Navsahyadri Group of Institutions
Faculty of Engineering, PUNE

DEPARTMENT OF AIML ENGINEERING

LAB MANUAL
OSL

(Operating System Lab)

Academic Year 2023-24

S.E. AIML (SEM — II)

Prepared By - Prof. R. R. Bhuvad

ASSIGNMENT No: 1-A

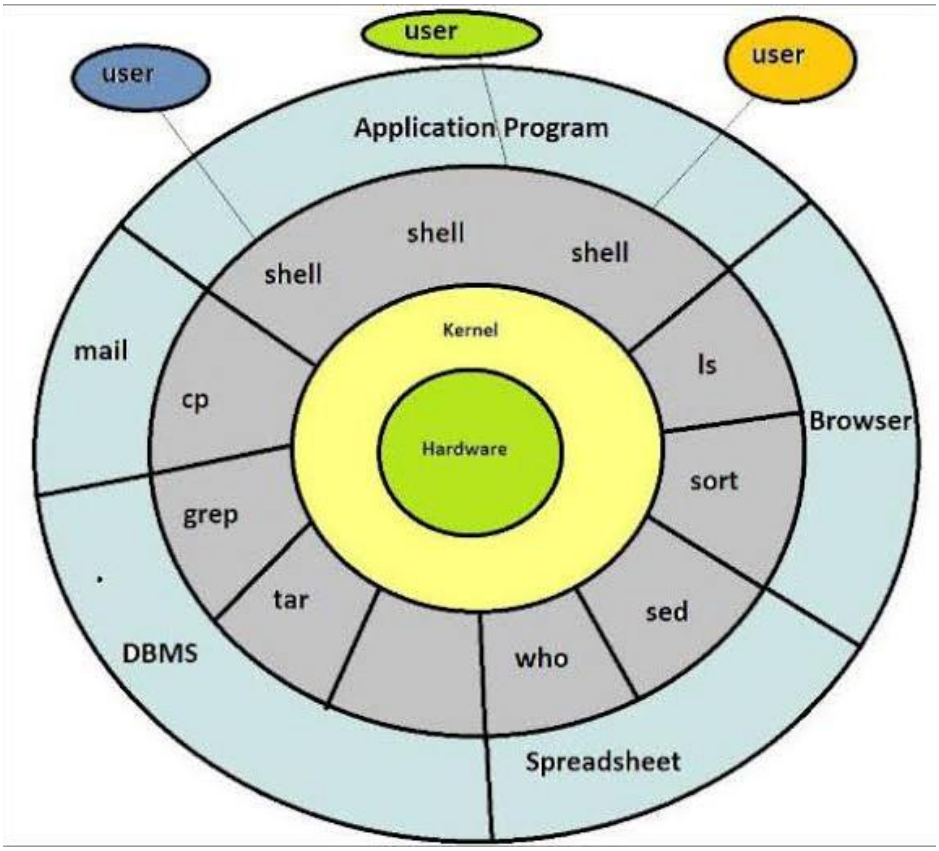
TITLE: Basic Linux Commands

AIM: Study of Basic Linux Commands: echo,ls,read,cat,touch,test,loops, arithmetic comparison, conditional loops, grep, sed etc.

THEROY:

Introduction to Linux Operating System

- Linux is a free-open source operating system.
- It is a free version of Unix.
- Linux is a powerful, multitasking, multiuser and flexible operating system that ate free to use and share.
- It was created by a person named Linus Torralds in 1991.
- In this source code is open for everyone to explore and modify.
- It is known for being efficient, means it can do a lot of task quickly, which means it doesn't cost a lot to use.



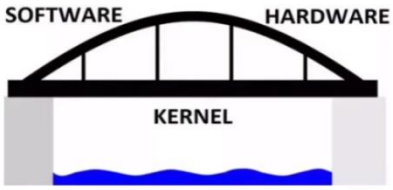
- ## Linux is divided in two parts

1. **Kernel**- It deals with central functions such as program execution, memory management, input/output management

2. **Set of commands and applications**–it deals with language compiler.

- Layers of Linux Operating System

1. **Hardware** –Hardware layer contains all the physical components (peripheral devices) of the computer such as RAM, HDD, CPU, Input/Output Devices. This layer is responsible for interacting with Linux OS.
2. **Kernel** – It is the core of the Linux based Operating System. It is a collection of routines. It is loaded into memory when system is booted and communicates directly with hardware.It deals with central functions such as program execution, memory management, input/output management, schedule processes, gives priority to resources.



3. **Shell** – It is an interface between user and Linux OS. It only provided services which is requested by the Linux user and further process it. For example, creating a file, opening a file, removing a file etc.
4. **Applications Utilities** – it contains all the applications or utilities which are present in your Linux such as – Text editor Document Processor, email Services, Web browser, Imageeditor, Database, GUI etc.

