

UNIVERSITY SCHOOL OF INFORMATION COMMUNICATION AND TECHNOLOGY



LINUX AND WIN32 PROGRAMMING PRACTICAL FILE

IT – 361

Submitted To:

Ms. Shweta Sharma

Submitted By:

Name: Aryan Garg

Batch: BTech IT 5th Sem

Enrollment Number: 01416401520

Index

S.No	List of Commands	Page No
1.	Basic Commands <ul style="list-style-type: none">● echo● printf● bc● whoami● clear● passwd● uname● uptime● logname● calender● date● hostname● tty	
2.	Directory Commands <ul style="list-style-type: none">● pwd● cd● mkdir● rmdir	
3.	File Commands <ul style="list-style-type: none">● cat● cp● copy● rm● wc● cmp	

	<ul style="list-style-type: none"> • diff • comm 	
4.	List Commands <ul style="list-style-type: none"> • ls • ls -a • ls -l • ls -r 	
5.	Permission: <ul style="list-style-type: none"> • chmod • chgrp • chown 	
6.	Process Commands: <ul style="list-style-type: none"> • ps • top • bg • fg • nice • renice • kill 	
7.	Grep Commands	
8.	Word Count	
9.	Wild Characters: *, ?, []	
10.	Introduction to Vi Editor and its commands	
11.	WSS to enter 2 strings	
12.	WSS to add, subtract, multiply and divide two numbers.	
13.	WSS to find the largest of two numbers.	

14.	WSS to check whether a given number is even or odd.	
15.	WSS to list all the directory files in a directory.	
16.	WSS to find factorial of a number.	
17.	WSS to design a calculator	
18.	WSS to create a child process and allow parent to display "Parent" and child to display "child" on the screen.	

Lab 1

Basic Commands

- **echo**

- echo command in Linux is used to display line of text/string that are passed as an argument. This is a built-in command that is mostly used in shell scripts and batch files to output status text to the screen or a file.

- Syntax:

echo [option] [string]

echo [string]

- Options:

- \b backspace
- \c produces no further output
- \e escape
- \f form feed
- \n new line
- \r carriage return
- \t horizontal tab
- \v vertical tab
- -n do not output the trailing newline-e enable interpretation of backslash escapes
- -E disable interpretation of backslash escapes (default)
- --help display this help and exit
- --version output version information and exit
- If -e is in effect, the following sequences are recognized:

```
aryan@aryan-VirtualBox:~$ echo "This is Aryan's PC"
This is Aryan's PC
```

- **printf**

- “**printf**” command in Linux is used to display the given string, number or any other format specifier on the terminal window. It works the same way as “printf” works in programming languages like C.

- Syntax:

\$printf [-v var] format [arguments]

- OPTIONS:

- --help display this help and exit
- --version output version information and exit

```
aryan@aryan-VirtualBox:~$ printf "%s\n" "Magic"
Magic
```

- **bc**

- bc command is used for command line calculator. It is similar to basic calculator by using which we can do basic mathematical calculations.
- Arithmetic operations are the most basic in any kind of programming language. Linux or Unix operating systems provides the bc command and expr command for doing arithmetic calculations. You can use these commands in bash or shell script also for evaluating arithmetic expressions.
- **Syntax:**
 - `bc [-h,-l,-w,-s,-q,-v] [long-options] [file ...]`
- **Options:**
 - `-h, {- -help }` : Print the usage and exit
 - `-i, {- -interactive }` : Force interactive mode
 - `-l, {- -mathlib }` : Define the standard math library
 - `-w, {- -warn }` : Give warnings for extensions to POSIX bc
 - `-s, {- -standard }` : Process exactly the POSIX bc language
 - `-q, {- -quiet }` : Do not print the normal GNU bc welcome
 - `-v, {- -version }` : Print the version number and copyright and quit.

```
aryan@aryan-VirtualBox:~$ echo "76 * 5" | bc
380
```

- **whoami**

- whoami command is used both in Unix Operating System and as well as in Windows Operating System.
- It is basically the concatenation of the strings “who”, “am”, “i” as
 - whoami.
- It displays the username of the current user when this command is invoked.
- It is similar as running the id command with the options -un.
- The earliest versions were created in 2.9 BSD as a convenience form for who am i, the Berkeley Unix who command's way of printing just the logged in user's identity. The GNU version was written by Richard Mlynarik and is part of the GNU Core Utilities (coreutils).
- Syntax:

Whoami

```
aryan@aryan-VirtualBox:~$ whoami  
aryan
```


- **clear**

- o clear is a standard Unix computer operating system command that is used to clear the terminal screen. This command first looks for a terminal type in the environment and after that, it figures out the terminfo database for how to clear the screen. And this command will ignore any command-line parameters that may be present. Also, the clear command doesn't take any argument and it is almost similar to cls command on a number of other Operating Systems.

- o Syntax:

\$clear

```
aryan@aryan-VirtualBox:~$ echo "This is Aryan's PC"
This is Aryan's PC
aryan@aryan-VirtualBox:~$ whoami
aryan
aryan@aryan-VirtualBox:~$ clear
```

After executing clear command:

```
aryan@aryan-VirtualBox:~$ █
```

- **passwd**

- o passwd command in Linux is used to change the user account passwords. The root user reserves the privilege to change the password for any user on the system, while a normal user can only change the account password for his or her own account.

- o Syntax:

passwd [options] [username]

- o OPTIONS:

- -k: The option -k, is used to indicate that the update should only be for expired authentication tokens (passwords); the user wishes to keep their non-expired tokens as before.
- -l: This option is used to lock the specified account and it is available to root only. The locking is performed by rendering the encrypted password into an invalid string (by prefixing the encrypted string with an !).
- -stdin: This option is used to indicate that passwd should read the new password from standard input, which can be a pipe.
- -u: This is the reverse of the -l option - it will unlock the account password by removing the !
- -d: This is a quick way to delete a password for an account. It will set the named account password less. Available to root only.

