Practical 1

Installation of Linux Operating System.

1) **Download the Linux distribution of your choice.**
   If you're new to Linux, consider trying a lightweight and easy to use distribution, such as Ubuntu or Linux Mint. Linux distributions (known as "distros") are typically available for free to download in ISO format. You can find the ISO for the distribution of your choice at the distribution's website. This format needs to be burned to a CD or USB stick before you can use it to install Linux. This will create a Live CD or Live USB.
   a) A Live CD or Live USB is a disk that you can boot into, and often contains a preview version of the operating system that can be run directly from the CD or USB stick.
   b) Install an image burning program, or use your system's built-in burning tool if you are using Windows 7, 8, or Mac OS X. Pen Drive Linux and UNetBootin are two popular tools for burning ISO files to USB sticks.
2) **Boot into the Live CD or Live USB.**
   Most computers are set to boot into the hard drive first, which means you will need to change some settings to boot from your newly-burned CD or USB. Start by rebooting the computer.
   a) Once the computer reboots, press the key used to enter the boot menu. The key for your system will be displayed on the same screen as the manufacturer's logo. Typical keys include F12, F2, or Del.
   b) Once you're in the boot menu, select your live CD or USB. Once you've changed the settings, save and exit the BIOS setup or boot menu. Your computer will continue with the boot process.
3) **Start the installation process.**
   If you're trying out the distro, you can launch the installation from the application on the desktop. If you decided not to try out the distribution, you can start the installation from the boot menu.
   a) You will be asked to configure some basic options, such as language, keyboard layout, and timezone.
4) **Create a username and password.**
   You will need to create login information to install Linux. A password will be required to log into your account and preform administrative tasks.
5) **Set up the partition.**
   Linux needs to be installed on a separate partition from any other operating systems on your computer if you intend dual booting Linux with another OS. A partition is a portion of the hard drive that is formatted specifically for that operating system. You can skip this step if you don't plan on dual booting.
   a) Distros such as Ubuntu will set a recommended partition automatically. You can then adjust this manually yourself. Most Linux installations require at least 20 GB, so be sure to set aside enough room for both the Linux operating system and any other programs you may install and files you may create.

b) If the installation process does not give you automatic partitions, make sure that the partition you create is formatted as Ext4. If the copy of Linux you are installing is the only operating system on the computer, you will most likely have to manually set your partition size.

6) **Boot into Linux.**

Once the installation is finished, your computer will reboot. You will see a new screen when your computer boots up called “GNU GRUB”. This is a boot loader that handles Linux installations. Pick your new Linux distro from the list. This screen may not show up if you only have one operating system on your computer. If this screen isn't being presented to you automatically, then you can get it back by hitting shift right after the manufacturer splash screen.

Practical 2

Introduction to the structure of Linux Operating System.

The structure diagram of LINUX operating system includes following parts

1. **Kernel**

   It is the heart of LINUX operating system which is responsible for all major activities. Kernel interacts with actual hardware in machine language directly. It is collection of programs written in C language which will directly communicate with hardware. There is only one kernel for each system. it is part LINUX operating system that is loaded in memory, when system is booted it manages system resources allocated time between users and processes, decides priority, preform requests from hardware and all other things related to user.

   The kernel has various functions as follows-

   - To manages computer memory
   - To manages the file system
   - To control access to the computer
   - To handle the interrupts
   - To handle the errors to perform i/p services
   - To carries out all data transfer between file system and hardware
   - To calculate resources of computer among the users
   - To schedule various programs running in memory to allocate C.P.U time to all running programs

3. **Shell**

   It is part of Linux O.S. or it is software program acts as mediator between kernel and user. Shell reads command prompt and send request to execute program, so that shell is also called as command interpreter. The shell program stored at file called as '**sh**'.

   For each working with Linux at any time different shell programs are running. Thus, at a particular time there may several shells running in memory but only one kernel.

4. **Hardware**

   It includes the actual parts of computer through which a computer system works such as all the peripheral devices like CPU, Hard Disk Drive and RAM etc.

5. **Tools and applications**

   This layer includes user written application using shell program language C,C++ application and so on

Practical 3

Introduction to basic commands.

- ls

  The command “ls” stands for (List Directory Contents), List the contents of the folder, be it file or folder, from which it runs.

- history

  The “history” command stands for History (Event) Record, it prints the history of long list of executed commands in terminal.

- sudo

  The “sudo” (super user do) command allows a permitted user to execute a command as the superuser or another user, as specified by the security policy in the sudoers list.

