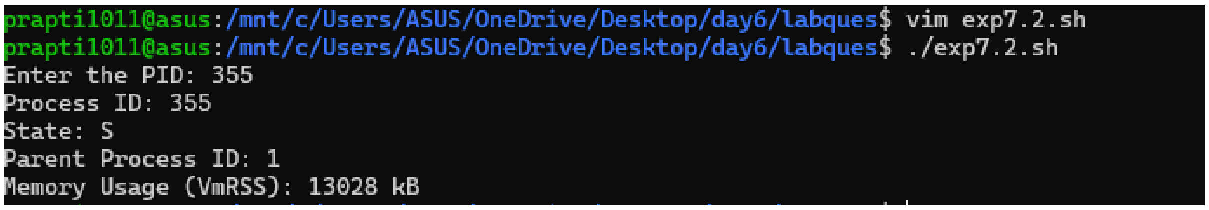
```
prapti1011@asus:/mnt/c/Users/ASUS/OneDrive/Desktop/day6/labques$ vim exp7.1.sh
prapti1011@asus:/mnt/c/Users/ASUS/OneDrive/Desktop/day6/labques$ ./exp7.1.sh
Monitoring top 5 CPU-consuming processes every 10 seconds...
Log file: cpu_log.txt
Press Ctrl+C to stop.
^C
prapti1011@asus:/mnt/c/Users/ASUS/OneDrive/Desktop/day6/labques$ cat cpu_log.txt
------ Wed Sep 24 09:17:37 UTC 2025 ------
    PID    PPID CMD                         %MEM %CPU
    822     820 ps -eo pid,ppid,cmd,%mem,%c  0.1  200
    820     409 /bin/bash ./exp7.1.sh        0.0 25.0
      1       0 /sbin/init                   0.3  0.0
     93       1 /usr/lib/systemd/systemd-ud  0.1  0.0
     46       1 /usr/lib/systemd/systemd-jo  0.4  0.0

------ Wed Sep 24 09:17:47 UTC 2025 ------
    PID    PPID CMD                         %MEM %CPU
    820     409 /bin/bash ./exp7.1.sh        0.0  0.1
      1       0 /sbin/init                   0.3  0.0
     93       1 /usr/lib/systemd/systemd-ud  0.1  0.0
     46       1 /usr/lib/systemd/systemd-jo  0.4  0.0
    194       1 /usr/libexec/wsl-pro-servic  0.3  0.0

------ Wed Sep 24 09:17:57 UTC 2025 ------
    PID    PPID CMD                         %MEM %CPU
    820     409 /bin/bash ./exp7.1.sh        0.0  0.1
      1       0 /sbin/init                   0.3  0.0
     93       1 /usr/lib/systemd/systemd-ud  0.1  0.0
     46       1 /usr/lib/systemd/systemd-jo  0.4  0.0
    194       1 /usr/libexec/wsl-pro-servic  0.3  0.0

------ Wed Sep 24 09:18:07 UTC 2025 ------
    PID    PPID CMD                         %MEM %CPU
    820     409 /bin/bash ./exp7.1.sh        0.0  0.1
      1       0 /sbin/init                   0.3  0.0
     93       1 /usr/lib/systemd/systemd-ud  0.1  0.0
     46       1 /usr/lib/systemd/systemd-jo  0.4  0.0
    194       1 /usr/libexec/wsl-pro-servic  0.3  0.0
```

EXERCISE 2 : Write a script that accepts a PID from the user and displays its details(state,parent process,memory usage)

Command(s):

```
#!/bin/bash

read -p "Enter the PID: " pid

if [ ! -d "/proc/$pid" ]; then
    echo "Process with PID $pid does not exist."
    exit 1
fi

state=$(grep -i "State" /proc/$pid/status | awk '{print $2}')
ppid=$(grep -i "PPid" /proc/$pid/status | awk '{print $2}')
vmrss=$(grep -i "VmRSS" /proc/$pid/status | awk '{print $2, $3}')

echo "Process ID: $pid"
echo "State: $state"
echo "Parent Process ID: $ppid"
echo "Memory Usage (VmRSS): $vmrss"
```

Output:



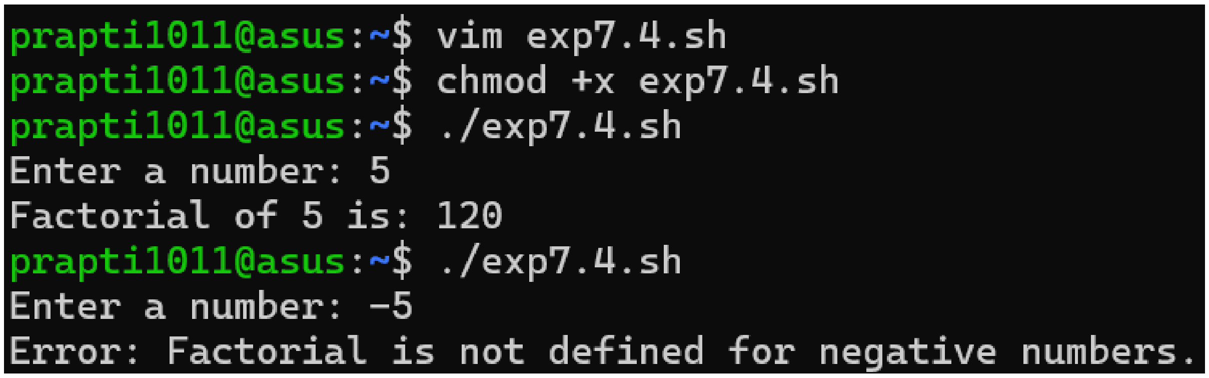
```
prapti1011@asus:/mnt/c/Users/ASUS/OneDrive/Desktop/day6/labques$ vim exp7.2.sh
prapti1011@asus:/mnt/c/Users/ASUS/OneDrive/Desktop/day6/labques$ ./exp7.2.sh
Enter the PID: 355
Process ID: 355
State: S
Parent Process ID: 1
Memory Usage (VmRSS): 13028 kB
```

EXERCISE 3 : Modify the factorial function to check if input is negative,if yes, display an error message.