- -e: This is a quick way to expire a password for an account. The user will be forced to change the password during the next login attempt. Available to root only.
- -n: This will set the minimum password lifetime, in days, if the user's account supports password lifetimes. Available to root only.
- -x: This will set the maximum password lifetime, in days, if the user's account supports password lifetimes. Available to root only.
- -w: This will set the number of days in advance the user will begin receiving warnings that her password will expire, if the user's account supports password lifetimes. Available to root only.
- -l: This will set the number of days which will pass before an expired password for this account will be taken to mean that the account is inactive and should be disabled, if the user's account supports password lifetimes. Available to root only.
- -S: This will output a short information about the status of the password for a given account. Available to root user only.

```
Current password:
New password:
Retype new password:
Bad: new password is just a wrapped version of the old one
New password:
Retype new password:
passwd: password updated successfully
```

- **uname**

- `uname` command is used to display basic information about the operating system and hardware. With options, `Uname` prints kernel details, and system architecture. `Uname` is the short name for 'UNIX name'. `Uname` command works on all Linux and Unix-like operating systems.

- SYNTAX:

`uname [OPTION]`

- Options:

- `-a, --all`: print all information, in the following order, except omit `-p` and `-i` if unknown:
 - `-s, --kernel-name`: print the kernel name
- `-n, --nodename`: print the network node hostname
- `-r, --kernel-release`: print the kernel release
- `-v, --kernel-version`: print the kernel version
- `-m, --machine`: print the machine hardware name

- -p, --processor: print the processor type or "unknown"
- -i, --hardware-platform: print the hardware platform or "unknown"
- -o, --operating-system: print the operating system
- --help: display this help and exit
- --version: output version information and exit

```
aryan@aryan-VirtualBox:~$ uname -a
Linux aryan-VirtualBox 5.15.0-43-generic #46-Ubuntu SMP Tue Jul 12 10:30:17 UTC 2022 x86_64 x86_64 x86_64 GNU/Linux
aryan@aryan-VirtualBox:~$ uname -s
Linux
aryan@aryan-VirtualBox:~$ uname -n
aryan-VirtualBox
aryan@aryan-VirtualBox:~$ uname -v
#46-Ubuntu SMP Tue Jul 12 10:30:17 UTC 2022
aryan@aryan-VirtualBox:~$ uname -m
x86_64
aryan@aryan-VirtualBox:~$ uname -r
5.15.0-43-generic
aryan@aryan-VirtualBox:~$ uname -p
x86_64
aryan@aryan-VirtualBox:~$ uname -i
x86_64
aryan@aryan-VirtualBox:~$ uname -i
x86_64
aryan@aryan-VirtualBox:~$ uname -o
GNU/Linux
aryan@aryan-VirtualBox:~$ uname --i
uname: unrecognized option '--i'
Try 'uname --help' for more information.
aryan@aryan-VirtualBox:~$
```

uptime

- o Uptime Command In Linux: It is used to find out how long the system is active (running). This command returns set of values that involve, the current time, and the amount of time system is in running state, number of users currently logged into, and the load time for the past 1, 5 and 15 minutes respectively.

- o Syntax:

uptime [-options]

- o Options:

- -p, --pretty show uptime in pretty format
- -h, --help display this help and exit
- -s, --since system up since
- -V, --version output version information and exit

```
aryan@aryan-VirtualBox:~$ uptime
18:13:57 up 15 min,  1 user,  load average: 0.20, 0.06, 0.05
```

logname

- o The Linux logname command is a simple utility that is part of the GNU Core Utilities. It has a single purpose, to print the name of the current user. There are no options and the command takes no arguments. You simply call it and it prints the current users login name.
- o Syntax:

\$ logname

```
aryan@aryan-VirtualBox:~$ logname  
aryan
```

Calender

- o If a user wants a quick view of the calendar in the Linux terminal, cal is the command for you. By default, the cal command shows the current month calendar as output.
- o cal command is a calendar command in Linux which is used to see the calendar of a specific month or a whole year.
- o Syntax:

cal [[month] year]

```
> cal
    December 2022
Su Mo Tu We Th Fr Sa
                1  2  3
 4  5  6  7  8  9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30 31

~ -----
> █
```


- **date**

- o date command is used to display the system date and time. date command is also used to set date and time of the system. By default the date command displays the date in the time zone on which unix/linux operating system is configured. You must be the super-user (root) to change the date and time.

- o Syntax:

`date [OPTION]... [+FORMAT]`

`date [-u|--utc|--universal] [MMDDhhmm[[CC]YY][.ss]]`

```
aryan@aryan-VirtualBox:~$ date
Sunday 11 December 2022 06:16:13 PM IST
aryan@aryan-VirtualBox:~$ date -u
Sunday 11 December 2022 12:46:16 PM UTC
aryan@aryan-VirtualBox:~$
```

- **hostname**

- o hostname command in Linux is used to obtain the DNS(Domain Name System) name and set the system's hostname or NIS(Network Information System) domain name. A hostname is a name which is given to a computer and it attached to the network. Its main purpose is to uniquely identify over a network.
- o Syntax :

hostname -[option] [file]

```
aryan@aryan-VirtualBox:~$ hostname  
aryan-VirtualBox
```

- **tty**

- o The tty command of terminal basically prints the file name of the terminal connected to standard input. tty is short of teletype, but popularly known as a terminal it allows you to interact with the system by passing on the data (you input) to the system, and displaying the output produced by the system.

- o Syntax:

tty [OPTION]....

- o Options:

- -s, —silent, —quiet: Prints nothing, only returns an exit status.
- —help: It will display the help message and exit.
- —version: Prints the version information and exits.

```
aryan@aryan-VirtualBox:~$ tty  
/dev/pts/0
```

Lab 2

Directory command

- **pwd**

- o pwd stands for Print Working Directory. It prints the path of the working directory, starting from the root.

- o pwd is shell built-in command(pwd) or an actual binary(/bin/pwd).

- o Syntax:

pwd [-options]

- o Options:

- pwd -L: Prints the symbolic path.
 - pwd -P: Prints the actual path.

```
aryan@aryan-VirtualBox:~/Desktop$ pwd  
/home/aryan/Desktop
```

- **cd**

- o cd command in linux known as change directory command. It is used to change current working directory.

- o Syntax:

\$ cd [directory]

```
aryan@aryan-VirtualBox:~/Desktop$ cd Demo1
aryan@aryan-VirtualBox:~/Desktop/Demo1$ pwd
/home/aryan/Desktop/Demo1
```

- **mkdir**

- o mkdir command in Linux allows the user to create directories (also referred to as folders in some operating systems). This command can create multiple directories at once as well as set the permissions for the directories. It is important to note that the user executing this command must have enough permissions to create a directory in the parent directory, or he/she may receive a 'permission denied' error.

- o Syntax:

mkdir [options...] [directories ...]

