WINTER – 19 EXAMINATION

Subject Name: Open Source Operating System and Scripting Language

Model Answer

Subject Code: 22522

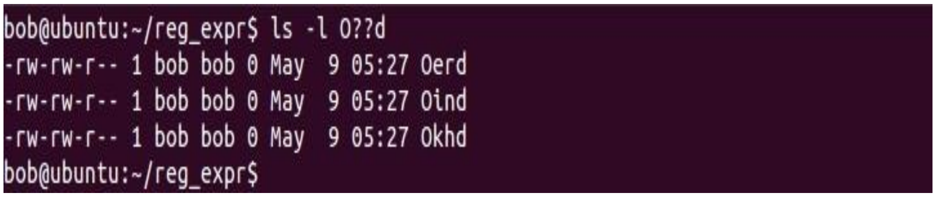
Important Instructions to examiners:

1) The answers should be examined by key words and not as word-to-word as given in the model answer scheme.
2) The model answer and the answer written by candidate may vary but the examiner may try to assess the understanding level of the candidate.
3) The language errors such as grammatical, spelling errors should not be given more Importance (Not applicable for subject English and Communication Skills.
4) While assessing figures, examiner may give credit for principal components indicated in the figure. The figures drawn by candidate and model answer may vary. The examiner may give credit for any equivalent figure drawn.
5) Credits may be given step wise for numerical problems. In some cases, the assumed constant values may vary and there may be some difference in the candidate's answers and model answer.
6) In case of some questions credit may be given by judgement on part of examiner of relevant answer based on candidate's understanding.
7) For programming language papers, credit may be given to any other program based on equivalent concept.

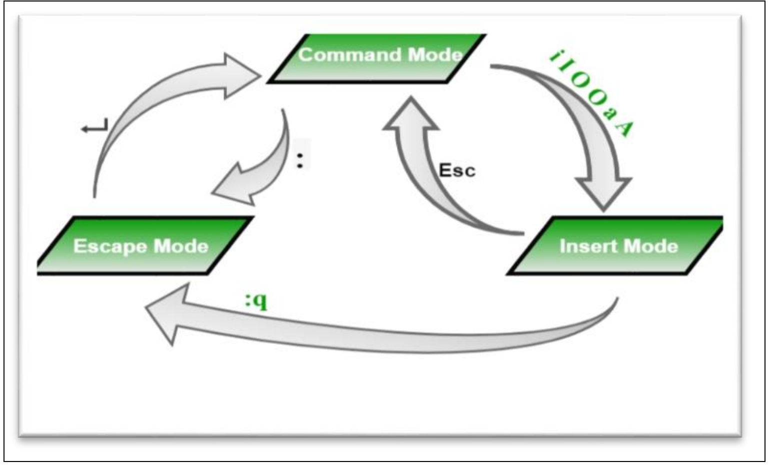
Q. No.	Sub Q. N.	Answer	Marking Scheme
1		**Attempt any FIVE of the following :**	**10 M**
	a	**State any two commands for managing disk space and also state their use**	**2 M**
	Ans	There are primarily two commands that can be used for managing Disk space: 1. df 2. du 1. df – disk filesystem • df command (short for disk filesystem) is used to show disk utilization for a Linux system. This reports the amount of disk space on a system. • To display information of device name, total blocks, total disk space, used disk space, available disk space and mount points on a file system. 2. du – disk usage • du command (short for disk usage) is useful command which is used to find disk usage for files & directories.	**Any two, 1M for one command with their use**

		• du command when used with various options provides results in many formats. This shows the amount of space used by specific files. • It is used to list the disk space used by files on a machine and crucial for disk space management on unix and linux systems.	
	b	**List any four features of open source operating system.**	**2 M**
	Ans	1. **Open Source:** open source OS code is freely available and it is community based development project. Multiple team's works in collaboration to enhance the capability of operating system and it is continuously evolving. 2. **Flexibility:** The software can be customized to meet specific business needs. Engineers can write more code to add an extra functionality and vice versa – delete unnecessary parts. 3. **Portable:** Portability means software can works on different types of hardware in same way. Linux kernel and application programs support their installation on any kind of hardware platform. 4. **Security:** Open source os provides user security using authentication features like password protection/ controlled access to specific files/ encryption of data. 5. **Lesser hardware costs:** Since Linux and open source solutions are easily portable and compressed, it takes lesser hardware power to carry out the same tasks when compared to the hardware power it takes on servers, such as, Solaris, Windows or workstations. With this less hardware power advantage, you can even use cheaper or older hardware and still get the desired results. 6. **High-quality software:** Open source software is mostly high-quality software. When you use the open source software, the source code is available. Most open source software are well-designed. Open source software can also be efficiently used in coding. These reasons make open source software an ideal choice for organizations. 7. **Simple license management:** When you use open source software, you would no longer need to worry about licenses. Open source software enables you to install it several times and also use it from any location. You will be free from monitoring, tracking or counting license compliance. 8. **Lower software costs:** Using open source software can help you minimize your expenses. You can save on licensing fees and maintenance fees. The only expenses that you would encounter would be expenditure for documentation, media and support. 9. **Abundant support:** You will get ample support when you use open source software. Open source support is mostly freely available and can be easily accessed through online communities. There are also	**Any four, ½ M each**

		many software companies that provide free online help and also varied levels of paid support. Most organization who create open source software solutions also provide maintenance and support. 10. **Scaling and consolidating:** Linux and open source software can be easily scaled. With varied options for clustering, load balancing and open source applications, such as email and database, you can enable your organization to either scale up and achieve higher growth or consolidate and achieve more with less. 11. **Application Support:** Linux has its own software repository from where users can download and install thousands of applications just by issuing a command in Linux Terminal or Shell. Linux can also run Windows applications if needed.	
	c	**Describe the use of wild card with example.**	**2 M**

