**Experiment No.: 2**

1. A motherboard is the main circuit board inside a computer that connects all of the computer's hardware components together, such as the central processing unit (CPU), memory, storage drives, and expansion cards. It serves as a communication hub between these components, allowing them to work together and perform various functions. The motherboard typically includes several important components, including the chipset and the BIOS (Basic Input/Output System). Modern motherboards come in different form factors, such as ATX, microATX, and mini-ITX, which determine their size and layout. Overall, the motherboard is a critical component that plays a central role in the operation of a computer.

2. RAM (Random Access Memory) modules are computer components that store data and instructions temporarily while the computer is running. RAM is a type of volatile memory, which means that its contents are erased when the computer is turned off or restarted. RAM modules come in various types, speeds, and capacities. They are typically installed in slots on the motherboard and can be easily upgraded or replaced. RAM modules are also used to support multitasking, where multiple programs can run simultaneously. RAM modules are an essential component of a computer that provides temporary storage for data and instructions. They help to improve the computer's performance and support multitasking.

3. A daughter card, also known as a daughterboard or expansion card, is a circuit board that connects to the main motherboard of a computer to add new functionality or enhance existing features. Daughter cards are commonly used to expand the capabilities of a computer, such as adding additional ports, memory, or processing power. Examples of daughter cards include graphics cards, sound cards, network interface cards (NICs), and storage expansion cards.

4. A bus slot, also known as an expansion slot, is a socket on the motherboard of a computer that allows expansion cards to be inserted and connected to the computer's bus system. There are several types of bus slots commonly used in computers, including Peripheral Component Interconnect (PCI), PCI Express (PCIe), and Accelerated Graphics Port (AGP) slots. These slots vary in their bandwidth, power, and physical size, and are designed to accommodate different types of expansion cards. Expansion cards, such as graphics cards, sound cards, and network interface cards, are connect to the computer's bus system through the bus slot, allowing them to communicate with other components and exchange data.

5. SMPS stands for Switched-Mode Power Supply, and it is a type of power supply used in computers and other electronic devices. The SMPS is responsible for converting AC power from a wall outlet into DC power that the computer can use to operate. SMPS uses highfrequency switching and regulation to convert AC power to DC power more efficiently. SMPS units are widely used in modern computers, as they are more reliable, energyefficient, and generate less heat than linear power supplies. SMPS is an essential component of modern computers that plays a vital role in powering the system and ensuring its proper operation.

6. Internal storage devices are electronic components used to store data within a computer or other electronic devices. There are two main types of internal storage devices: Hard Disk Drives (HDDs) and Solid State Drives (SSDs). HDDs are the traditional type of internal storage device and are typically larger in capacity than SSDs. They consist of spinning disks that store data magnetically and read/write heads that move over the disks to access the data. SSDs, on the other hand, use flash memory to store data and have no moving parts, which makes them faster, more durable, and more energy-efficient than HDDs. They are typically more expensive than HDD.

7. Interfacing ports refers to the process of connecting different devices or components together in order to exchange information or perform certain tasks. Ports are used to facilitate communication between devices, and they can come in different forms, such as USB ports, Ethernet ports, HDMI ports, and so on. Interfacing ports is an important aspect of modern computing, as it allows us to connect a wide range of devices and components together in order to create complex systems and networks. Whether it's connecting a keyboard and mouse to a computer, or setting up a network of servers and workstations in a large enterprise, interfacing ports is a crucial part of the process.

**Experiment No.: 3**

**Aim: Familarization of linux commands.**

**CO2: Perform System Administration Tasks.**

**Procedure**

1. Pwd-Print working directory

$ pwd

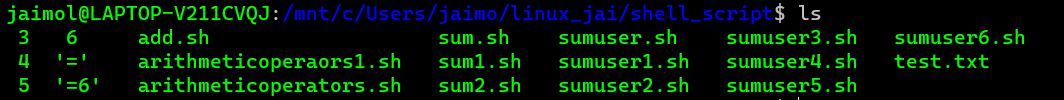
**Output Screenshot**



1. ls- to display a list of contents of a directory.

$ ls

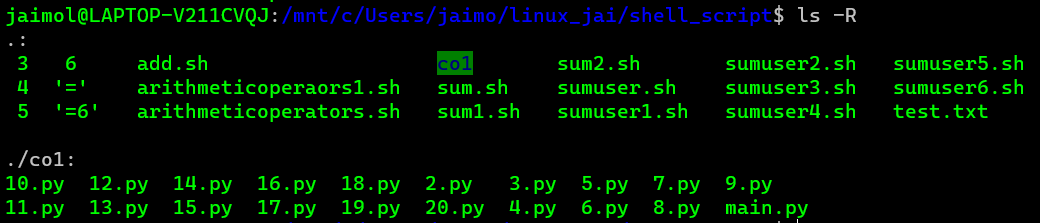
**Output Screenshot**



2.1 ls -R -show contents in the subdirectory.

$ls -R

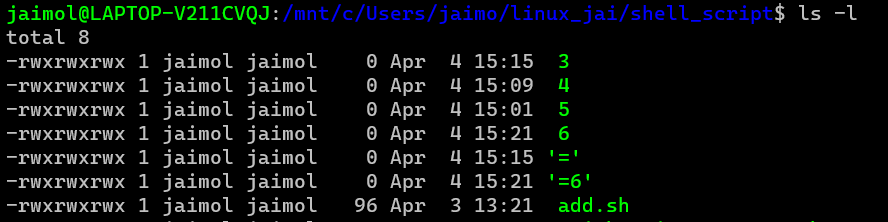
**Output Screenshot**



2.2 ls -l - to longlist a list.

$ls -l

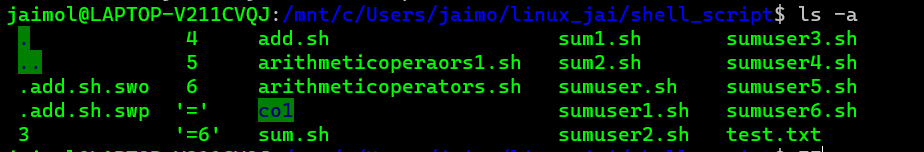
**Output Screenshot**



2.3 ls-a – to list all hidden files.

$ls -a

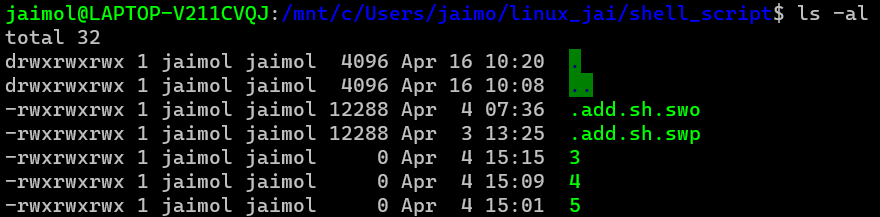
**Output Screenshot**



2.4 ls -al – to list the files and directories with detailed information.

$ls -al

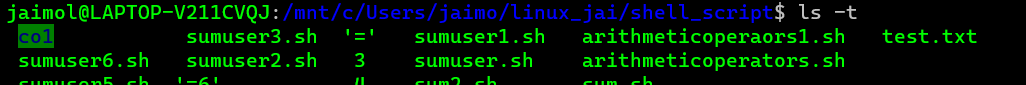
**Output Screenshot**



2.5 ls -t – to list the files as in last modified order.