- o Options:

- **-version:** It displays the version number, some information regarding the license and exits.
- **-help:** It displays the help related information and exits.
- **-v or -verbose:** It displays a message for every directory created.
- **-p:** A flag which enables the command to create parent directories as necessary. If the directories exist, no error is specified.
- **-m:** This option is used to set the file modes, i.e, permissions, etc. for the created directories

```
aryan@aryan-VirtualBox:~/Desktop/Demo1$ cd ..  
aryan@aryan-VirtualBox:~/Desktop$ mkdir Demo3  
aryan@aryan-VirtualBox:~/Desktop$ mkdir Demo4
```

- **rmdir**

- o rmdir command is used to remove empty directories from the filesystem in Linux. The rmdir command removes each and every directory specified in the command line only if these directories are empty. So if the specified directory has some directories or files in it then this cannot be removed by rmdir command.

- o Syntax:

```
rmdir [-p] [-v | -verbose] [-ignore-fail-on-non-empty]  
directories ...
```

- o Options:

- -help: It will print the general syntax of the command along with the various options that can be used with the rmdir command as well as give a brief description about each option.
- rmdir -p: In this option each of the directory argument is treated as a pathname of which all components will be removed, if they are already empty, starting from the last component.
- rmdir -v, -verbose: This option displays verbose information for every directory being processed.
- rmdir -ignore-fail-on-non-empty: This option does not report a failure which occurs solely because a directory is non-empty. Normally, when rmdir is being instructed to remove a non-empty directory, it simply reports an error. This option consists of all those error messages.

- `rmdir -version`: This option is used to display the version information and exit.

```
aryan@aryan-VirtualBox:~/Desktop$ rmdir Demo4  
aryan@aryan-VirtualBox:~/Desktop$ rmdir Demo4  
rmdir: failed to remove 'Demo4': No such file or directory
```


Lab 3

File commands

- **cat**

- o Cat(concatenate) command is very frequently used in Linux. It reads data from the file and gives their content as output. It helps us to create, view, concatenate files. So let us see some frequently used cat commands.

- o Syntax:

`$ cat [OPTION] [FILE]...`

```
aryan@aryan-VirtualBox:~/Desktop/Demo1$ cat f00  
This is text file1
```

- **cp**

- o cp stands for copy. This command is used to copy files or group of files or directory. It creates an exact image of a file on a disk with different file name. cp command requires at least two filenames in its arguments.

- o Syntax:

cp [OPTION] Source Destination

[OPTION] Source Directory

**cp [OPTION] Source-1 Source-2 Source-3 Source-n
Directory**

```
aryan@aryan-VirtualBox:~/Desktop/Demo1$ cp f00 f11
aryan@aryan-VirtualBox:~/Desktop/Demo1$ cat f11
This is text file1
```

- **Copy**

- o cp stands for copy. This command is used to copy files or group of files or directory. It creates an exact image of a file on a disk with different file name. cp command requires at least two filenames in its arguments.

- o Syntax:

cp [OPTION] Source Destination

[OPTION] Source Directory

cp [OPTION] Source-1 Source-2 Source-3 Source-nDirectory

```
aryan@aryan-VirtualBox:~/Desktop/Demo1$ cp f00 f11
aryan@aryan-VirtualBox:~/Desktop/Demo1$ cat f11
This is text file1
```

- **rm**

- o rm stands for remove here. rm command is used to remove objects such as files, directories, symbolic links and so on from the file system like UNIX. To be more precise, rm removes references to objects from the filesystem, where those objects might have had multiple references (for example, a file with two different names). By default, it does not remove directories.

- o Syntax:

rm [OPTION]... FILE...

```
aryan@aryan-VirtualBox:~/Desktop/Demo1$ rm f11
aryan@aryan-VirtualBox:~/Desktop/Demo1$ ls
f00
```

- **wc**

- o wc stands for word count. As the name implies, it is mainly used for counting purpose.
- o It is used to find out number of lines, word count, byte and characters count in the files specified in the file arguments.
- o By default it displays four-columnar output.
- o First column shows number of lines present in a file specified, second column shows number of words present in the file, third column shows number of characters present in file and fourth column itself is the file name which are given as argument.
- o Syntax:

wc [OPTION]... [FILE]...

```
aryan@aryan-VirtualBox:~/Desktop/Demo1$ wc f00
1  4 19 f00
aryan@aryan-VirtualBox:~/Desktop/Demo1$ wc -l f00
1 f00
aryan@aryan-VirtualBox:~/Desktop/Demo1$ wc -w f00
4 f00
aryan@aryan-VirtualBox:~/Desktop/Demo1$
```

- **cmp**

- o cmp command in Linux/UNIX is used to compare the two files byte by byte and helps you to find out whether the two files are identical or not.
- o When cmp is used for comparison between two files, it reports the location of the first mismatch to the screen if difference is found and if no difference is found i.e the files compared are identical.
- o cmp displays no message and simply returns the prompt if the the files compared are identical.
- o Syntax:

cmp [OPTION]... FILE1 [FILE2 [SKIP1 [SKIP2]]]

```
aryan@aryan-VirtualBox:~/Desktop/Demo1$ cat f00
This is text file1
aryan@aryan-VirtualBox:~/Desktop/Demo1$ cat f11
This is test file 2

aryan@aryan-VirtualBox:~/Desktop/Demo1$ cmp f00 f11
f00 f11 differ: byte 11, line 1
```

- **diff**

- o diff stands for difference. This command is used to display the differences in the files by comparing the files line by line.

- Unlike its fellow members, cmp and comm, it tells us which lines in one file have to be changed to make the two files identical.

- o Special symbols are:

- o a : add

- o c : change

- o d : delete

- o Syntax :

diff [options] File1 File2

```
aryan@aryan-VirtualBox:~/Desktop/Demo1$ diff f00 f11
1c1,2
< This is text file1
---
> This is test file 2
>
```

- **comm**

- o comm compare two sorted files line by line and write to standard output; the lines that are common and the lines that are unique.