	Ans	• A wildcard in Linux is a symbol or a set of symbols that stands in for other characters. It can be used to substitute for any other character or characters in a string.
		• Three types of wildcards are common in Linux:
		i. ? – matches a single character. For example, O??d matches anything that begins with O, ends with d and has two characters in between (like Oind, Okhd, Oerd, but not Oereed, Oad, Oerererd.)
		ii. * – matches any character or set of characters, including no character. For example, O*d matches anything that begins with O and ends with d (like Oind, Okhd, Oerd, Oereed, Oad, Oerererd, Od, Oarmeerrd). The number of characters in between O and d is not important.
		iii. Bracketed values – match characters enclosed in square brackets. For example, O[ac]d matches only Oad and Ocd. You can also specify a range of values: O[a-e]d matches Oad, Obd, Ocd, Odd and Oed.
		• For example, you can use a wildcard to get a list of all files in a directory that begin with the letter O.
		If we want to list all files that begin with **O,** end with **d** and have two characters in between, we can use the following syntax:
		
		`bob@ubuntu:~/reg_expr$ ls -l O??d`
		`-rw-rw-r-- 1 bob bob 0 May  9 05:27 Oerd`
		`-rw-rw-r-- 1 bob bob 0 May  9 05:27 Oind`
		`-rw-rw-r-- 1 bob bob 0 May  9 05:27 Okhd`
		`bob@ubuntu:~/reg_expr$`
			1 Command for 1 M

	d	**State modes of vi editor.**	**2 M**
	Ans	**Command Mode:** When vi starts up, it is in Command Mode. This mode is where vi interprets any characters we type as commands and thus does not display them in the window. This mode allows us to move through a file, and to delete, copy, or paste a piece of text. To enter into Command Mode from any other mode, it requires pressing	**Any two , 1M for each**

		the [Esc] key. If we press [Esc] when we are already in Command Mode, then vi will beep or flash the screen. **Insert mode:** This mode enables you to insert text into the file. Everything that's typed in this mode is interpreted as input and finally, it is put in the file. The vi always starts in command mode. To enter text, you must be in insert mode. To come in insert mode you simply type i. To get out of insert mode, press the Esc key, which will put you back into command mode. **Last Line Mode(Escape Mode):** Line Mode is invoked by typing a colon [:], while vi is in Command Mode. The cursor will jump to the last line of the screen and vi will wait for a command. This mode enables you to perform tasks such as saving files, executing commands.  Command Mode iIOOaA Insert Mode Esc : ↵ Escape Mode :q	
	e	**Define internal and external commands.**	**2 M**
	Ans	**Internal Commands:** Commands which are built into the shell. For all the shell built-in commands, execution of the same is fast in the sense that the shell doesn't have to search the given path for them in the PATH variable and also no process needs to be spawned for executing it. Examples: source, cd, fg etc. **External Commands:** Commands which aren't built into the shell. When an external command has to be executed, the shell looks for its path given in PATH variable and also a new process has to be spawned and the command gets executed. They are usually located in /bin or /usr/bin. For example, when you execute the "cat" command, which usually is at /usr/bin, the executable /usr/bin/cat gets executed. Examples: ls, cat etc.	**Internal Command:1M External Commands: 1M**
	f	**List role of administrator.**	**2 M**
	Ans	**1. Installing and configuring server** • A server is basically a computer program that facilitate the same computer or other computer by providing services to them. • It is most important element of Modern OS and network design.	**Any 4 , ½ M for each**

- It is of system administrator to configure server so that the most essential server remain inaccessible. He must be aware of types of attack and security bugs.

2. Installing and configuring application software

- In order to ensure a correct execution environment, administrator must provide software which is well configured and validate.
- He should ensure adequate memory allotment and resolve software failure and dependency issues.
- He must provide a set of activities to control hardware and software configuration and maintain policies for users.

3. Creating and maintaining user accounts

- User can access his own account but administrator has access to every user account.
- He can add, modify, delete or copy user account.
- He is responsible for maintaining security by providing role on a user account that define the level of access.

4. Backing up and restoring files

- To minimize the loss of data, administrator must maintain backup of files nd he should restore it whenever required.
- Administrator can take backup in removable media such as hard drives or tapes as protection against loss.
- Before creating backup administrator must decide.
  - What are necessary to backup?
  - How frequently backup should perform.

