K. J. Somaiya College of Engineering, Mumbai-77
(A Constituent College of Somaiya Vidyavihar University)
Department of Computer Engineering

Batch:C2 Roll No.: 16010121110
Experiment No. 02
Grade: AA / AB / BB / BC / CC / CD /DD
Signature of the Staff In-charge with date

TITLE: Shell Programming and system calls

AIM: To study the shell script and write the program using shell.

Expected Outcome of Experiment:

CO 1. To introduce basic concepts and functions of operating systems.

Books/ Journals/ Websites referred:

1. **Silberschatz A., Galvin P., Gagne G. "Operating Systems Principles", Willey Eight edition.**
2. **William Stallings "Operating Systems" Person, Seventh Edition Edition.**
3. **Sumitabha Das " UNIX Concepts & Applications", McGraw Hill Second Edition.**

Pre Lab/ Prior Concepts:

The shell provides you with an interface to the UNIX system. It gathers input from you and executes programs based on that input. When a program finishes executing, it displays that program's output.

Shell Scripts
The basic concept of a shell script is a list of commands, which are listed in the order of execution. A good shell script will have comments, preceded by a pound sign, #, describing the steps.

K. J. Somaiya College of Engineering, Mumbai-77
(A Constituent College of Somaiya Vidyavihar University)
Department of Computer Engineering

Steps to create a Shell Script:
create a file using any text editor say vi, gedit, nano etc
1.$ vi filename

2.Insert the script/ commands in file and save the file to execute the file we need to give execute permission to the file
3.$ chmod 775 filename
4.Now execute the above file using any of following methods:
$ sh filename
OR
$ ./filename

NOTE: Before adding anything to your script, you need to alert the system that a shell script is being started. This is done using the shebang construct. For example −
#!/bin/sh.

Description of the application to be implemented:

1. Write a shell Script that accepts two file names as command line arguments and compare two file contents and check whether contents are same or not. If they are same, then delete second file.

2. Write a shell script that accepts integer and find the factorial of number.

3. Write a shell script for adding users.

4. Write a shell script for counting no of logged in users.

5. Write a shell script for counting no of processes running on system

Program for System Call:

1. Write a Program for creating process using System call (E.g fork()) Create a child process. Display the details about that process using getpid and getppid functions. In a child process, Open the file using file system calls and read the contents and display.

Implementation details: (printout of code / screen shot)

```
#!/bin/bash
```

```
echo "hello world"
```

kjsce@ubuntu:~$ which bash

/usr/bin/bash

kjsce@ubuntu:~$ cd Desktop

kjsce@ubuntu:**~/Desktop**$ touch exp2.sh

kjsce@ubuntu:**~/Desktop**$ ls

exp1 exp2.sh F2.txt **folder** **Smit-2** **Smit_OS** **Smit_OS_Lab**

kjsce@ubuntu:**~/Desktop**$ ls -al

total 32

drwxr-xr-x 6 kjsce kjsce 4096 Jul 24 16:23 **.**

drwxr-xr-x 17 kjsce kjsce 4096 May 24 20:04 **..**

-rw-rw-r-- 1 kjsce kjsce   67 Jul 17 16:52 exp1

-rw-rw-r-- 1 kjsce kjsce   0 Jul 24 16:23 exp2.sh

-rw-rw-r-- 1 kjsce kjsce   13 Jul 18 15:50 F2.txt

drwxrwxr-x 2 kjsce kjsce 4096 Jul 17 17:36 **folder**

drwxrwxr-x 2 kjsce kjsce 4096 Jul 18 14:45 **Smit-2**

drwxrwxr-x 2 kjsce kjsce 4096 Jul 18 15:16 **Smit_OS**

drwxrwxr-x 2 kjsce kjsce 4096 Jul 18 15:51 **Smit_OS_Lab**

kjsce@ubuntu:**~/Desktop**$ ./exp2.sh

bash: ./exp2.sh: Permission denied

kjsce@ubuntu:**~/Desktop**$ chmod +x exp2.sh

kjsce@ubuntu:**~/Desktop**$ ./exp2.sh

hello world

```
#!/bin/bash


# Check if two file names are provided as
arguments

if [ $# -ne 2 ]; then

echo "Usage: $0 <file1> <file2>"

exit 1

fi


file1="$1"

file2="$2"


# Check if both files exist

if [ ! -f "$file1" ]; then

echo "Error: File '$file1' does not exist."

exit 1

fi
```

```
if [ ! -f "$file2" ]; then

echo "Error: File '$file2' does not exist."

exit 1

fi



# Compare the contents of the files

if cmp -s "$file1" "$file2"; then

echo "The contents of '$file1' and '$file2' are
the same. Deleting '$file2'..."

rm "$file2"

else

echo "The contents of '$file1' and '$file2' are
different."

fi
```

kjsce@ubuntu:**~/Desktop**$ touch a

kjsce@ubuntu:**~/Desktop**$ touch b

kjsce@ubuntu:**~/Desktop**$ ./exp2.sh

Usage: ./exp2.sh <file1> <file2>

kjsce@ubuntu:**~/Desktop**$ ./exp2.sh a c

Error: File 'c' does not exist.