- o Syntax:

`$comm [OPTION]... FILE1 FILE2`

- o Options for comm command:

- -1 :suppress first column(lines unique to first file).
 - -2 :suppress second column(lines unique to second file).
 - -3 :suppress third column(lines common to both files).
 - – -check-order :check that the input is correctly sorted, even if all input lines are pairable.
 - – -nocheck-order :do not check that the input is correctly sorted.
 - – -output-delimiter=STR :separate columns with string STR
 - – -help :display a help message, and exit.
 - – -version :output version information, and exit.

```
aryan@aryan-VirtualBox:~/Desktop/Demo1$ comm f00 f11
      This is text file
This is text file 1
```


Lab 4

List command

- **ls**

- o ls is a Linux shell command that lists directory contents of files and directories

- o Syntax

ls

```
aryan@aryan-VirtualBox:~/Desktop/Demo1$ ls
f00  f11
aryan@aryan-VirtualBox:~/Desktop/Demo1$ ls -a
.  ..  f00  f11
aryan@aryan-VirtualBox:~/Desktop/Demo1$ ls -l
total 8
-rw-rw-r-- 1 aryan aryan 20 Dec 11 18:31 f00
-rw-rw-r-- 1 aryan aryan 18 Dec 11 18:31 f11
aryan@aryan-VirtualBox:~/Desktop/Demo1$ ls -x
f00  f11
```

- **ls -a**

- o To show all the hidden files in the directory, use '-a option'.

- o Syntax:

- ls -a**

- **ls -l**

- o Will get the details of directories content.

- o Syntax:

- ls -l**

Lab 5

Permissions

- **chmod**

The chmod command is used to change the access mode of a file.

Syntax : chmod [reference][operator][mode] file...

Reference	Class	Description
u	owner	File's owner
g	group	Users who are the member of the file's group
o	others	Users who are neither the file's owner nor members of the file's group
a	all	All three of the above

Operator	Description
+	Adds the specified modes to the specified classes
-	Removes the specified modes from the specified classes
=	The modes specified are to be made the exact modes for the specified classes

r	Permission to read the file
w	Permission to write (or delete) the file
x	Permission to execute the file, or, in the case of a director search it

```
aryan@aryan-VirtualBox:~/Desktop$ cd Demo1
aryan@aryan-VirtualBox:~/Desktop/Demo1$ touch f00
aryan@aryan-VirtualBox:~/Desktop/Demo1$ ls -l f00
-rw-rw-r-- 1 aryan aryan 20 Dec 11 18:34 f00
aryan@aryan-VirtualBox:~/Desktop/Demo1$ chmod u+x f00
aryan@aryan-VirtualBox:~/Desktop/Demo1$ ls -l f00
-rwxrw-r-- 1 aryan aryan 20 Dec 11 18:34 f00
aryan@aryan-VirtualBox:~/Desktop/Demo1$
```

- **chgrp**

chgrp command in Linux is used to change the group ownership of a file or directory. All files in Linux belong to an owner and a group.

Syntax : chgrp [OPTION]... GROUP FILE...
 chgrp [OPTION]... -reference=RFILE
 FILE...

```
aryan@aryan-VirtualBox:~/Desktop/Demo1$ ls -l f00
-rwxrw-r-- 1 aryan aryan 20 Dec 11 18:34 f00
aryan@aryan-VirtualBox:~/Desktop/Demo1$ sudo addgroup linux_file
[sudo] password for aryan:
Adding group `linux_file' (GID 1001) ...
Done.
aryan@aryan-VirtualBox:~/Desktop/Demo1$ sudo chgrp linux_file f00
aryan@aryan-VirtualBox:~/Desktop/Demo1$ ls -l f00
-rwxrw-r-- 1 aryan linux_file 20 Dec 11 18:34 f00
aryan@aryan-VirtualBox:~/Desktop/Demo1$
```

- **chown**

The chown command allows you to change the user and/or group ownership of a given file, directory, or symbolic link.

In Linux, all files are associated with an owner and a group and assigned with permission access rights for the file owner, the group members, and others

Syntax : chown [OPTION]... [OWNER][:[GROUP]] FILE...
 chown [OPTION]... --reference=RFILE
 FILE...

Types of file Permissions:

- User: These types of file permissions affect the owner of the file.
- Group: These types of file permissions affect the group which owns the file. Instead of the group permissions, the user permissions will apply if the owner user is in this group.
- Other: These types of file permissions affect all other users on the system.

```
aryan@aryan-VirtualBox:~/Desktop/Demo1$ chown --help
Usage: chown [OPTION]... [OWNER][:[GROUP]] FILE...
       or: chown [OPTION]... --reference=RFILE FILE...
Change the owner and/or group of each FILE to OWNER and/or GROUP.
```

Lab 6

Process Commands

- **ps**

Linux provides us a utility called ps for viewing information related with the processes on a system which stands as abbreviation for “Process Status”. ps command is used to list the currently running processes and their PIDs along with some other information depends on different options. It reads the process information from the virtual files in /proc file-system. /proc contains virtual files, this is the reason it's referred to as a virtual file system.

Syntax: ps [options]

Options

-A, -e	All except the session leaders
-a	All with tty, except session leaders.
A	All with tty including other users.
-d	All except sessions leaders
-N, --deselect	Negate selection
r	Only running processes
T	All processes on this terminal

x

Processes without controlling ttysps

```
aryan@aryan-VirtualBox:~$ ps
  PID TTY          TIME CMD
 4764 pts/0        00:00:00 bash
 4772 pts/0        00:00:00 ps
```

```
aryan@aryan-VirtualBox:~$ ps x
  PID TTY          STAT TIME   COMMAND
   769 ?            Ss      0:00   /lib/systemd/systemd --user
   774 ?            S        0:00   (sd-pam)
   824 ?          S<sl     0:00   /usr/bin/pipewire
   825 ?          Ssl      0:00   /usr/bin/pipewire-media-session
   828 ?          S<sl     0:00   /usr/bin/pulseaudio --daemonize=no --log-target=jou
   830 ?          Sll      0:00   /usr/bin/gnome-keyring-daemon --daemonize --login
   840 tty2        Ssl+     0:00   /usr/libexec/gdm-wayland-session env GNOME_SHELL_SE
   842 ?          Ss       0:00   /usr/bin/dbus-daemon --session --address=systemd: -
   846 tty2        Sl+      0:00   /usr/libexec/gnome-session-binary --session=ubuntu
   847 ?          Ssl      0:00   /usr/libexec/gvfsd
   859 ?          Sl       0:00   /usr/libexec/gvfsd-fuse /run/user/1000/gvfs -f
   909 ?          Ssl      0:00   /usr/libexec/gnome-session-ctl --monitor
   943 ?          Ssl      0:00   /usr/libexec/gnome-session-binary --systemd-service
  958 ?          Ssl      2:01   /usr/bin/gnome-shell
```

```
aryan@aryan-VirtualBox:~$ ps a
  PID TTY          STAT TIME   COMMAND
   840 tty2        Ssl+     0:00   /usr/libexec/gdm-wayland-session env GNOME_SHELL_SE
   846 tty2        Sl+      0:00   /usr/libexec/gnome-session-binary --session=ubuntu
  4764 pts/0        Ss       0:00   bash
  4777 pts/0        R+       0:00   ps a
```


- **top**

top command is used to show the Linux processes. It provides a dynamic real-time view of the running system. Usually, this command shows the summary information of the system and the list of processes or threads which are currently managed by the Linux Kernel.