5. Monitoring and tuning performance

- Monitoring and tuning of performance is essential for Linux to work more efficiently.
- Administrator must identify system bottleneck and should solve them.
- Administrator can use system tools to increase performance, he can determine when hardware need to be upgrade.
- He should identify early sign of failure .

6. Configuring a secure system

- It is a duty of administrator to involve tasks and decisions to run secure Linux system and maintaining data integrity.
- It provide strong protection to individuals and corporate bodies and protecting parts of system even if it is under attack.
- Administrator should ensure
  - System has firewall.
  - Not allow connection from unknown network.
  - Not install software if not needed.

7. Using tools to monitor security

- Linux is the preferred operating system who demands secure networks, buy it can be easily crack by hackers.

		• It is important for administrator to be aware of tools hackers use and software used to monitor and counter such activity. • It is duty of administrator to prevent unauthorized use of his system.	
	g	**Write any two advantages and disadvantages of NIS. (Network Information Service)**	**2 M**
	Ans	**Advantages:** 1. Allows multiple computers to use the same files, so everyone on the network can access the same data 2. Reduces storage costs by having computers share applications instead of needing local disk space for each user application 3. Provides data consistency and reliability because all users can read the same set of files 4. Makes mounting of file systems transparent to users 5. Makes accessing remote files transparent to users 6. Supports heterogeneous environments 7. Reduces system administration overhead **Disadvantages:** 1. It is not a secure mechanism for providing authentication and authorization services. 2. If NIS clients use the broadcast service to locate NIS servers on the network, intruders can easily introduce their own NIS server with their own privileged accounts. Once a client binds to the rogue NIS server, the intruder can gain access to that client and perform unauthorized operations. 3. The NIS server's only security policy is the securenets setting. The securenets setting identifies which NIS clients to accept queries from. If an intruder impersonates a client that the securenets setting allows the NIS server to accept, he can download all of the NIS data. Even if an intruder fails the securenets test, he could potentially inspect all of the NIS requests and decode the data to gain access. 4. If NIS is used for authentication, password hashes are sent around the network in clear text and can be easily captured and cracked, making client systems vulnerable.	**Two Advantages: 1M** **Two Disadvantages: 1 M**
2		**Attempt any THREE of the following :**	**12 M**
	a	**With suitable example, explain CpiO and tar commands.**	**4 M**
	Ans	**cpio:** The **cpio command** is one of standard Unix backup utilities. It stands for "copy in/out." It is much less well known and more rarely used Unix utility in comparison with tar. • cpio works as a filter accepting standard input and writing to standard output. cpio allows you to copy files into and out of a cpio archive.	**2 M for each command example**

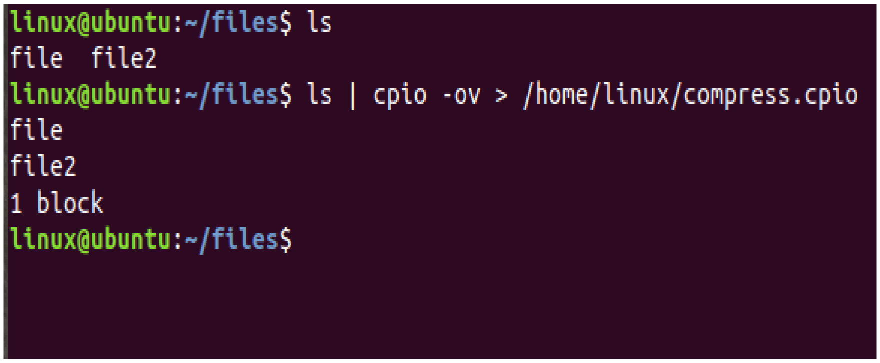
- The input to cpio is the list of files. That means that results of ls or find command can be piped directly into cpio. You can specify a device or file to which cpio will send its output, rather than sending it to stdout.
- Cpio uses two key options: -o (output) and –I (input) wither of which (but not both) must be there in the command line.

Examples:

To create a *.cpio file : We can create *.cpio files containing files and directory with the help of cpio command.

Syntax:

cpio -ov < name-list > archive
Here -ov is used as -o create the new archive and -v list the files processed.



```
linux@ubuntu:~/files$ ls
file  file2
linux@ubuntu:~/files$ ls | cpio -ov > /home/linux/compress.cpio
file
file2
1 block
linux@ubuntu:~/files$
```

Options of cpio are:

Option	**Description**
-d	Creates directories as and when needed.
-c	Writes header information in ASCII character form for portability.
-r	Renames files in interactive manner.
-t	Lists file in archive.
-u	Overwrites newer file with older version.
-v	Verbose option; prints lists of files that are being copied.
-m	Retains original file modification time.
-f exp	Copies all files except those in exp.
-Csize	Sets input-output block size to size bytes.
-A –O device	Appends files to device.
-H tar	Creates or reads a tar header format.

tar: For creating a disk archive that contains a group of files or an entire directory structure, we need to use tar.
The tar command was originally used to write files to a tape device for archiving. The **tar** program is used to create, modify, and access files archived in the **tar** format. "**tar**" stands for ***tape archive***.

Syntax: tar function [options] object1 object2 ...