- mkdir

  The “mkdir” (Make directory) command create a new directory with name path. However is the directory already exists, it will return an error message “cannot create folder, folder already exists”

- touch

  The “touch” command stands for (Update the access and modification times of each FILE to the current time). touch command creates the file, only if it doesn’t exist. If the file already exists it will update the timestamp and not the contents of the file.

- chmod

  The Linux “chmod” command stands for (change file mode bits). chmod changes the file mode (permission) of each given file, folder, script, etc.. according to mode asked for.

- cp

  The “copy” stands for (Copy), it copies a file from one location to another location

- mv

  The “mv” command moves a file from one location to another location.

- pwd

  The command “pwd” (print working directory), prints the current working directory with full path name from terminal.

- cd

  Finally, the frequently used “cd” command stands for (change directory), it change the working directory to execute, copy, move write, read, etc. from terminal itself.

- rm

  The command ‘rm‘ stands for remove. rm is used to remove files (s) and directories.

Practical 4

Introduction to more advance commands.

- grep

  The ‘grep‘ command searches the given file for lines containing a match to the given strings or words.

- man

  The ‘man‘ is the system’s manual pager. Man provides online documentation for all the possible options with a command and its usages. Almost all the command comes with their corresponding manual pages

- ps

  ps (Process) gives the status of running processes with a unique Id called PID.

- kill

  This command is used to kill process which is not relevant now or is not responding.

- cmp

  compare two files of any type and writes the results to the standard output. By default, ‘cmp‘ returns 0 if the files are the same; if they differ, the byte and line number at which the first difference occurred is reported.

- mount

  Mount is an important command which is used to mount a filesystem that don’t mount itself. You need root permission to mount a device.

- gcc

  gcc is the in-built compiler for ‘c‘ language in Linux Environment. A simple c program, save it on ur desktop as Hello.c (remember ‘.c‘ extension is must).

- g++

  g++ is the in-built compiler for ‘C++‘ , the first object oriented programming language. A simple c++ program, save it on ur desktop as Add.cpp (remember ‘.cpp‘ extension is must).

- chown

  The Linux “chown” command stands for (change file owner and group). Every file belongs to a group of user and a owner.

- echo

  echo echoes a text on the standard output.

Practical 5

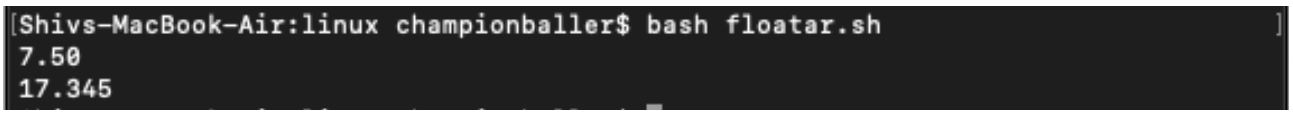
Write a shell script to perform integer arithmetic operations.

```
#!/bin/sh

echo "Please enter x:"
read x
echo "Please enter y:"
read y

echo "x+y = $(( x + y ))"
echo "x*y = $(( x * y ))"
echo "x/y = $(( x / y ))"
echo "x-y = $(( x - y ))"

echo "x%y = $(( x % y ))"
echo "x**y = $(( x ** y ))"
echo "++x = $(( ++x ))"
```

Practical 6

Write a shell script to perform floating point arithmetic operations..

```
#!/bin/sh

echo "scale=2; 15 / 2" | bc
echo "15.345 + 2" | bc
```



Practical 7

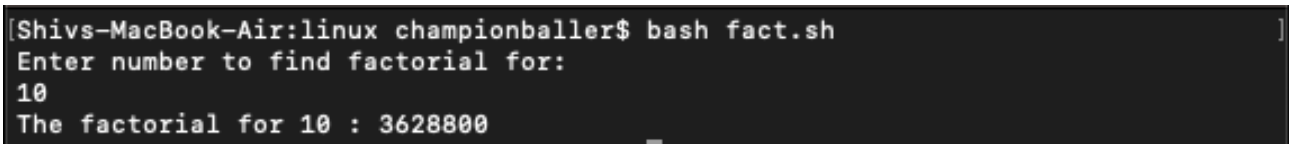
Write a shell script to display first 10 natural numbers.

```
#!/bin/sh

num=1
while [ "$num" -le "10" ]
do
echo $num
num=$((num + 1))
done
```