$ls -t

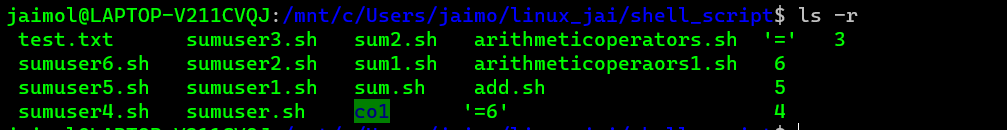
**Output Screenshot**



2.6 ls -r - to remove the natural sorting order.

$ls -r

**Output Screenshot**

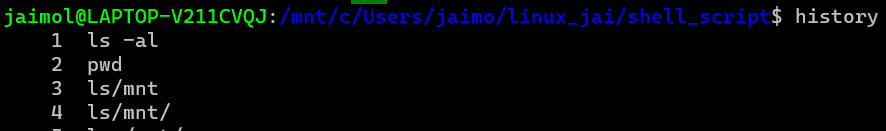


3. history- to review the commands that has been previously executed for certain period of

Time.

$history

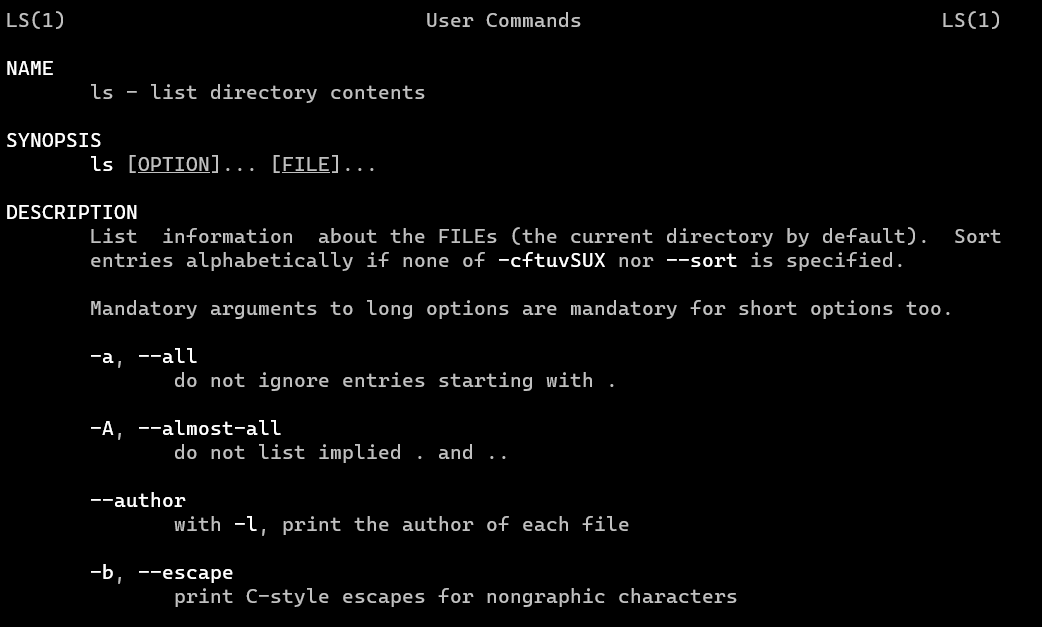
**Output Screenshot**



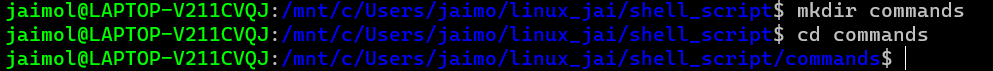
1. man – list options of various commands right from the shell.

$man ls

**Output Screenshot**



5. mkdir – make a new directory

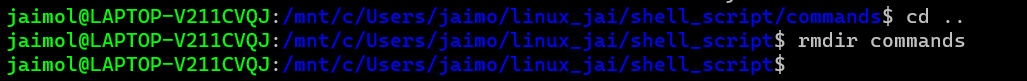


6. cd – change directory

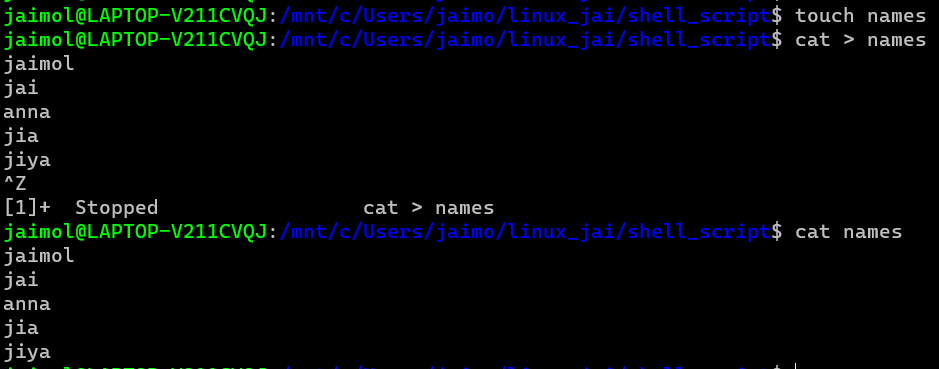
7.cd.. – move to previous directory.