Command(s):

```
#!/bin/bash

read -p "Enter a number: " num

if [ "$num" -lt 0 ]; then
    echo "Error: Factorial is not defined for negative numbers."
    exit 1
fi

fact=1
for (( i=1; i<=num; i++ ))
do
    fact=$((fact * i))
done

echo "Factorial of $num is: $fact"
```

Output:



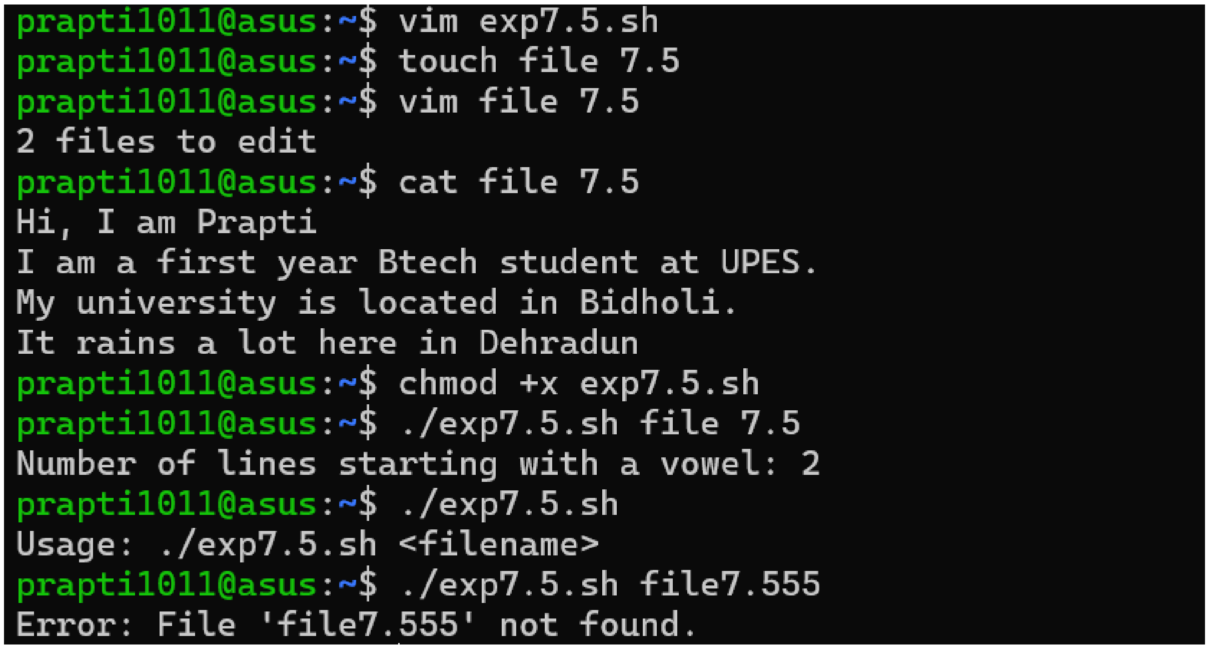
```
prapti1011@asus:~$ vim exp7.4.sh
prapti1011@asus:~$ chmod +x exp7.4.sh
prapti1011@asus:~$ ./exp7.4.sh
Enter a number: 5
Factorial of 5 is: 120
prapti1011@asus:~$ ./exp7.4.sh
Enter a number: -5
Error: Factorial is not defined for negative numbers.
```

EXERCISE 4 : Write e a script that accepts a filename as an argument.If the file exists,display the number of lines starting with a vowel

Command(s):

```
#!/bin/bash

if [ -z "$1" ]; then
    echo "Usage: $0 <filename>"
    exit 1
fi

filename="$1"

if [ ! -f "$filename" ]; then
    echo "Error: File '$filename' not found."
    exit 1
fi

count=$(grep -iE '^[aeiou]' "$filename" | wc -l)

echo "Number of lines starting with a vowel: $count"
```

Output:



```
prapti1011@asus:~$ vim exp7.5.sh
prapti1011@asus:~$ touch file 7.5
prapti1011@asus:~$ vim file 7.5
2 files to edit
prapti1011@asus:~$ cat file 7.5
Hi, I am Prapti
I am a first year Btech student at UPES.
My university is located in Bidholi.
It rains a lot here in Dehradun
prapti1011@asus:~$ chmod +x exp7.5.sh
prapti1011@asus:~$ ./exp7.5.sh file 7.5
Number of lines starting with a vowel: 2
prapti1011@asus:~$ ./exp7.5.sh
Usage: ./exp7.5.sh <filename>
prapti1011@asus:~$ ./exp7.5.sh file7.555
Error: File 'file7.555' not found.
```

Challenges faced:

Remembering the `crontab` time format. Solved by using online crontab generators and practice.

Laerning:

- Gained hands-on knowledge of process creation and termination.

Result

All the exercises and tasks were completed successfully and these tasks helped a lot in learning about killing process, scheduling, etc.