Example: tar -cvf test.tar test1 test2
To create an archive, we need to specify the name of the archive (with –f), the copy or write operation (-c) and the filenames as arguments. The use of –v (verbose) option is to display the progress while tar works. The above command creates the file archive test.tar from the two uncompressed files test1 and test2.

The tar command functions:

Function	**Long name**	**Description**
-A	--concatenate	Append an existing tar archive file to another existing tar archive file.
-c	--create	Create a new tar archive file.
-d	--diff	Check the differences between a tar archive file and the filesystem.
	--delete	Delete from an existing tar archive file.
-r	--append	Append files to the end of an existing tar archive file.
-t	--list	List the contents of an existing tar archive file.
-u	--update	Append files to an existing tar archive file that are newer than a file with the same name in the existing archive.
-x	--extract	Extract files from an existing archive file.

b **Describe expr statement with example.** **4 M**

Ans

- The **expr** command is used to evaluate an arithmetic expression from the command line.
- The expr command can be used to perform arithmetic operations, comparison operations and string operations. The following table specify the way how to use the different operators with expr command in shell and their meaning. Most of the operators are escaped with backslash as these operators are shell built in operators and having specific meaning for example < and > are redirection operators.
- **Syntax:** expr operator value2

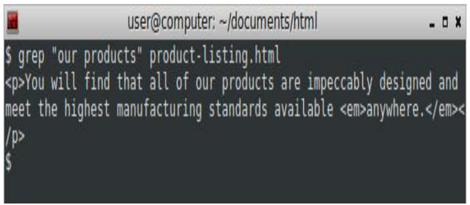
Explanation: 2 M,
Example: 2 M

Example :

```
$ expr 1 + 2
3
$ expr 5 \* 2
10
$
$ y= ‘expr 7+9’
$ echo $y
16
$ a=10; b=17
$ c=’expr $a * $b’
$ echo $c
170
$ expr 5 “*” 10
50
```

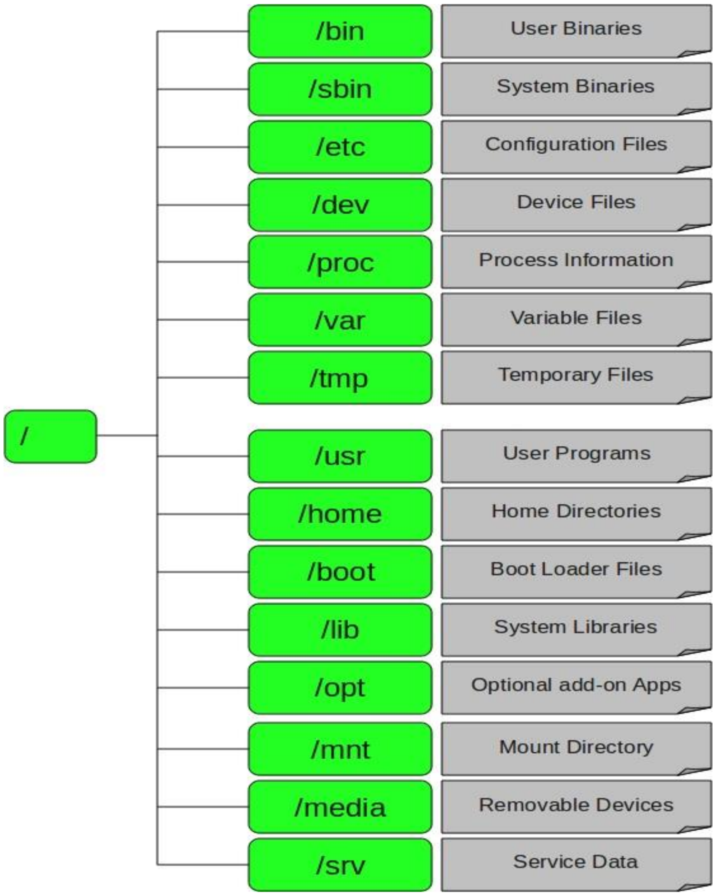
	c	**With suitable examples differentiate between grep and fgrep command.**	**4 M**
	Ans		**Any 4 points: 1 M each**

grep command	**fgrep command**
grep is an acronym that stands for "Global Regular Expressions Print".	fgrep is an acronym that stands for "Fixed-string Global Regular Expressions Print".
grep is a program which scans a specified file or files line by line, returning lines that contain a pattern. A pattern is an expression that specifies a set of strings by interpreting characters as meta-characters.	fgrep (which is the same as grep -F) is fixed or fast grep and behaves as grep but does NOT recognize any regular expression meta-characters as being special. The search will complete faster because it only processes a simple string rather than a complex pattern.
For example the asterisk meta character (*) is interpreted as meaning "zero or more of the preceding element". This enables users to type a short series of characters and meta characters into a grep command to have the computer show us what lines in which files match.	For example, if I wanted to search my .bash_profile for a literal dot (.) then using grep would be difficult because I would have to escape the dot because dot is a meta character that means 'wild-card, any single character':
Syntax: grep [OPTIONS] PATTERN [FILE….]	Syntax: fgrep [-b] [-c] [-h] [-i] [-1] [-n] [-s] [-v] [-x] [-e pattern_list] [-f pattern-file] [pattern] [file]
Example:	Example:

<table>
<tr>
<td></td>
<td></td>
<td>
<table>
<tr>
<td>


user@computer: ~/documents/html

<pre>
$ grep "our products" product-listing.html
<p>You will find that all of our products are impeccably designed and
meet the highest manufacturing standards available <em>anywhere.</em><
/p&gt;
$
</pre>
</td>
<td>fgrep “support” myfile.txt
Search for “support” in the myfile.txt</td>
</tr>
</table>
</td>
<td></td>
</tr>
<tr>
<td></td>
<td><b>d</b></td>
<td><b>Describe Hierarchical file system.</b></td>
<td><b>4 M</b></td>
</tr>
<tr>
<td></td>
<td><b>Ans</b></td>
<td>
All files in UNIX are “related” to one another. The file system in UNIX is a collection of all these related files organised in a hierarchical (an inverted tree) structure. This system has been adopted by MS-DOS and is visually represented by:



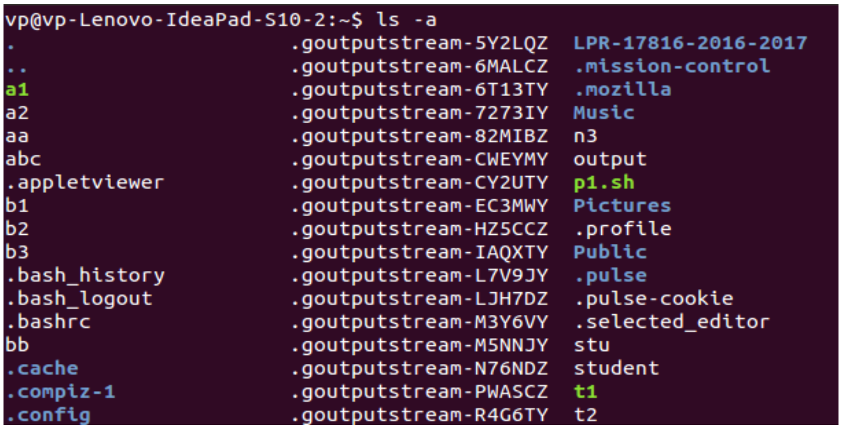
<b>Linux file System</b>

Each of directories under the root is organized to store specific types of files.

- All the files are stored on the disk in a main directory, which is called a <b>root</b> (/) directory. The root directory is subdivided into other directories: bin, home, usr, etc and dev. Every single file and directory starts from the root directory. Only root user has write privilege under this directory.
</td>
<td><b>Description: 4 M</b></td>
</tr>
</table>

		• The **/bin** directory stores many utilities available in Linux. These utilities are the commands in the Linux operating system and stores in the binary format. Common Linux commands you need to use in single-user modes are located under this directory. For example: ps, ls, ping, grep, cp • Just like /bin, **/sbin** also contain binary (executable) files, usually for system administration. For example fdisk, iptable, reboot and ifconfig utlities. • The **/etc** directory stores the data related to the operating system, including the essential operating system programs and configuration files. This also contains startup and shutdown shell scripts used to start/stop individual programs. For example passwd • The **/dev** directory stores all the device-related files for the Linux operating system. These include terminal devices, usb, or any device attached to the system. For example: /dev/tty1, /dev/usbmon0 • The **/proc** directory contains information about system processes like process id, process priority and other information. • The **/var** directory contains information specific to different utilities available in Linux. Var stands for variable files. This includes system log files (/var/log); packages and database files (/var/lib); emails (/var/mail); print queues (/var/spool); lock files (/var/lock); temp files needed across reboots (/var/tmp); • The **/temp** directory contains temporary files created by system and users. Files under this directory are deleted when system is rebooted. • The **/usr** directory stores the operating system files that are not required during the startup process. It Contains binaries, libraries, documentation, and source-code for second level programs. The /usr/bin directory is different from the /bin directory. • The **/home** directory contains the home directories of all the users to stores their personal files. For example /home/Vijay, /home/Santosh. • The **/boot** directory contains the files for booting the system. Kernel initrd, vmlinux, grub files are located under /boot • The **/lib** directory contains the libraries for the compilers, such as C, java installed in the Linux operating system. Contains shared library files and sometimes other kernel-related files. • The **/opt** directory contains add-on applications from individual vendors. Add-on applications should be installed under either /opt/ or /opt/ sub-directory. • The **/mnt** directory are used to mount other temporary file systems, such as cdrom and floppy for the CD-ROM drive and floppy diskette drive, respectively • The **/media** directory are used to mount directory for removable devices temporary.	