8.rmdir- remove an existing directory

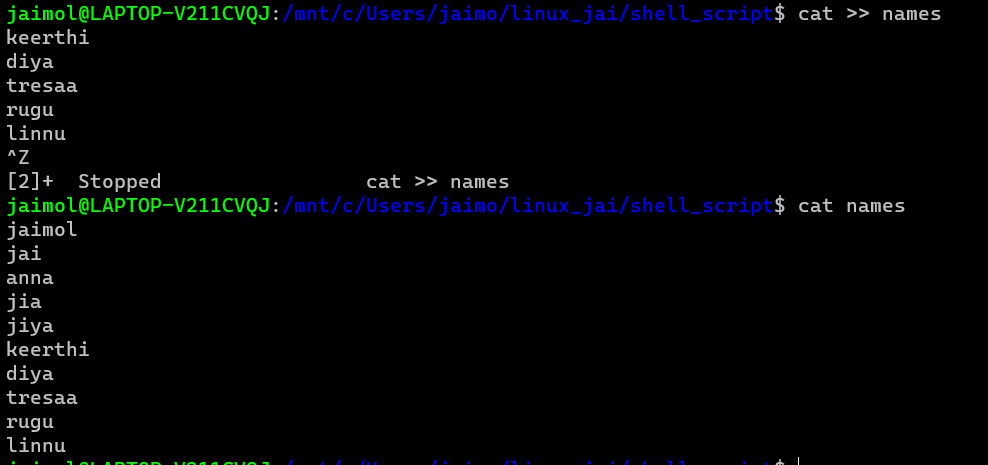
$rmdir



9.touch- create a new file

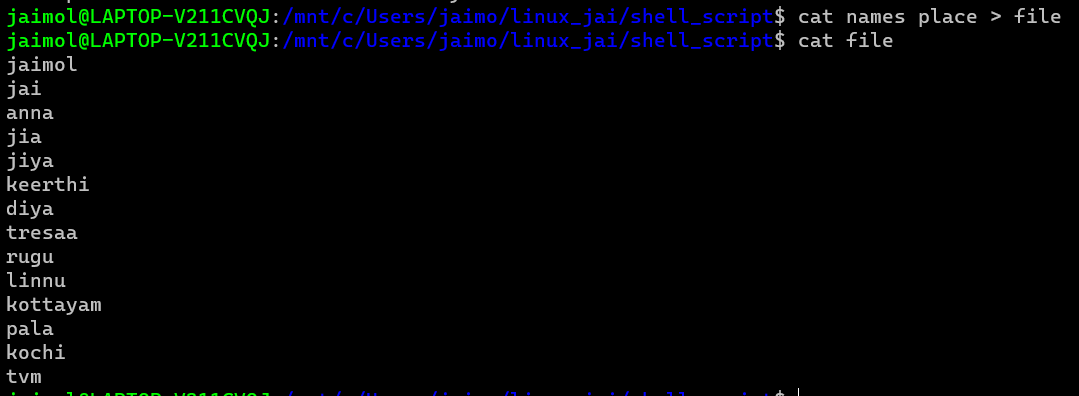


10. cat > filename: create a new file and append the content.



12.cat filename filename > filename – combine the contents of the two file and store it in another file.

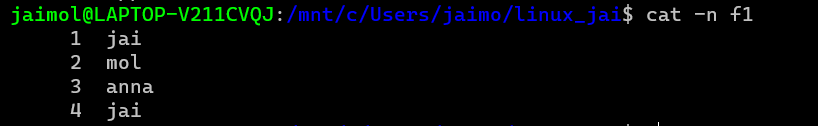
**Output Screenshot**



14. cat -n filename – display the content with line numbers

$cat -n c.txt

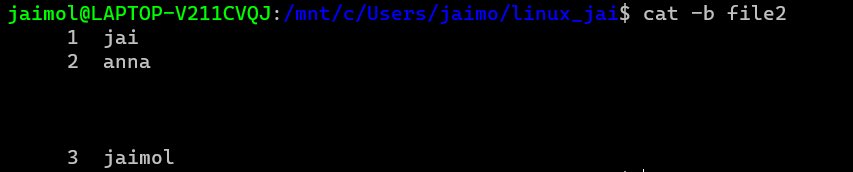
**Output Screenshot**



14.1 cat -b filename – remove empty line numbers

$cat -b car.txt

**Output Screenshot**



14. cat -e filename – append $ at end of each line.

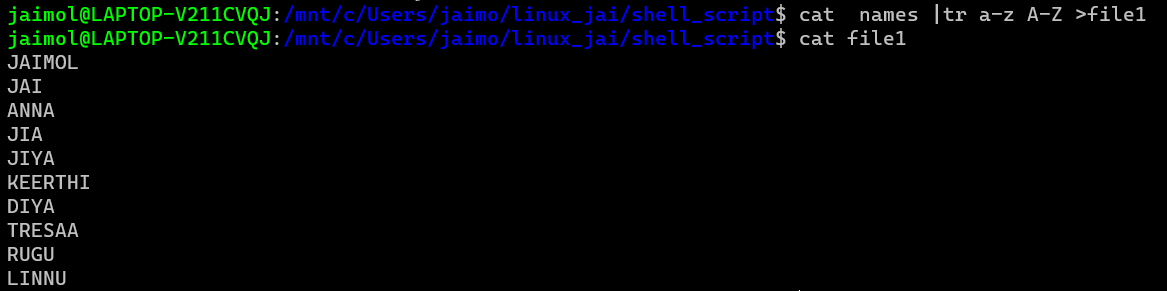
$cat -e car.txt

**Output Screenshot**

15. cat filename| tr a-z A-Z > filename – change context of file to uppercase.

$cat car.txt| tr a-z A-Z > output.txt

**Output Screenshot**



**Result**

The program was executed and the result was successfully obtained. Thus CO2 was obtained.

**Experiment No.: 4**

**Aim: Familarization of linux commands.**

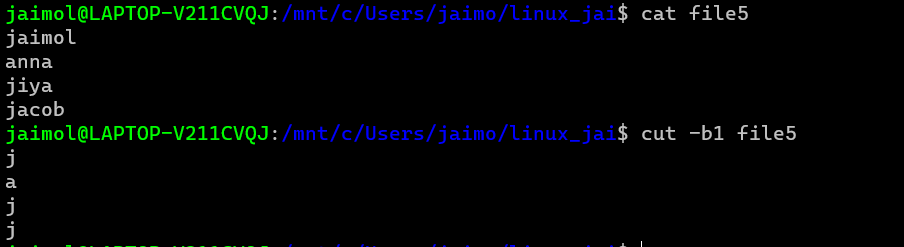
**CO2: Perform System Administration Tasks.**

**Procedure**

1. cut – for cutting out the sections from each lines of files and writing the result to standard output.

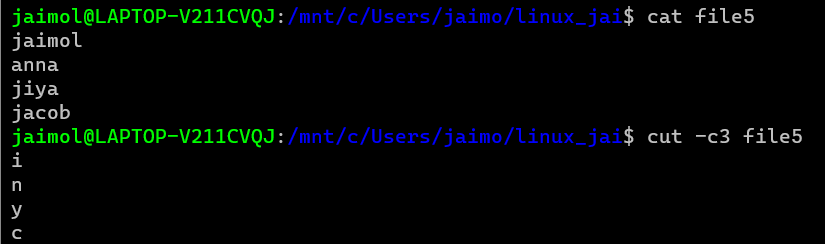
1.1 cut -b filename – cut the letters of words from each lines

**Output Screenshot**



1.2 $cut -c3 marvel1 – cutting the third letter of words from each line.

**Output Screenshot**

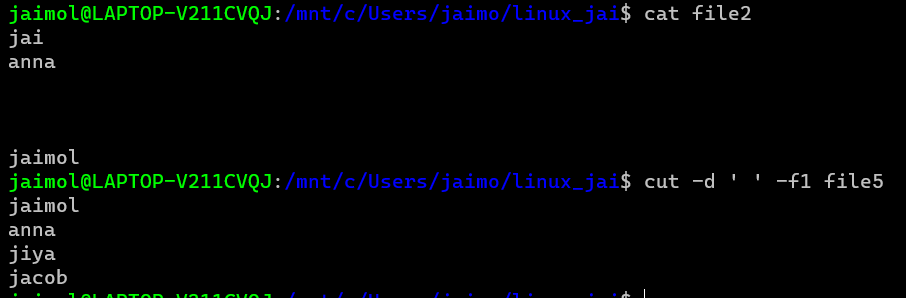


1.3 cut -d - -f1 filename – cutting the highfen(delimiter) from the file.

**Output Screenshot**

1.4 $cut -c 1,4,6 marvel1 – cutting 1st,4th and 6th letters.

**Output Screenshot**



2. paste- join files horizontally[each file consisting of different lines]

2.1 paste filename1 filename2

$paste mark1 mark2

**Output Screenshot**



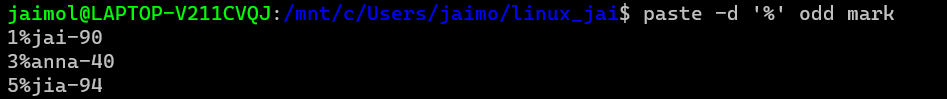
2.2 $paste mark1 mark2 > mark3

$cat mark3

**Output Screenshot**

2.3 $paste -d ‘%’ mark1 mark2 – appending % at list.