Practical 8

Write a shell script to find out the factorial of a given number.

```
#!/bin/sh

echo "Enter number to find factorial for:"

read n

current=1
product=1

while [ "$current" -le "$n" ]
do
product=$((product*current))
current=$((current+1))
done

echo "The factorial for $n : $product"
```



Pratical 9

Write a shell script to find out whether the given number is prime or not

```
#!/bin/sh

echo "Please enter number to be checked for prime property: "

read n

current=2

flag=0

while [ $((current*current)) -le $n ]
do
if (($n % $current == 0 )); then
flag=1
break
fi

current=$((current+1))
done

if [ "$flag" = "1" ]; then
echo not prime
else echo prime
fi
```

```
[Shivs-MacBook-Air:linux championballer$ bash prime.sh
Please enter number to be checked for prime property:
13
prime
```

Practical 10

Write a shell script to check the given file is writable or not.

```
#!/bin/bash

FILE="$1"

[ $# -eq 0 ] && exit 1

if [ -w "$FILE" ]
then
echo "Write permission is granted on $FILE"
else
echo "Write permission is NOT granted on $FILE"
fi
```

```
Shivs-MacBook-Air:linux championballer$ bash writable.sh prime.sh
Write permission is granted on prime.sh
```

Practical 11

Write a shell script that displays a list of all the files in the current directory to which the user has read, write and execute permissions

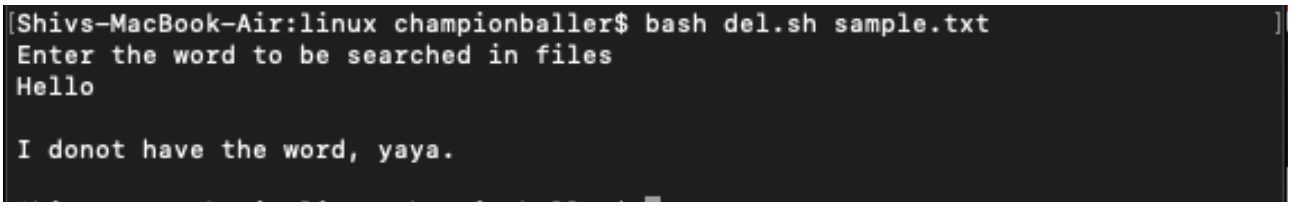
```
echo "List of Files which have Read, Write and Execute Permissions in Current Directory"

read=0
write=0
exe=0

for file in *
do
read=0
write=0
exe=0
if [ -r $file ]
then
read=1
fi
if [ -w $file ]
then
write=1
fi
if [ -x $file ]
then
exe=1
fi

echo "$file r:$read w:$write x:$exe"

done
```

```
Shivs-MacBook-Air:linux championballer$ bash display.sh
List of Files which have Read, Write and Execute Permissions in Current Director
y
LIST.md r:1 w:1 x:0
cat r:1 w:1 x:1
cmds r:1 w:1 x:1
compare.sh r:1 w:1 x:0
del.sh r:1 w:1 x:0
display.sh r:1 w:1 x:1
fact.sh r:1 w:1 x:0
file1.txt r:1 w:1 x:0
file2.txt r:1 w:1 x:0
floatar.sh r:1 w:1 x:0
intar.sh r:1 w:1 x:0
internal r:1 w:1 x:1
internal.sh r:1 w:1 x:0
name r:1 w:1 x:0
natural.sh r:1 w:1 x:0
prime.sh r:1 w:1 x:0
sample r:1 w:1 x:1
sample.txt r:1 w:1 x:0
sample1.txt r:1 w:1 x:0
sample2.txt r:1 w:1 x:0
vars.sh r:1 w:1 x:0
```

Practical 12

Write a shell script that deletes all lines containing a specified word in one or more files supplied as arguments to it.

```
if [ $# -eq 0 ]
then
echo "Please enter one or more filenames as argument"
exit
fi
echo "Enter the word to be searched in files"
read word
for file in $*
do
sed "/$word/d" $file | tee tmp
mv tmp $file
done
```



Practical 13

Write a shell script to accept two file names and check if both exists. If the second filename exists, then the contents of the first filename should be appended to it. If the second file name does not exist, then create a new file with the contents of the first file.

```
echo enter the file name
read first
echo enter the second file name
read second
if [ -e $first ]
then
if [ -e $second ]
then
cat $first>>$second
else
cat $first>$second
fi
fi
```