		• **srv** stands for service. Contains server specific services related data. For example, /srv/cvs contains CVS related data.	
3		**Attempt any THREE of the following :**	**12 M**
	a	**Write the output for the following commands:** **(i) ls –a** **(ii) date “+%D”**	**4 M**
	Ans	**i) ls –a :** list all files including hidden files. These are files that start with “.”.	**Proper output of each command : 2 M**

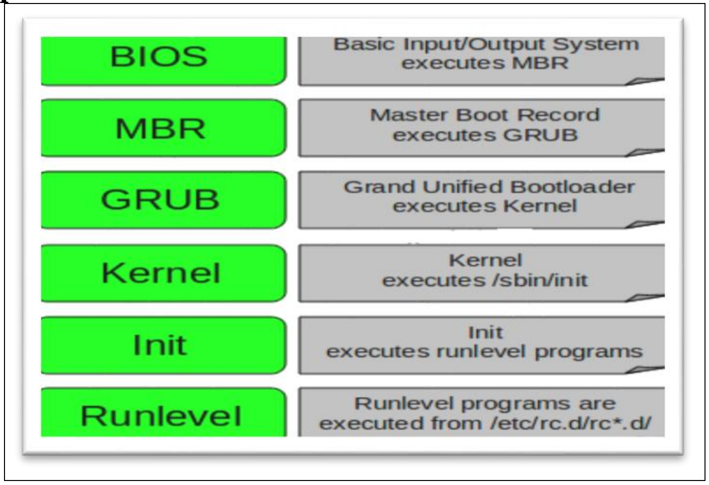


```
vp@vp-Lenovo-IdeaPad-S10-2:~$ ls -a
.                       .goutputstream-5Y2LQZ  LPR-17816-2016-2017
..                      .goutputstream-6MALCZ  .mission-control
a1                      .goutputstream-6T13TY  .mozilla
a2                      .goutputstream-7273IY  Music
aa                      .goutputstream-82MIBZ  n3
abc                     .goutputstream-CWEYMY  output
.appletviewer           .goutputstream-CY2UTY  p1.sh
b1                      .goutputstream-EC3MWY  Pictures
b2                      .goutputstream-HZ5CCZ  .profile
b3                      .goutputstream-IAQXTY  Public
.bash_history           .goutputstream-L7V9JY  .pulse
.bash_logout            .goutputstream-LJH7DZ  .pulse-cookie
.bashrc                 .goutputstream-M3Y6VY  .selected_editor
bb                      .goutputstream-M5NNJY  stu
.cache                  .goutputstream-N76NDZ  student
.compiz-1               .goutputstream-PWASCZ  t1
.config                 .goutputstream-R4G6TY  t2
```

ii) date “+%D” : Display date as mm/dd/yy.

output: 11/26/19

	b	**Explain the start up and shut down operations.**	**4 M**
	Ans	**Startup operation:**	**Suitable Explanation: 2 M each**


BIOS
Basic Input/Output System executes MBR
MBR
Master Boot Record executes GRUB
GRUB
Grand Unified Bootloader executes Kernel
Kernel
Kernel executes /sbin/init
Init
Init executes runlevel programs
Runlevel
Runlevel programs are executed from /etc/rc.d/rc*.d/

BIOS: BIOS stands for Basic Input/Output System. Performs some system integrity checks

Searches, loads, and executes the boot loader program. It looks for boot loader in floppy, cd-rom, or hard drive. You can press a key (typically F12 of F2, but it depends on your system) during the BIOS startup to change the boot sequence. Once the boot loader program is detected and loaded into the memory, BIOS gives the control to it. So, in simple terms BIOS loads and executes the MBR boot loader.

MBR stands for Master Boot Record.: It is located in the 1st sector of the bootable disk. Typically /dev/hda, or /dev/sda. MBR is less than 512 bytes in size. This has three components 1) primary boot loader info in 1st 446 bytes 2) partition table info in next 64 bytes 3) mbr validation check in last 2 bytes. It contains information about GRUB (or LILO in old systems). So, in simple terms MBR loads and executes the GRUB boot loader.

GRUB stands for Grand Unified Bootloader: If you have multiple kernel images installed on your system, you can choose which one to be executed. GRUB displays a splash screen, waits for few seconds, if you don’t enter anything, it loads the default kernel image as specified in the grub configuration file. GRUB has the knowledge of the filesystem.Grub configuration file is /boot/grub/grub.conf. GRUB just loads and executes Kernel and initrd images.

Kernel: Mounts the root file system as specified in the “root=” in grub.conf. Kernel executes the /sbin/init program. Since init was the 1st program to be executed by Linux Kernel, it has the process id (PID) of 1. Do a ‘ps -ef | grep init’ and check the pid. initrd stands for Initial RAM Disk. initrd is used by kernel as temporary root file system until kernel is booted and the real root file system is mounted. It also contains necessary drivers compiled inside, which helps it to access the hard drive partitions, and other hardware.

Init: Following are the available run levels
0 – halt
1 – Single user mode
2 – Multiuser, without NFS
3 – Full multiuser mode
4 – unused
5 – X11
6 – reboot
Init identifies the default initlevel from /etc/inittab and uses that to load all appropriate program. Execute ‘grep initdefault /etc/inittab’ on your system to identify the default run level. If you want to get into trouble, you can set the default run level to 0 or 6. Since you know what 0 and 6 means, probably you might not do that. Typically you would set the default run level to either 3 or 5.

Runlevel programs: When the Linux system is booting up, you might see various services getting started.