**Output Screenshot**



2.5 paste -s filename – display the content in the same line.

**Output Screenshot**

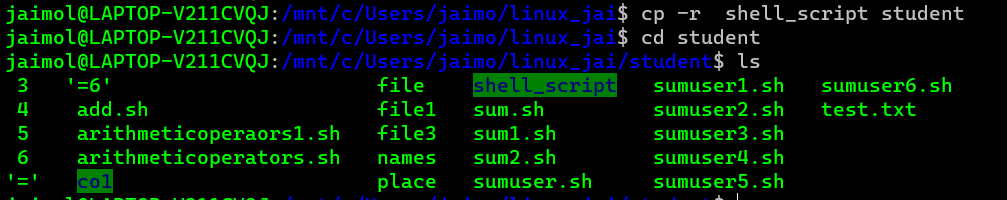


3. cp file1 file2- to copy the content to an existing file.

**Output Screenshot**

3.1 $cp -r – copy a directory along with subdirectory to another directory.

**Output Screenshot**



**Result**

The program was executed and the result was successfully obtained. Thus CO2 was obtained.

**Experiment No.: 5**

**Aim: Familarization of linux commands.**

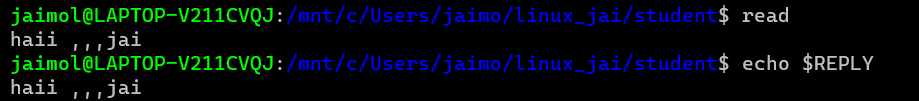
**CO2: Perform System Administration Tasks.**

**Procedure**

1. read- to read the content of a file

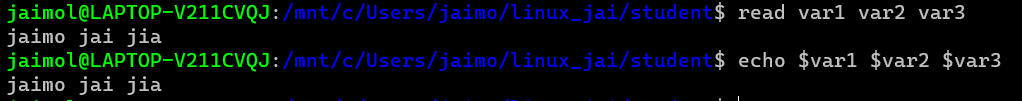
**Output Screenshot**





* 1. $read var1 var2 var3

**Output Screenshot**

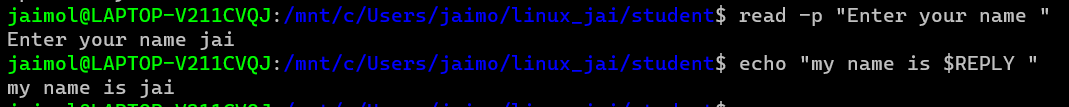


1.2 $ read

**Output Screenshot**

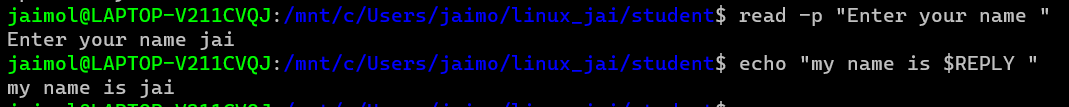
1.3 read -p -> prompt an input from the user.

**Output Screenshot**



$ echo “my name is $REPLY”

**Output Screenshot**



1.4 $read -n 6 -p “ “ -> able to enter 6 characters only.

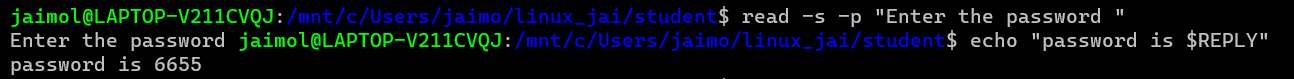
$ read -n 6 -p “Enter 6 characters only.

**Output Screenshot**



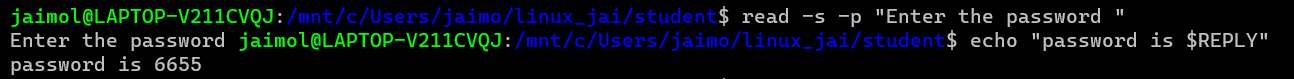
1.5 $ read -s -p “enter the password:”

**Output Screenshot**



$ echo “password is $REPLY”

**Output Screenshot**



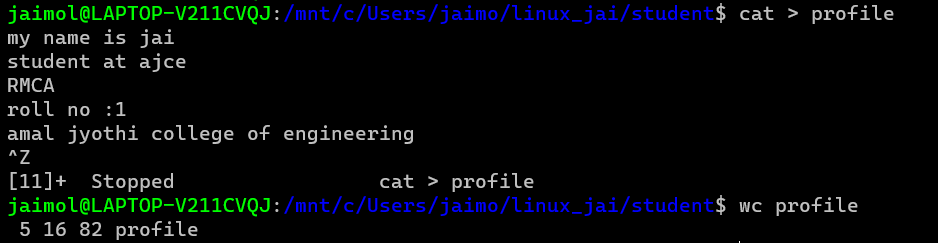
$ cat > profile

**Output Screenshot**

2. wc – to display the word count in an format.

$ wc profile

**Output Screenshot**

****

2.1 $wc -l profile : to display the no.of lines.

**Output Screenshot**

****

2.2 $wc -m profile : display the no.of characters.

**Output Screenshot**

****

2.3 $ wc -c profile : display no.of bytes.

**Output Screenshot**

****

2.4 $ wc – w profile : display the no.of words

**Output Screenshot**

****

2.5 $wc -L profile : print length of the longest line.

**Output Screenshot**

****

$ cat contentlong.txt

**Output Screenshot**

3. more – similar to cat to display the content, the only difference is that in case of large file cat command output will scroll off your screen while more command display output one screenful at a time.

$ more contentlong.txt

**Output Screenshot**

3.1 $ more +20 contentlong.txt -> display contents after 20 lines.

**Output Screenshot**

3.2 $ more +/ms-dos{pattern} contentlong.txt -> search the pattern string from the content and view all the instances,by navigating through the result.

**Output Screenshot**

3.3 $more -p contentlong.txt -> clear the screen and show the output.

**Output Screenshot**

3.4 $ more -d contentlong.txt -> help the users to navigate,press space to continue,q to quit

**Output Screenshot**

**Result**

The program was executed and the result was successfully obtained. Thus CO2 was obtained.

**Experiment No.: 6**

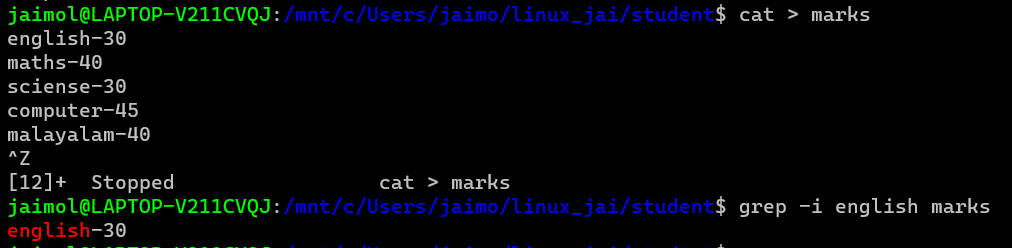
**Aim: Familarization of linux commands.**

**CO2: Perform System Administration Tasks.**

**Procedure**

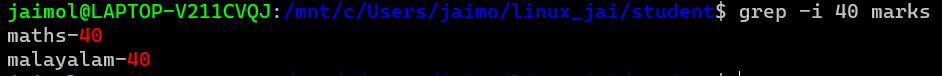
1. grep-  is used to filter the contents,which makes our search easy.