Syntax : top [options]

Pressing q will exit

```
top - 11:48:32 up 1:35, 1 user, load average: 0.00, 0.00, 0.00
Tasks: 178 total, 3 running, 175 sleeping, 0 stopped, 0 zombie
%Cpu(s): 7.7 us, 1.8 sy, 0.0 ni, 90.5 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 4868.0 total, 1018.1 free, 837.9 used, 3011.9 buff/cache
MiB Swap: 1070.0 total, 1070.0 free, 0.0 used, 3748.5 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
958	aryan	20	0	3814576	389120	133504	R	6.2	7.8	2:09.70	gnome-shell
4746	aryan	20	0	568820	50856	38440	R	3.1	1.0	0:01.67	gnome-terminal-
1	root	20	0	168004	13184	8212	S	0.0	0.3	0:02.85	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kthreadd
3	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	rcu_gp
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	rcu_par_gp
5	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	netns
7	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/0:0H-events_highpri
10	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	mm_percpu_wq
11	root	20	0	0	0	0	S	0.0	0.0	0:00.00	rcu_tasks_rude_
12	root	20	0	0	0	0	S	0.0	0.0	0:00.00	rcu_tasks_trace
13	root	20	0	0	0	0	S	0.0	0.0	0:00.88	ksoftirqd/0
14	root	20	0	0	0	0	I	0.0	0.0	0:01.10	rcu_sched
15	root	rt	0	0	0	0	S	0.0	0.0	0:00.03	migration/0
16	root	-51	0	0	0	0	S	0.0	0.0	0:00.00	idle inject/0

Highlight running process in top:

```
aryan@aryan-VirtualBox:~$ top -h
procs-ng 3.3.17
Usage:
top -hv | -bcEeHiOSs1 -d secs -n max -u|U user -p pid(s) -o field -w [cols]
```

- **bg**

bg command in linux is used to place mentioned foreground jobs in background.

Syntax : bg [job_spec...]

```
aryan@aryan-VirtualBox:~$ jobs
aryan@aryan-VirtualBox:~$ sleep 500

^^Z
[1]+  Stopped                  sleep 500
aryan@aryan-VirtualBox:~$ jobs
[1]+  Stopped                  sleep 500
aryan@aryan-VirtualBox:~$ bg %1
[1]+ sleep 500 &
aryan@aryan-VirtualBox:~$ jobs
[1]+  Running                  sleep 500 &
aryan@aryan-VirtualBox:~$ █
```

- fg

fg command in linux is used to place background jobs in foreground.

Syntax: fg [jobs_spec...]

```
aryan@aryan-VirtualBox:~$ jobs
aryan@aryan-VirtualBox:~$ sleep 500

^^Z
[1]+  Stopped                  sleep 500
aryan@aryan-VirtualBox:~$ jobs
[1]+  Stopped                  sleep 500
aryan@aryan-VirtualBox:~$ bg %1
[1]+ sleep 500 &
aryan@aryan-VirtualBox:~$ jobs
[1]+  Running                  sleep 500 &
aryan@aryan-VirtualBox:~$ fg %1
sleep 500
```

- **nice**

nice command in Linux helps in execution of a program/process with modified scheduling priority.

Syntax : nice [Option][Command][ARG]...

```
aryan@aryan-VirtualBox:~$ ps -el | grep terminal
0 S  1000    5006    769  1  80   0 - 142535 do_pol ?        00:00:00 gnome-terminal-
aryan@aryan-VirtualBox:~$ nice -10 gnome-terminal
```

As soon as we change priority of terminal, a new terminal opens

- **renice:**

Renice command is used to change priority of running process using id

Syntax: renice -n [priority] [PID]

```
aryan@aryan-VirtualBox:~$ renice -n 15 -p 5006
5006 (process ID) old priority 0, new priority 15
aryan@aryan-VirtualBox:~$ renice -n -g 4
renice: invalid priority '-g'
Try 'renice --help' for more information.
```

- **Kill:** *kill* command in Linux (located in /bin/kill), is a built-in command which is used to terminate processes manually. *kill* command sends a signal to a process which terminates the process.

Syntax : kill [PID]

```
aryan@aryan-VirtualBox:~$ ps
  PID TTY          TIME CMD
  5024 pts/0        00:00:00 bash
  5176 pts/0        00:00:00 ps
aryan@aryan-VirtualBox:~$ kill 5176
bash: kill: (5176) - No such process
```

Lab 7

Grep Commands

GREP stands for Global search for regular expression and print out.

The grep filter searches a file for a particular pattern of characters, and display all lines that contain that pattern. The pattern that is searched in the files is referred to as the regular expression.

Syntax: grep [options] pattern [files]

Some Options:

- c: This prints only a count of the lines that match a pattern
- h: Display the matched lines, but do not display the filenames.
- i: Ignores, case for matching
- l: Displays list of a filenames only.
- n: Display the matched lines and their line numbers.
- v: This prints out all the lines that do not matches the pattern
- e **exp** : Specifies expression with this option. Can use multiple times.
- f **file** : Takes patterns from file, one per line.
- E: Treats pattern as an extended regular expression (ERE)
- w: Match whole word
- o: Print only the matched parts of a matching line, with each such part on a separate output line.
- A **n**: Prints searched line and n lines after the result.
- B **n**: Prints searched line and n line before the result.
- C **n**: Prints searched line and n lines after before the result.

Demo text file:

```
aryan@aryan-VirtualBox:~$ cat grepDemo.txt
Linux is great OS. Linux was developed in 1991 by Linus Torvalds.
Linux is open source operating system based on C language.
It is more stable than windows
Linux security feature is the main reason for it being used globally.
```

1. -i: Case insensitive Search

```
aryan@aryan-VirtualBox:~$ grep -i LiNuX grepDemo.txt
Linux is great OS. Linux was developed in 1991 by Linus Torvalds.
Linux is open source operating system based on C language.
Linux security feature is the main reason for it being used globally.
```

2. -c: Count number of matches: (It is case sensitive)