Shutdown operation:

- The shutdown command brings down system in a secure way. All the logged-in users are notified about the system shutdown.
- Signal SIGTERM notifies all the processes that the system is going down, so that processes can be saved and exit properly.
- Command shutdown signals the init process to change the runlevel.
- **Runlevel 0** halts the system
- **Runlevel 6** reboots the system
- **Runlevel 1** is default state.
- You can shutdown a system by passing a definite time (in minutes). System will automatically shutdown after specified minute giving a message and time to save all work
- Syntax: shutdown <time>

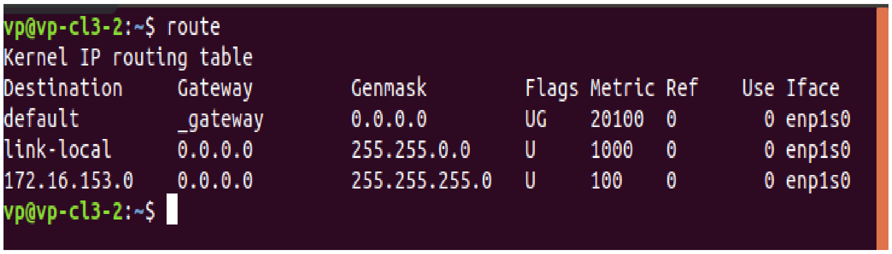
c **Describe use of following commands with example:** **4 M**

i) route
ii) net stat

Ans **Route:** Packets are transmitted from one computer to another. The route determines where the process starts and what computer system needs to send the packet for it to reach its destination. In computer networking, a router is a device responsible for forwarding network traffic. When datagrams arrive at a router, the router must determine the best way to route them to their destination. Route command is used to show/manipulate the IP routing table. It is primarily used to setup static routes to specific host or networks via an interface.

Uses: 1 M, Any suitable example: 1 M

To display the routing table entries, use route command:
$ route



```
vp@vp-cl3-2:~$ route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         _gateway        0.0.0.0         UG    20100  0        0 enp1s0
link-local      0.0.0.0         255.255.0.0     U     1000   0        0 enp1s0
172.16.153.0    0.0.0.0         255.255.255.0   U     100    0        0 enp1s0
vp@vp-cl3-2:~$
```

Each entry in the routing table has several fields.

Field	Description
Destination	Destination IP address of the route
Gateway	IP address or hostname of the gateway the route uses; * indicates no gateway is used
Genmask	The netmask for the route

Flags	Type of route; U=up, H=host, G=gateway, D=dynamic, M=modified, C=cache entry, !=reject route, R=Reinstate route for dynamic routing
Metric	Metric cost of route
Ref	Number of routes that depend on this one
Use	Number of times used
Iface	Type of interface this route uses
Window	TCP window for AX25 networks

Netstat: netstat ("network statistics") is a command-line tool that displays network connections (both incoming and outgoing), routing tables, and a number of network interface (network interface controller or software-defined network interface) and network protocol statistics.

Uses: 1 M, Any suitable example: 1 M

Options of Netstat:

Option	**Description**	**Example**
-a	Show both listening and non-listening sockets. With the –interfaces option, show interfaces that are not up	# netstat -a
-at	To list all tcp ports.	# netstat -at
-au	To list all udp ports.	# netstat -au
-l	To list only the listening ports.	# netstat -l
-lt	To list only the listening tcp ports.	# netstat -lt
-lu	To list only the listening udp ports.	# netstat -lu
-lx	To list only the listening UNIX ports.	# netstat -lx
-s	To list the statistics for all ports.	# netstat -s
-st	To list the statistics for TCP ports.	# netstat -st(TCP)
-su	List the statistics for UDP ports.	# netstat -su(UDP)
-pt	To display the PID and program names.	**#netstat -pt**
-c	To print the netstat information continuously.	# netstat -c
-r	To get the kernel routing information.	# netstat -r
-i	To get the list of network interfaces.	# netstat -i

d **Describe cat and rm file handling commands with example.** **4 M**

Ans

cat Command: 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.

- cat is used to display the contents of file. To do that for the file stu.txt, simply specify the filename as the argument: cat stu.txt
- cat is also useful for creating a file. Enter the command cat, followed by the '>' character and the filename (for ex. abc) and press the <enter> key. Enter text messages and then press <control-d>:

Description :1M, Any suitable example: 1M for each command

```
$ cat > abc
This is simple text file.
Cat is used to represent a rudimentary editor.
<control-d>
$_
```

- cat is also used to append the content in to existing file. Example: cat >> abc
- Copy the contents of one file to another file using cat command: cat file1 > file2

Rm command: 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.