**Output Screenshot**



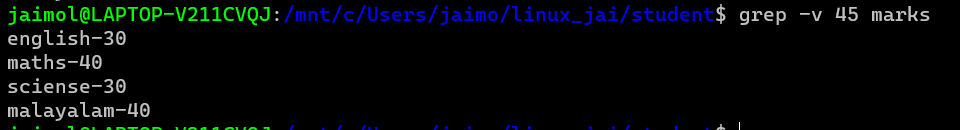
1.1  $grep -i content file\_name   -> case sensitive search

**Output Screenshot**



1.2 $grep -v content filename   ->inverted search(all content except the search content displays)

**Output Screenshot**



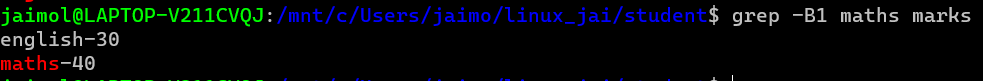
1.3 $grep -A1 content file\_name   ->view the content along with the  line after.

**Output Screenshot**



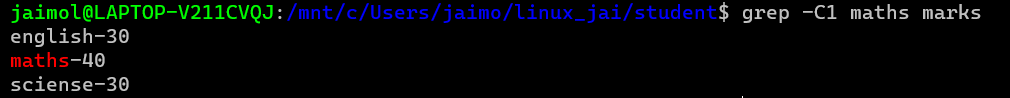
1.4 $ grep -B1 content file\_name     ->display content before also.

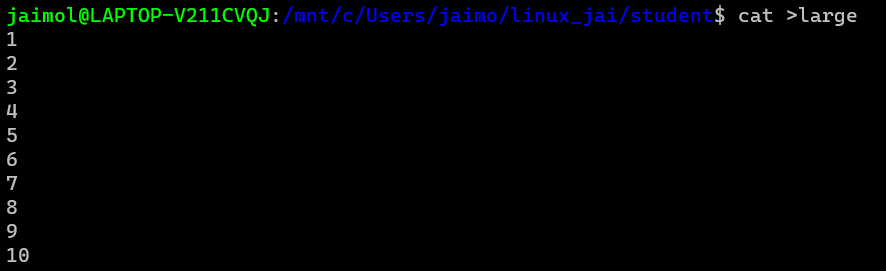
**Output Screenshot**



1.5 $ grep -C1 content file\_name    ->before and after the content displays.

**Output Screenshot**





1. head - display the top content of the file by default.

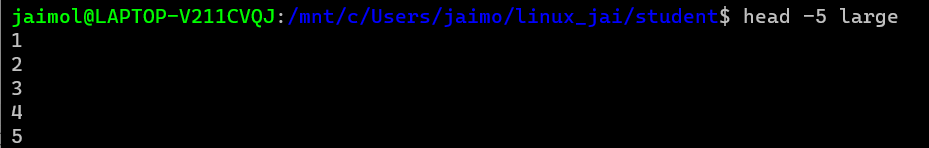
$ head file\_name

**Output Screenshot**



2.1 $ head -5 file\_name  ->displays first 5 lines.

**Output Screenshot**



1. tail- display the content of the file by default it displays last ten lines.

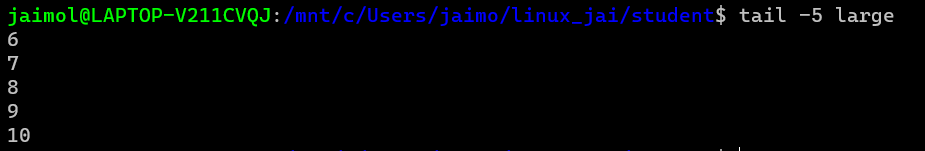
$tail file\_name

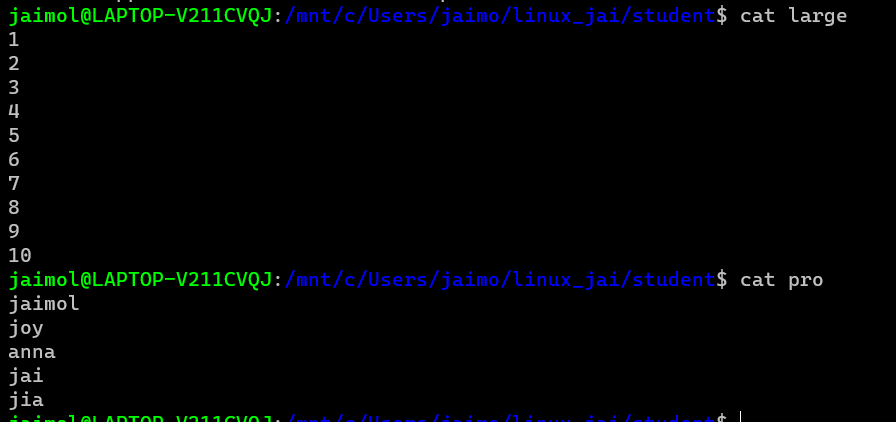
**Output Screenshot**



3.2 $tail -5 file\_name

**Output Screenshot**

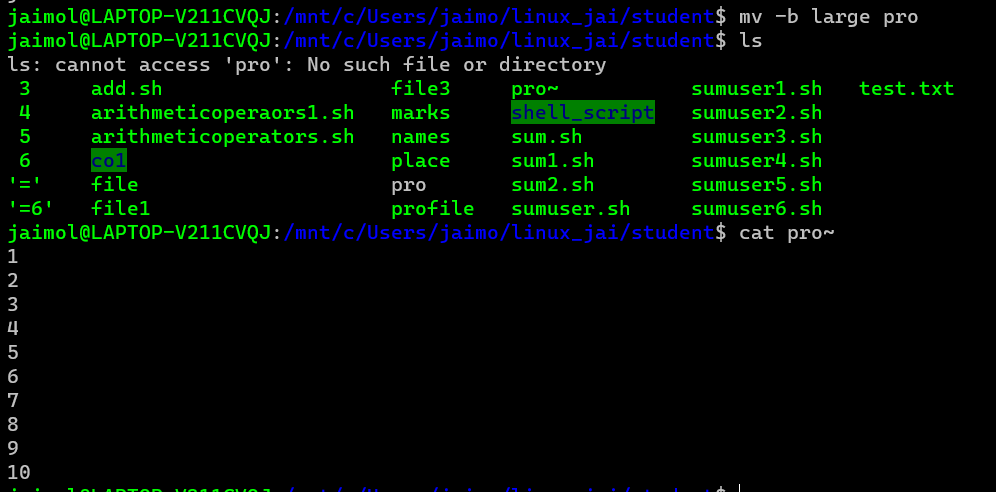




4.. mv - move

$mv -b file1 flie2   ->backups

**Output Screenshot**



4.1 $mv -i file1 file2 ->displays prompt message.

**Output Screenshot**

**Result**

The program was executed and the result was successfully obtained. Thus CO2 was obtained.