```
aryan@aryan-VirtualBox:~$ grep -c LiNuX grepDemo.txt
0
aryan@aryan-VirtualBox:~$ grep -c Linux grepDemo.txt
3
```

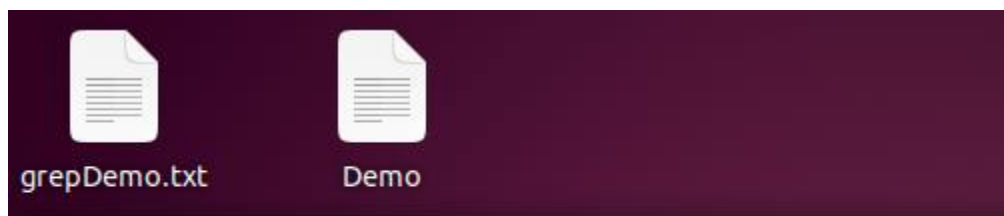
3. -n: Show line number while displaying the output

```
aryan@aryan-VirtualBox:~$ grep -n is grepDemo.txt
1:Linux is great OS. Linux was developed in 1991 by Linus Torvalds.
2:Linux is open source operating system based on C language.
3:It is more stable than windows
4:Linux security feature is the main reason for it being used globally.
```

4. -v: Inverting the pattern match: (Display the lines that are not matched with the searched pattern)

```
aryan@aryan-VirtualBox:~$ grep -v "open" grepDemo.txt
Linux is great OS. Linux was developed in 1991 by Linus Torvalds.
It is more stable than windows
Linux security feature is the main reason for it being used globally.
aryan@aryan-VirtualBox:~$ grep -v "Linux" grepDemo.txt
It is more stable than windows
```

5. -l: Display the files names that match the pattern.



```
aryan@aryan-VirtualBox: ~/Desktop
aryan@aryan-VirtualBox:~/Desktop$ grep -l "Linux" *
Demo
grepDemo.txt
```

Lab 8

Word Count

There may be some scenarios where one needs to keep track of the number of lines and number of words in a particular file. In that scenario, any of the following methods can be used to count the words , lines in the text file.

1. wc Command:

Syntax: wc [options] filename

```
aryan@aryan-VirtualBox:~/Desktop$ cat Demo
Linux is different from other OS. Different parts of Linux are developed by different organizations.

Different parts include kernel, shell utilities, X server, system environment, graphical programs, etc.
If you want you can access the codes of all these parts and assemble them yourself. But its not an easy
task seeking a lot of time and all the parts has to be assembled correctly in order to work properly.

From here on distribution (also called as distros) comes into the picture. They assemble all these parts
for us and give us a compiled operating system of Linux to install and use.
aryan@aryan-VirtualBox:~/Desktop$ wc Demo
5 101 594 Demo
```

```
aryan@aryan-VirtualBox:~/Desktop$ wc --lines Demo
5 Demo
aryan@aryan-VirtualBox:~/Desktop$ wc --words Demo
101 Demo
```

2. Shell script to execute word count command for a file.

```
#!/bin/bash

file_path="/home/aryan/Desktop/Demo"

number_of_lines=`wc --lines < $file_path`

number_of_words=`wc --words < $file_path`

echo "Number of lines: $number_of_lines"
echo "Number of words: $number_of_words"
```

```
aryan@aryan-VirtualBox:~/Desktop$ touch wcSpt.sh
aryan@aryan-VirtualBox:~/Desktop$ chmod u+x wcSpt.sh
aryan@aryan-VirtualBox:~/Desktop$ ./wcSpt.sh
Number of lines: 5
Number of words: 101
```

Lab 9

Wildcard Characters in Linux

Wildcards (also referred to as meta characters) are symbols or special characters that represent other characters. You can use them with any command such as ls command or rm command to list or remove files matching a given criteria, receptively.

These wildcards are interpreted by the shell and the results are returned to the command you run. There are three main wildcards in Linux:

1. An asterisk (*) – matches one or more occurrences of any character, including no character.
2. Question mark (?) – represents or matches a single occurrence of any character.
3. Bracketed characters ([]) – matches any occurrence of character enclosed in the square brackets. It is possible to use different types of characters (alphanumeric characters): numbers, letters, other special characters etc.