- Linux Basic Commands

1. **pwd -** Use the pwd command to find out the path of the current working directory (folder) you're in. The command will return an absolute (full) path, which is basically a path of all the directories that starts with a forward slash (/). An example of an absolute path is /home/username. It is used to display the location of current working directory

   Syntax: pwd
2. **cal –** View the calendar for a particular month in the terminal.

   Syntax – cal month year

   Ex- cal jan 2024

3. **ls -** The ls command is used to view/display the files available in the current directory. By default, this command will display the contents of your current working directory. If you want to see the content of other directories, type ls and then the directory's path.

   -rw-rw-r-- 1 user user 1176 Feb 16 00:19 1.c

   There are variations you can use with the ls command:

   ls Field Explanation

   - **:** normal file, d : directory,
   s : socket file, l : link file

   - **Field 1 – File Permissions**: Next characters specify the files permission. Every 3 characters specify read, write, execute permissions for user(root), group and others respectively in order. Taking the above example, -rw-rw-r– indicates read-write permission for user(root), read permission for group, and no permission for others respectively. If all three permissions are given to user(root), group and others, the format looks like -rwxrwxrwx
   - **Field 2 – Number of links**: Second field specifies the number of links for that file. In this example, 1 indicates only one link to this file.
   - **Field 3 – Owner**: Third field specifies owner of the file. In this example, this file is owned by username 'maverick'.
   - **Field 4 – Group**: Fourth field specifies the group of the file. In this example, this file belongs to" maverick' group.
   - **Field 5 – Size**: Fifth field specifies the size of file in bytes. In this example, '1176' indicates the file size in bytes.
   - **Field 6 – Last modified date and time**: Sixth field specifies the date and time of the last modification of the file. In this example, 'Feb 16 00:19' specifies the last modification time of the file.
   - **Field 7 – File name**: The last field is the name of the file. In this example, the file name is 1.c.

4. **cd -** To navigate through the Linux files and directories, use the cd command. It requires either the full path or the name of the directory, depending on the current working directory that you're in. It is used to change the directory

   Syntax: cd <directory name >/

   Ex- cd demo/

5. **cd –** come out of directory.
   Syntax: cd --

6. **mkdir -** Use mkdir command to make a new directory. If you type mkdir Music it will create a directory called Music.
   Syntax: mkdir directory_name
   Ex- mkdir Music

7. **touch -** The touch command allows you to create a blank new file through the Linux command line. As an example, enter touch demo.txt to create an text file entitled demo under the current directory.
   Syntax: touch directory_name.file_extension
   Ex- touch demo.txt

8. **rm -** The rm command is used to delete directories/files and the contents within them.
   Syntax- rm file_name
   Ex- rm music

9. **mv -** The primary use of the mv command is to rename files.
   Syntax- mv oldname.extnesion newname.extension
   Ex- mv demo.txt demo1.txt

10. **cat** - See the content of file.
    Syntax- cat file_name.ext
    Ex- demo.txt

11. **cat -** cat (short for concatenate) is one of the most frequently used commands in Linux. It joins two files (1 and 2) and stores the output of them in a new file (3).
    Syntax- cat filename1.ext filename.ext > filename3.ext
    Ex- cat unit1.txt unit2.txt > unit3.txt

12. **cp -** Use the cp command to copy files from the current directory to a different directory. For instance, the command cp scenery.jpg /home/username/Pictures would create a copy of scenery.jpg (from your current directory) into the Pictures directory.

13. **grep -** It lets you search through all the text in a given file. To illustrate, grep blue notepad.txt will search for the word blue in the notepad file. Lines that contain the searched word will be displayed fully.
    Syntax- <any linux command> | grep “String to find”
    Ex- cat unit1.txt | grep “Linux”

14. **echo** - Used to print the content written with echo.
    Syntax- echo string or echo “string”
    Ex- echo Hello world or echo “Hello World”

15. **chmod -** chmod is another Linux command, used to change the read, write, and execute permissions of files and directories.

- Shell Script commands-

- If you want to create a Script i.e. serious of commands you have to execute then you have to create a file with .sh extension.
- Create new script
  touch test.sh
- Always start your script with
  #!bin/bash
- To Print something
  #!bin/bash
  Echo Hello
- To run this file
  ./filename

./test.sh

- To give executable permissions
  chmod u+x filename.ext
  chmod u+x test.sh
- To see the permission is granted or not.
  ls –ltr
- To check the output
  ./test.sh

  Output - $Hello.