**Experiment No.: 7**

**Aim: Familarization of linux commands.**

**CO2: Perform System Administration Tasks.**

**Procedure**

1. expr- evaluate the given expression and display the output.

1.1 $ expr 12 + 8

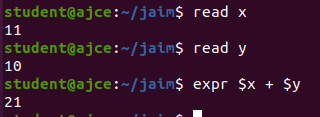
1.2 $ expr 12 - 8

1.3 $ expr 12 / 4

1.4 $ expr 12 /\* 3

1.5 $ expr $x + $y

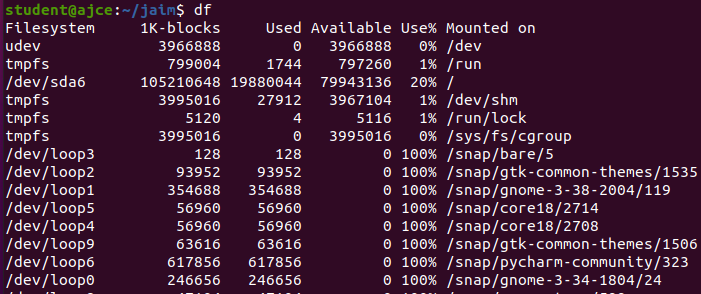
**Output Screenshot**



1. df -  disutilised information (represent system disk space usage)

$ df

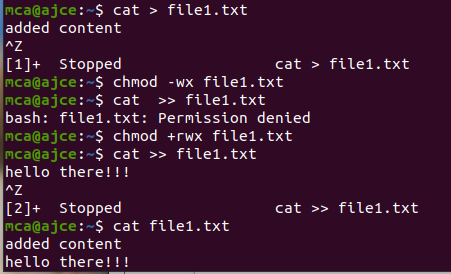
**Output Screenshot**



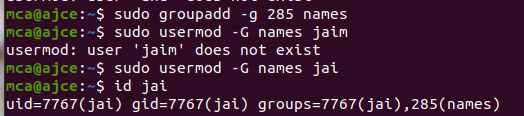
1. du - how much space a file or directory takes in the current directory

$ du file1.txt

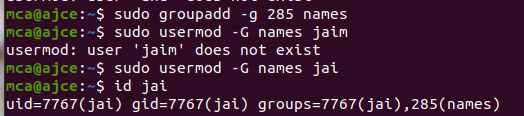
**Output Screenshot**



1. $ sudo add user\_name  -> add a new user to the system.

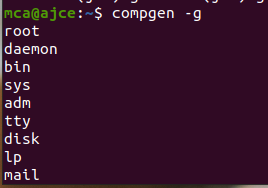


**Output Screenshot**



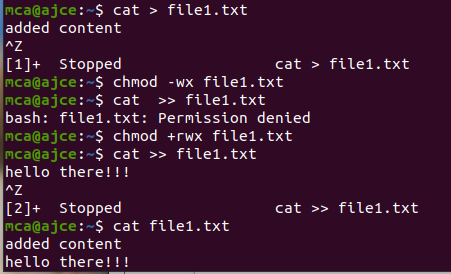
1. $ compgen -g

**Output Screenshot**



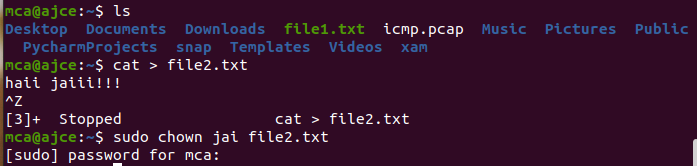
1. $ chmod -wx file1.txt  ->deny permission for write and execute.
2. $ chmod +rwx file1.txt   ->give permission to read,write,execute

**Output Screenshot**



8. $ sudo chown user\_name file1.txt ->change owner permission

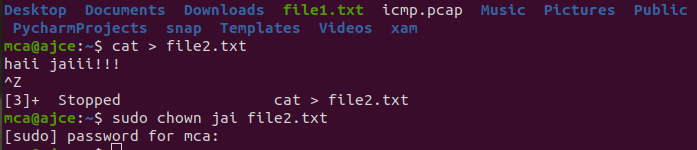
1. **Output Screenshot**



10. $ sudo userdel username

1. $ sudo groupdel groupname

**Output Screenshot**





**Result**

The program was executed and the result was successfully obtained. Thus CO2 was obtained.

**Experiment No.: 8**

**Aim: Familarization of linux commands.**

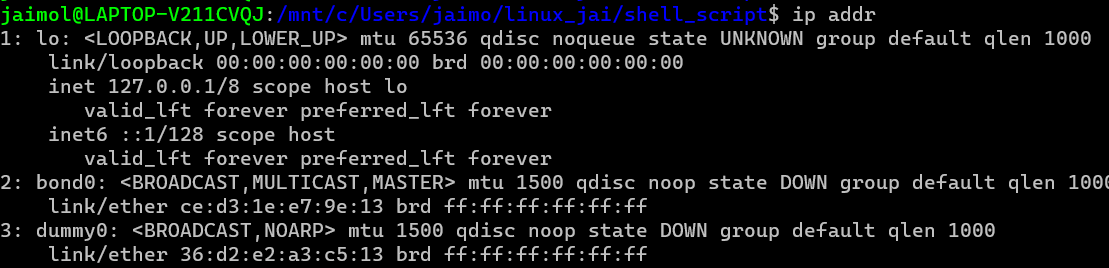
**CO2: Perform System Administration Tasks.**

**Procedure**

1. ip addr -> ip address

$ ip addr

**Output Screenshot**



2. $ sudo apt-get update

**Output Screenshot**

3. $ sudo apt-get install openshh-server

**Output Screenshot**

4. $ sudo ufw allow 22 -> 22-port number

**Output Screenshot**

5. ssh user@portnumber -> ssh-secure shell protocol : used to securely connect to a remote server or a system.ssh is secured in reuse,it transfer data in encrypted form between host and clients.

**Output Screenshot**

6. $ shh-keygen

**Output Screenshot**

7. $ ps

**Output Screenshot**

7.1 $ ps -u user\_name

**Output Screenshot**

7.2 $ ps -c chrome

**Output Screenshot**

7.3 $ ps -f -p 2168

**Output Screenshot**

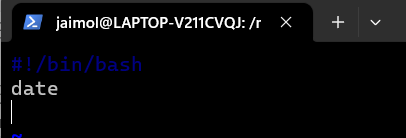
**RESULT**

The program was executed and the result was successfully obtained. Thus CO2 was obtained.