```
aryan@aryan-VirtualBox:~$ ls -x
demo.sh      Desktop      Documents    Downloads    Music        Pictures      prac10.sh
prac12.sh    prac13.sh    prac14.sh    prac15.sh    prac16.sh    prac17        prac17.c
prac18        prac18.c     prac18.sh    Public       snap         Templates     Videos
aryan@aryan-VirtualBox:~$ ls -l p*
-rwxrwxr-- 1 aryan aryan 141 Dec 30 13:41 prac10.sh
-rwxrwxr-- 1 aryan aryan 130 Dec 30 13:53 prac12.sh
-rwxrwxr-- 1 aryan aryan 174 Dec 30 14:21 prac13.sh
-rwxrwxr-x 1 aryan aryan 127 Dec 30 14:36 prac14.sh
-rwxrwxr-x 1 aryan aryan 52 Dec 30 14:40 prac15.sh
-rwxrwxr-x 1 aryan aryan 139 Dec 30 15:03 prac16.sh
-rwxrwxr-x 1 aryan aryan 16104 Dec 30 16:36 prac17
-rw-rw-r-- 1 aryan aryan 1161 Dec 30 16:39 prac17.c
-rwxrwxr-x 1 aryan aryan 16000 Dec 30 16:49 prac18
-rw-rw-r-- 1 aryan aryan 367 Dec 30 16:49 prac18.c
-rw-rw-r-- 1 aryan aryan 367 Dec 30 16:48 prac18.sh
aryan@aryan-VirtualBox:~$ ls -l prac1?.sh
-rwxrwxr-- 1 aryan aryan 141 Dec 30 13:41 prac10.sh
-rwxrwxr-- 1 aryan aryan 130 Dec 30 13:53 prac12.sh
-rwxrwxr-- 1 aryan aryan 174 Dec 30 14:21 prac13.sh
-rwxrwxr-x 1 aryan aryan 127 Dec 30 14:36 prac14.sh
-rwxrwxr-x 1 aryan aryan 52 Dec 30 14:40 prac15.sh
-rwxrwxr-x 1 aryan aryan 139 Dec 30 15:03 prac16.sh
-rw-rw-r-- 1 aryan aryan 367 Dec 30 16:48 prac18.sh
aryan@aryan-VirtualBox:~$
```

The ls -l p* command matches all files with names starting with p (which is the prefix) and ending with one or more occurrences of any character.

The ls -l prac1?.sh command matches all files with names beginning with prac1 followed by any single character and ending with .sh

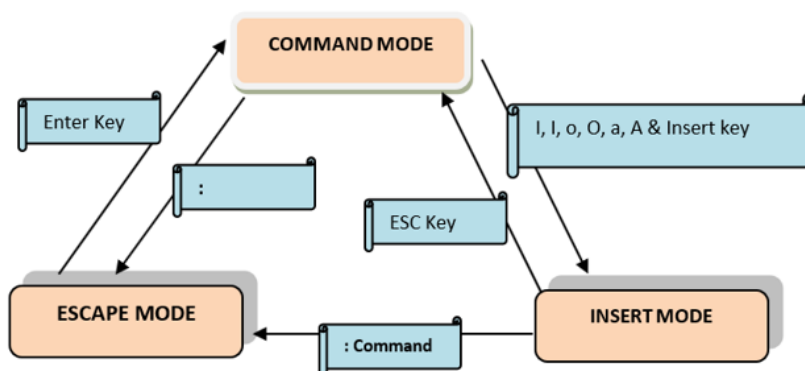
Lab 10

VI Editor

VI Editor: The vi editor is elaborated as **visual** editor. It is installed in every Unix system. In other words, it is available in all Linux distros. It is user-friendly and works same on different distros and platforms. It is a very powerful application. An improved version of vi editor is **vim**.

The vi-editor tool is an interactive tool as it displays changes made in the file on the screen while you edit the file. In vi editor you can insert, edit or remove a word as cursor moves throughout the file.

Modes of VI Editor:



While working with the vi editor, we usually come across the following modes:

- **Command mode:** This mode enables you to perform administrative tasks such as saving the files, executing the commands etc. In this mode, whatever you type is interpreted as a command.
- **Insert mode:** This mode enables you to insert text into the file. Everything that's typed in this mode is interpreted as input and placed in the file.
- VI always starts in the command mode. To enter text, you must be in the insert mode for which simply type
- To come out of the insert mode, press the Esc key, which will take you back to the command mode.

Following are some basic commands to use vi editor:

- vi [filename]: Creates a new file if does not exists already, otherwise opens the existing file.
- vi -R: Opens a file in read-only mode.

A tilde (~) on each line represents an unused line. If a line does not begin with a tilde and appears to be blank, there is a space, tab, newline, or some other non-viewable character present.

VI editing commands:

- i : Insert at current cursor position.
- a : Write after cursor.
- A : Write at end of line.
- ESC : Terminate insert mode.
- u : Undo last change.
- U : Undo all changes to the entire line.
- o : Open a new line (goes into insert mode).
- dd : Delete line.
- D : Delete contents of line after the cursor
- C : Delete contents of a line after the cursor and insert new text.
- dw : Delete word.
- cw : Change word.
- x : Delete character at the cursor.
- r : Replace character

Moving within a file:

- k : Move cursor up.
- j : Move cursor down.
- h : Move cursor left.
- l : Move cursor right.

Saving & Closing file:

- Shift+zz : Save the file and quit.
- :w : Save the file but keep it open.
- :q : Quit without saving.
- :wq : Save the file and quit

Lab 11

WSS to enter two strings

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

```
# Prompt the user to enter the first string
```

```
echo "Enter the first string: "
```

```
read string1
```

```
# Prompt the user to enter the second string
```

```
echo "Enter the second string: "
```

```
read string2
```

```
echo "The first string is: $string1"
```

```
echo "The second string is: $string2"
```

```
aryan@aryan-VirtualBox:~$ vi prac10.sh
aryan@aryan-VirtualBox:~$ ./prac10.sh
Enter 1st string:
Hi
Enter 2nd string:
I am Aryan
1st string is: Hi
2nd string is: I am Aryan
```

Lab 12

WSS to add, subtract, multiply and divide two numbers.

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

```
#Taking the inputs from user through prompt
```

```
read -p "Enter the first number: " num1
```

```
read -p "Enter the second number: " num2
```

```
#using expr and read to take operator
```

```
read -p "Enter expression: " expr
```

```
#Using bc for calculation
```

```
echo "The result is: " $num1$expr$num2 | bc
```

```
aryan@aryan-VirtualBox:~$ touch prac12.sh
aryan@aryan-VirtualBox:~$ chmod u+x prac12.sh
aryan@aryan-VirtualBox:~$ vi prac12.sh
aryan@aryan-VirtualBox:~$ ./prac12.sh
Enter 1st number: 12
Enter 2nd number: 22
Enter EXpression: +
34
aryan@aryan-VirtualBox:~$ ./prac12.sh
Enter 1st number: 12
Enter 2nd number: 22
Enter EXpression: *
264
aryan@aryan-VirtualBox:~$ ./prac12.sh
Enter 1st number: 12
Enter 2nd number: 22
Enter EXpression: -
-10
aryan@aryan-VirtualBox:~$ ./prac12.sh
Enter 1st number: 12
Enter 2nd number: 22
Enter EXpression: /
0
```

Lab 13

WSS to find largest of two number

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

```
read -p 'Enter the first number: ' num1  
read -p 'Enter the second number: ' num2
```

```
if [ $num1 -gt $num2]  
then  
    echo 'The larger number is: ' $num1  
else  
    echo 'Then larger number is: ' $num2  
fi
```

```
aryan@aryan-VirtualBox:~$ vi prac13.sh  
aryan@aryan-VirtualBox:~$ ./prac13.sh  
Enter 1st number:45  
Enter 2nd Number: 55  
55 is greater than 45
```

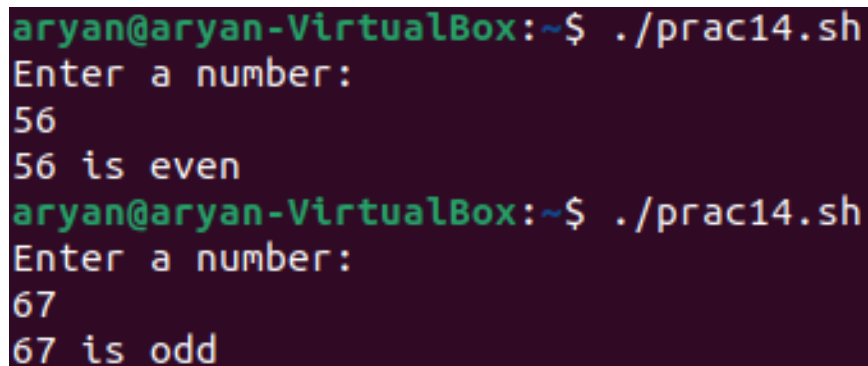
Lab 14

WSS to Check whether given number is even or odd.

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