Example 1:- Use of variables and print data

```
#!/bin/bash
echo hello
name=Rucha
age=28
echo My name is $name and age is $age.
```

Example 2:- Take input from user using read command

```
#!/bin/bash
#echo what is your name?
#read name1
#echo My name is $name1.
```

Example 3:- Take input from user and do addition of it using expr command

```
echo "Enter two numbers"
read x
read y
sum=`expr $x + $y`
echo "Sum = $sum"
```

Example 4:- Check if two numbers are equal using test command

```
#!/bin/bash
a=10
b=20
# using test command to check if numbers are equal
if test "$a" -eq "$b"  then
  echo "a is equal to b"
else
  echo "a is not equal to b"
fi
```

Example 5:- Loop Control using while loop

```
a=0
while [ $a -lt 10 ]
do
  echo $a
  if [ $a -eq 5 ]
  then
    break
  fi
  a=`expr $a + 1`
done
```

ASSIGNMENT No: 1-B

Title: Shell Programming

AIM: Write a program to implement an address book with options given below:
Create address book, View address book, Insert a record, and Delete a record, Modify a record, Exit.

THEORY:

What is Shell?

A shell is special user program which provide an interface to user to use operating system services. Shell accept human readable commands from user and convert them into something which kernel can understand. It is a command language interpreter that execute commands read from input devices such as keyboards or from files. The shell gets started when the user logs in or start the terminal.

What Is Shell Scripting?

Usually shells are interactive that mean, they accept command as input from users and execute them. However some time we want to execute a bunch of commands routinely, so we have type in all commands each time in terminal.

As shell can also take commands as input from file we can write these commands in a file and can execute them in shell to avoid this repetitive work. These files are called **Shell**

Scripts or **Shell Programming**.

1. Each shell script is saved with **.sh** file extension eg. **myscript.sh**
2. Shell or the Command interpreter is the mediator which interprets the commands and then conveys them to the kernel which ultimately executes them.
3. Kernel is usually stored in a file called 'UNIX' where as the shell program in a file called 'sh'.
4. Types of shells:-
    i. Bourne shell (sh) or Bourne again shell (bash)
    ii. C shell (csh)
    iii. Korn shell (ksh)
5. A shell program is nothing but a series of unix commands.
6. Instead of specifying one job at a time, the shell is given a to-do-list – a program – that carries out an entire procedure
7. Such programs are known as shell scripts. Shell programming language incorporates most of the features that most modern day programming languages offer.

Shell variables –

Rules for building shell variables are as follows:

1) A variable name is any combination of alphabets, digits and an underscore ('_').
2) No commas or blanks are allowed within a variable name.
3) The first character of a variable name must either be an alphabet or an underscore.
4) Variable names should be of any reasonable length.
5) Variable names are case sensitive.

Keywords for accepting input –

read

Displaying output - echo

Assigning value to variables – Values can be assigned to variables through read statement or also by using a simple assignment operator. For ex: age=30

Note : To print or access value of a variable use '$' . For ex: To print value of variable 'flag' write - echo $flag

Arithmetic in Shell script –

1. All shell variables are string variables, hence to carry out arithmetic operations use expr command which evaluates arithmetic expressions.
2. More than one assignment can be done in a single statement.

3. Before and at the end of expr keyword use ` (back quote) sign not the (single quote i.e. ') sign which is generally above TAB key.
4. Terms of the expression provided to expr must be separated by blanks. Thus expression expr 10+20 is invalid.
5. The '*" symbol must be preceded by a \ ,otherwise the shell treats it as a wildcard character for all files in the current directory

OPERATORS USED IN SHELL SCRIPT –

–gt Greater than
–lt Less than
–ge Greater than or equal to
–le Less than or equal to
–ne Not equal to
–eq Equal to
–a Logical AND
–o Logical OR
! Logical NOT

CONTROL INSTRUCTIONS IN SHELLS –

There are four types of control instructions in shell :

- Sequence Control Instruction.
- Selection or Decision control Instruction
- Repetition or Loop control Instruction
- Case Control Instruction

Decision statements –

If-then-else-fi statements: if condition then Commands else Commands fi
1) For statements : for control variable in value1 value 2 value3 do Command list done
2) While statements : while control command do Command list Done
3) Until statements: until control command do Command list done

Steps to write and execute a script

1. Open the terminal. Go to the directory where you want to create your script.
2. Create a file with . sh extension.
3. Write the script in the file using an editor.
4. Make the script executable with command chmod +x .
5. Run the script using ./<fileName>

Example:

Concatenate two strings. AIM: To write a shell program to concatenate two strings.

ALGORITHM:

Step 1: Start the program.
Step 2: Read the first string.
Step 3: Read the second string.
Step 4: Concatenate the two strings.
Step 5: Enter into the escape mode for the execution of the result and verify the output.
Step 7: Stop the program.

SOURCE CODE:

```
echo "enter the first string"
read str1 echo "enter the second string"
read str2
echo "the concatenated string is" $str1$str2
```

OUTPUT:

```
$ vi concat.sh
$ sh concat.sh
Enter first string: Hello
Enter first string: World
The concatenated string is HelloWord.
```