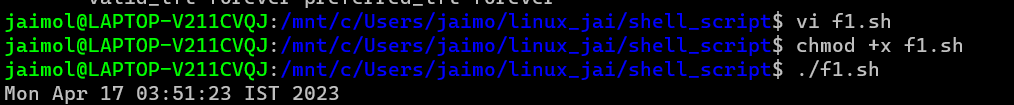
**Experiment No.: 10**

**Aim: Shell script to display date.**

**Procedure**



**Output Screenshot**

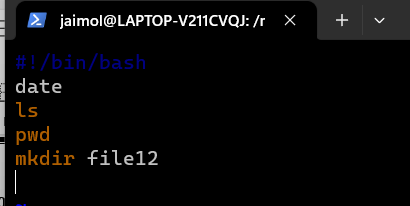


**RESULT**

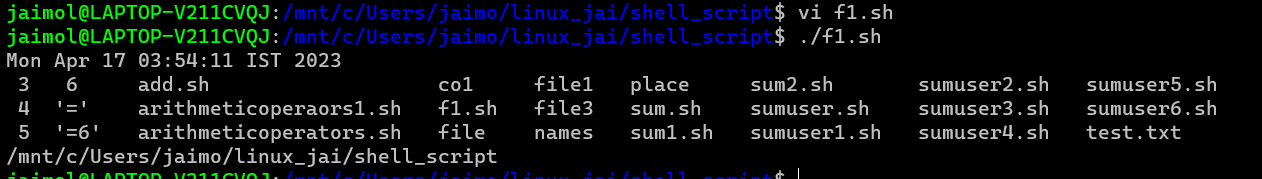
**Experiment No.:11**

**Aim: Shell script to display pwd,ls,date commands .**

**Procedure**



**Output Screenshot**

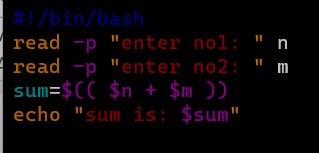


**RESULT**

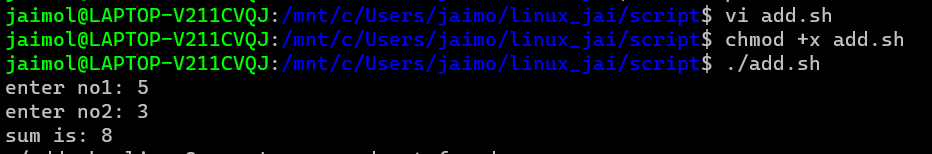
**Experiment No.: 16**

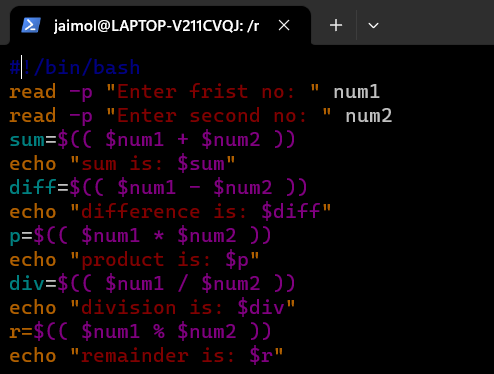
**Aim: Shell script to demonstrate Arithmetic operations.**