```
# Prompt the user to enter a number  
echo "Enter a number: "  
read number
```

```
# Check if the number is even or odd  
if [ $((($number % 2)) -eq 0 ]  
then  
    echo "$number is even"  
else  
    echo "$number is odd"  
fi
```



```
aryan@aryan-VirtualBox:~$ ./prac14.sh  
Enter a number:  
56  
56 is even  
aryan@aryan-VirtualBox:~$ ./prac14.sh  
Enter a number:  
67  
67 is odd
```

Lab 15

WSS to list all the directory files in a directory

Source code:

```
aryan@aryan-VirtualBox: ~  
echo "ls command can be run by shell script"  
  
ls -l  
~  
~  
~  
~  
~
```

Output:

```
aryan@aryan-VirtualBox:~$ vi prac15.sh  
aryan@aryan-VirtualBox:~$ chmod +x prac15.sh  
aryan@aryan-VirtualBox:~$ ./prac15.sh  
ls command can be run by shell script  
total 56  
-rwxrw-r-- 1 aryan aryan    0 Dec 30 13:33 demo.sh  
drwxr-xr-x 2 aryan aryan 4096 Dec 30 13:30 Desktop  
drwxr-xr-x 2 aryan aryan 4096 Dec 30 13:30 Documents  
drwxr-xr-x 2 aryan aryan 4096 Dec 30 13:30 Downloads  
drwxr-xr-x 2 aryan aryan 4096 Dec 30 13:30 Music  
drwxr-xr-x 2 aryan aryan 4096 Dec 30 13:30 Pictures  
-rwxrw-r-- 1 aryan aryan  141 Dec 30 13:41 prac10.sh  
-rwxrw-r-- 1 aryan aryan  130 Dec 30 13:53 prac12.sh  
-rwxrw-r-- 1 aryan aryan  174 Dec 30 14:21 prac13.sh  
-rwxrwxr-x 1 aryan aryan  127 Dec 30 14:36 prac14.sh  
-rwxrwxr-x 1 aryan aryan   52 Dec 30 14:40 prac15.sh  
drwxr-xr-x 2 aryan aryan 4096 Dec 30 13:30 Public  
drwx----- 3 aryan aryan 4096 Dec 30 13:30 snap  
drwxr-xr-x 2 aryan aryan 4096 Dec 30 13:30 Templates  
drwxr-xr-x 2 aryan aryan 4096 Dec 30 13:30 Videos
```

Lab 16

WSS to find factorial of a number

Source Code:

```
aryan@aryan-VirtualBox: ~  
read -p "Enter the number: " num  
  
fact = 1  
while [ $num -gt 1 ]  
do  
    fact=$((fact*num))  
    num=$((num-1))  
done  
  
echo "The factorial is: " $fact  
~  
~  
~
```

Output:

```
aryan@aryan-VirtualBox:~$ vi prac16.sh  
aryan@aryan-VirtualBox:~$ chmod +x prac16.sh  
aryan@aryan-VirtualBox:~$ ./prac16.sh  
Enter the number: 5  
The factorial is: 120
```


Lab 17

Write a c program to design a calculator

Source Code:

```
#include<stdio.h>

int main()
{
    int choice,flag=1;
    while(flag)
    {
        int num1,num2;
        printf("Enter first number: ");
        scanf("%d", &num1);
        printf("\n---Menu---\n");
        printf("1. Add\n");
        printf("2. Subtract\n");
        printf("3. Multiply\n");
        printf("4. Divide\n");
        printf("5. Exit\n");
        printf("Enter Choice: ");
        scanf("%d", &choice);
        switch(choice)
        {
            case 1: printf("Enter second number: ");
                    scanf("%d", &num2);
                    num1+=num2;
                    break;
            case 2: printf("Enter second number: ");
                    scanf("%d", &num2);
                    num1-=num2;
                    break;
            case 3: printf("Enter second number: ");
                    scanf("%d", &num2);
                    num1*=num2;
                    break;
            case 4: printf("Enter second number: ");
                    scanf("%d", &num2);
                    while(num2 == 0)
                    {
                        printf("Cannot divide by 0\n");
                        printf("Enter second number: ");
                        scanf("%d", &num2);
                    }
                    num1/=num2;
                    break;
            case 5: flag=0;
                    break;
            default: printf("Wrong Choice\n");
        }
        printf("The result is: %d\n", num1);
    }
    return 0;
}
```

Output:

```
aryan@aryan-VirtualBox:~$ gcc prac17.c -o prac17
aryan@aryan-VirtualBox:~$ ./prac17
```

```
Enter first number: 54
```

```
----Menu----
```

1. Add
2. Subtract
3. Multiply
4. Divide
5. Exit

```
Enter Choice: 3
```

```
Enter second number: 7
```

```
The result is: 378
```

```
Enter first number: 23
```

```
----Menu----
```

1. Add
2. Subtract
3. Multiply
4. Divide
5. Exit

```
Enter Choice: 4
```

```
Enter second number: 2
```

```
The result is: 11
```

```
Enter first number: 
```

Lab 18

Write a c program to create a child process and allow the parent to display “parent” and the child to display “child” on the screen.?

Source Code:

```
aryan@aryan-VirtualBox: ~  
#include <stdio.h>  
#include <unistd.h>  
#include <sys/types.h>  
#include <sys/wait.h>  
  
int main()  
{  
    pid_t pid;  
    //create a child process  
    pid = fork();  
  
    if (pid < 0){  
        //fork failed  
        printf( "Fork Failed.\n");  
        return 1;  
    }  
    else if (pid == 0){  
        //This is child process  
        printf("Child\n");  
    }else {  
        //This is parent process  
        printf("Parent\n");  
    }  
    return 0;  
}  
~  
~  
~
```

Output:

```
aryan@aryan-VirtualBox:~$ gcc prac18.c -o prac18  
aryan@aryan-VirtualBox:~$ ./prac18  
Parent  
aryan@aryan-VirtualBox:~$ Child
```