kjsce@ubuntu:**~/Desktop**$ ./exp2.sh a b

The contents of 'a' and 'b' are the same. Deleting 'b'...

kjsce@ubuntu:**~/Desktop**$

```
#!/bin/bash


# Check if an integer is provided as a command-line
argument
if [ $# -ne 1 ]; then
echo "Usage: $0 <integer>"
exit 1
fi


# Function to calculate factorial
calculate_factorial() {
local num=$1
local factorial=1


for (( i=1; i<=num; i++ )); do
factorial=$(( factorial * i ))
```

```
done



echo $factorial

}



# Check if the argument is a valid integer

if [[ "$1" =~ ^[0-9]+$ ]]; then

number=$1

result=$(calculate_factorial $number)

echo "Factorial of $number is $result."

else

echo "Error: Please provide a valid integer as the
argument."

fi
```

kjsce@ubuntu:~/Desktop$ ./exp2.sh 5

Factorial of 5 is 120.

kjsce@ubuntu:~/Desktop$ ./exp2.sh 50

Factorial of 50 is -3258495067890909184.

```
#!/bin/bash

for (( i=1; i<=2; i++ )); do
```

```
echo "please enter name:"

read name

adduser $name

passwd $name

done
```

```
kjsce@ubuntu:~/Desktop$ sudo ./exp2.sh
[sudo] password for kjsce:
please enter name:
jerry
Adding user `jerry' ...
Adding new group `jerry' (1005) ...
Adding new user `jerry' (1004) with group `jerry' ...
Creating home directory `/home/jerry' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for jerry
Enter the new value, or press ENTER for the default
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n] Y
New password:
Retype new password:
passwd: password updated successfully
please enter name:
jimmy
Adding user `jimmy' ...
Adding new group `jimmy' (1006) ...
Adding new user `jimmy' (1005) with group `jimmy' ...
Creating home directory `/home/jimmy' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for jimmy
Enter the new value, or press ENTER for the default
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n] Y
New password:
```

```
systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin/nologin
fwupd-refresh:x:127:134:fwupd-refresh user,,,:/run/systemd:/usr/sbin/nologin
aatmaj:x:1001:1002::/home/aatmaj:/bin/sh
abc:x:1002:1003:abc,1,1,:/home/abc:/bin/bash
xyz:x:1003:1004:,,,:/home/xyz:/bin/bash
jerry:x:1004:1005:,,,:/home/jerry:/bin/bash
jimmy:x:1005:1006:,,,:/home/jimmy:/bin/bash
```

```
#!/bin/bash
processcount=$(ps -e --no-header|wc -l)
echo "($processcount)"
```

kjsce@ubuntu:~/Desktop$ ./exp2.sh
(232)

Conclusion:

Thus we have understood how shell code works in Linux systems. Shell commands are commands that can be executed from the terminal or a file. They do not need any programming language as they are built in within Linux. We learnt the shell language basics. This will enable us to do various tks like interact with servers and write viruses. We understood how code works for system-specific tasks like finding out the number of processes or general tasks like finding the factorial of a number.

Post Lab Descriptive Questions

1. What are the different types of commonly used shells on a typical linux system?
   - Bourne Shell (sh) The Bourne shell was the first default shell on Unix systems, released in 1979.
   - C Shell (csh)
   - TENEX C Shell (tcsh)
   - KornShell (ksh)
   - Debian Almquist Shell (dash)
   - Bourne Again Shell (bash)
   - Z Shell (zsh)
   - Friendly Interactive Shell (fish)

2. How do you find out what's your shell?

```
printf "My current shell - %s\n" "$SHELL"
```

3. List the advantages and disadvantages of shell scripting.

Advantages of shell scripting:

- Automation of repetitive tasks.
- Quick and easy prototyping of commands and processes.
- Seamless integration with system utilities and pipelines.

Disadvantages of shell scripting:

- Limited performance for complex computations.
- Lack of strong data structures and object-oriented features.
- Platform-dependent syntax variations across different shells.

Date: 5 Aug 2023 **Signature of faculty in-charge**