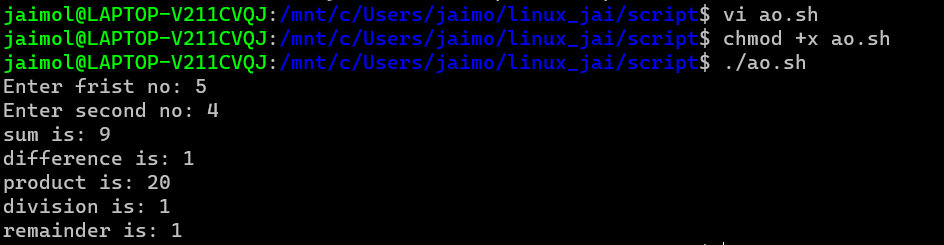
**Procedure**



**Output Screenshot**

****

****

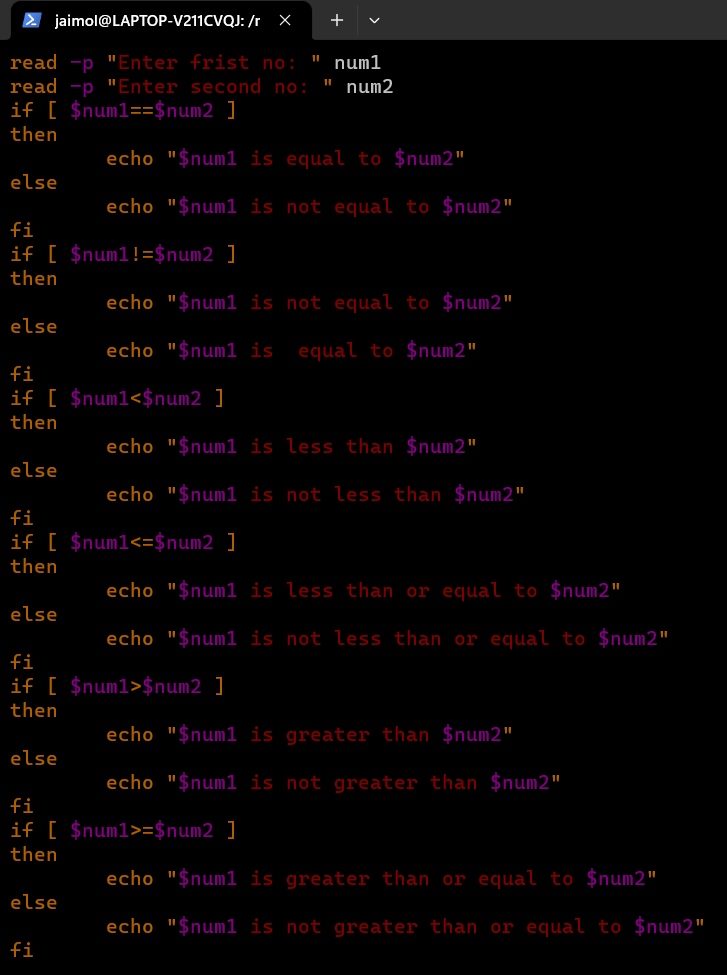
****

**RESULT**

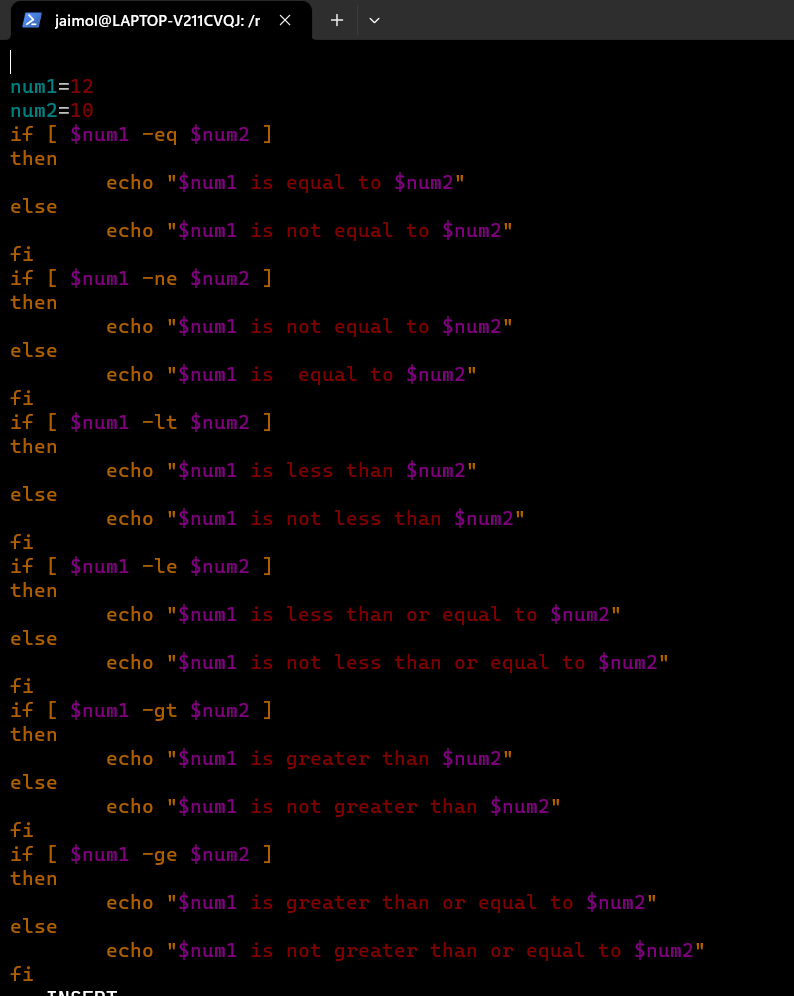
**Experiment No.: 17**

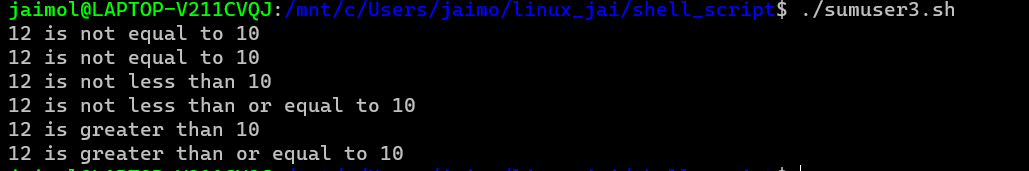
**Aim: Shell script to demonstrate Relational operations.**

**Procedure**

****

**Output Screenshot**

****

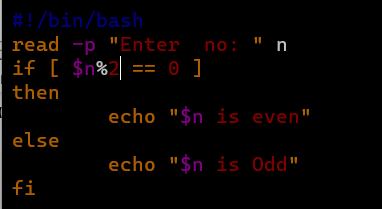
****

**RESULT**

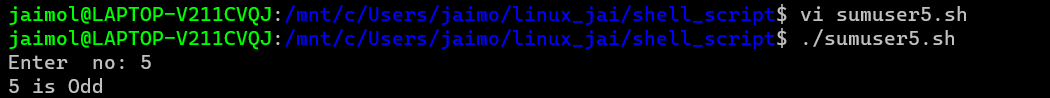
**Experiment No.: 18**

**Aim: Shell script to check whether a number is odd or even.**

**Procedure**

****

**Output Screenshot**

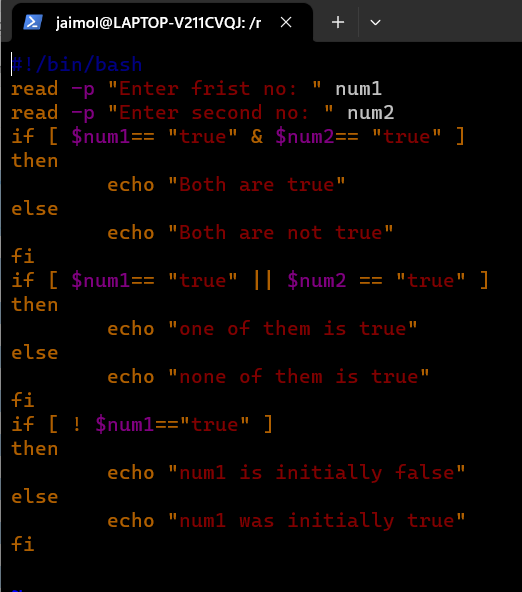
****

**RESULT**

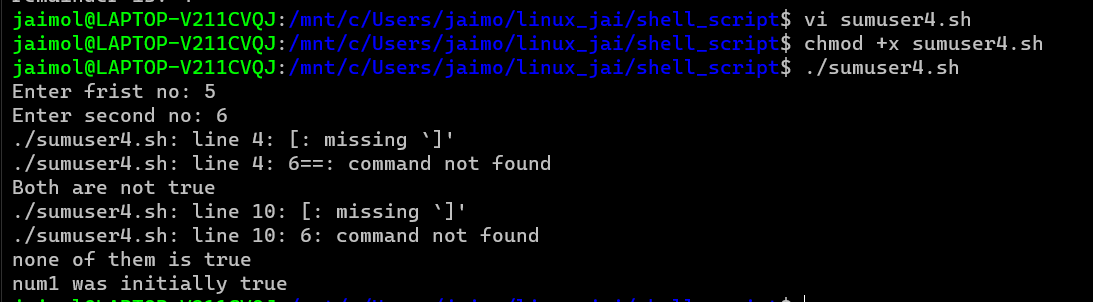
**Experiment No.: 19**

**Aim: Shell script to demonstrate Boolean operations.**

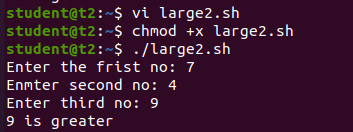
**Procedure**

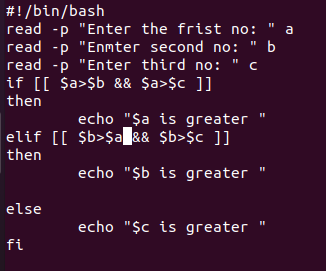
****

**Output Screenshot**

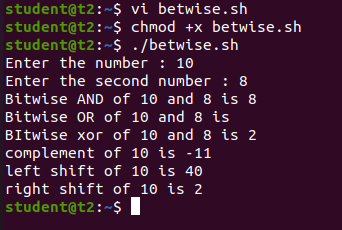
****

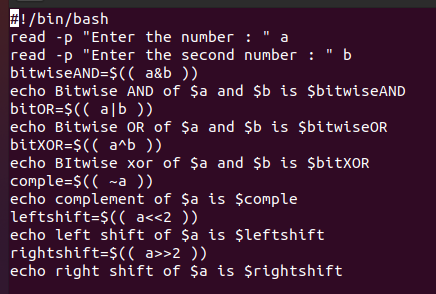
Largest among three nos:



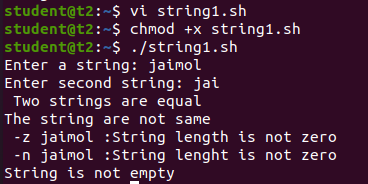


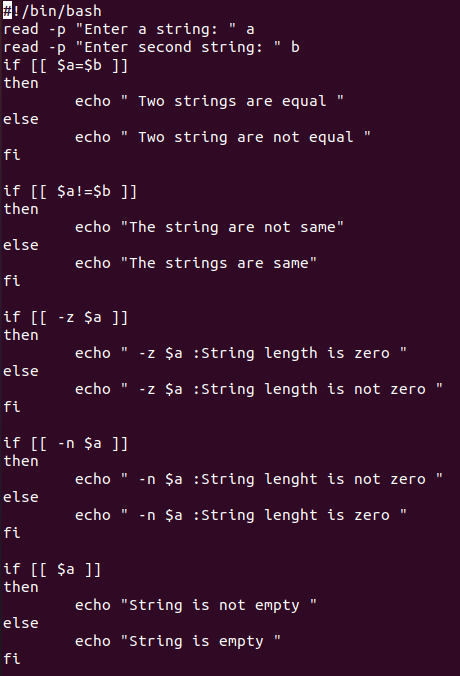
Bitwise operators





String operations





File-type operations