Files can be deleted with rm (remove). The following command deletes the first three chapters of the text.

```
$ rm chap1 chap2 chap3
$_
```

You can easily deletes all files in directory by

```
$ rm *
$_
```

Options of rm

rm –i: the -i option makes the command ask the user for confirmation before removing each file, you have to press **y** for confirm deletion, any other key leaves the file un-deleted.

```
$ rm -i stu.txt
rm: remove regular empty file 'stu.txt'? y
```

rm –r: With -r(or -R) option rm command performs a tree-walk and will delete all the files and sub-directories recursively of the parent directory.

```
$ rm -r
```

4 **Attempt any THREE of the following :** **12 M**

a **Compare between C shell and korn shell.** **4 M**

Ans

C shell	Korn Shell
Is slower than korn shell	Is faster than the C shell.
Command full-path name is **/bin/csh**.	Command full-path name is **/bin/ksh**.
Non-root user default prompt is hostname **%**.	Non-root user default prompt is **$**.
Home directory is $home	Home directory is $HOME
No command line editing	Command line editing
End of if statement with endif	End of if statement with fi

1 M for each point, any four valid points

b **Describe working of diff and cmp file handling commands with example.** **4 M**

Ans

Diff command: diff stands for difference. This command is used to display the differences in the files by comparing the files line by line. it tells us which lines in one file have is to be changed to make the two files identical.

```
$ cat file1
shukla p.k.
chanchal singh
s.n.dasgupta
chakroborthy
$_
$ cat file2
barun sengupta
shukla p.k.
anil agrawal
chowdhary
s.n.dasgupta
$_
$ diff file1 file2
oal
> barun sengupta
2c3, 4
-------
> anil agarwal
> chakroborthy
$_
```

the instruction oa1 indicates that a single line has to appended after line number 0 of the first file and the resultant line will have line number 1 and 2nd file.

Cmp Command: 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.

Example:

```
$ cmp chap1 chap2
chap1 chap2             differ : char 9, line1
$_
```

The two files are compared byte by byte and the location of the first mismatch is echoed to the screen. If the two files are identical then cmp displays no message, but simply returns the $ prompt.

The –l (list) option gives detailed list of the byte number and the differing bytes in octal for each character that differs in both the files.

```
$ cat file1
abcd
```

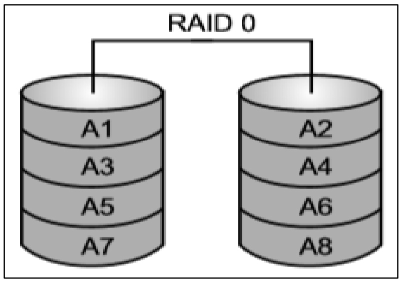
Describe:1 M, Any suitable example:1 M for each command

		xyz $ cat file2 Abed wxy $_ $cmp –l file1 file2 3 143 145 6 170 167 7 171 170 8 172 171 $_	
	c	**Write shell script to find factorial of a number.**	**4 M**

	Ans	i=1
		f=1
		echo " Enter the number"
		read n
		while [$i -le $n]
		do
		f=`expr $f \* $i`
		i=`expr $i + 1`
		done
		echo FACTORIAL = $f
		
		output:
		Enter the number
		5
		FACTORIAL=120
			Proper logic with syntax : 4 M

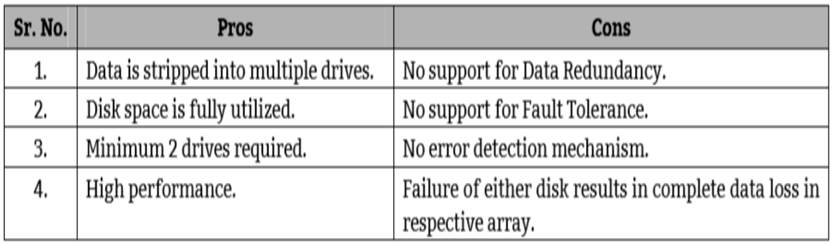
	d	**Describe working of RAID-0 and RAID-1 with suitable example.**	**4 M**
	Ans	**RAID 0 (Striped Disk Array Without Fault Tolerance):** • This level strips the data into multiple available drives equally giving a very high read and write performance but offering no fault tolerance or redundancy. • RAID level does not provides any of the RAID factor and cannot be considered in an organization looking for redundancy instead it is preferred where high performance is required. Simple striping is used in this level to gain in performance. This level does not offer any redundancy. Data is broken into stripes of user-defined size and written to a different drive in the array. • Minimum of two disks are required. It uses 100% of the storage capacity since no redundant information is written. Recommended use for this level is when your data changes infrequently and is backed up regularly and you require high-speed access.	**Each RAID level : 2 M**

- Web servers, graphics design, audio and video editing, and online gaming are some example applications that might benefit from this level.



Example:

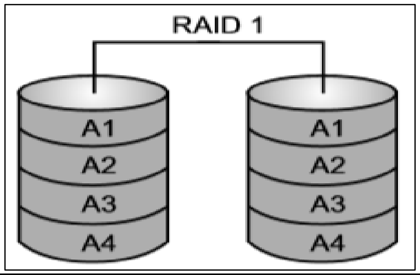
Calculation:
No. of Disk: 5
Size of each disk: 100GB
Usable Disk size: 500GB



Sr. No.	Pros	Cons
1.	Data is stripped into multiple drives.	No support for Data Redundancy.
2.	Disk space is fully utilized.	No support for Fault Tolerance.
3.	Minimum 2 drives required.	No error detection mechanism.
4.	High performance.	Failure of either disk results in complete data loss in respective array.

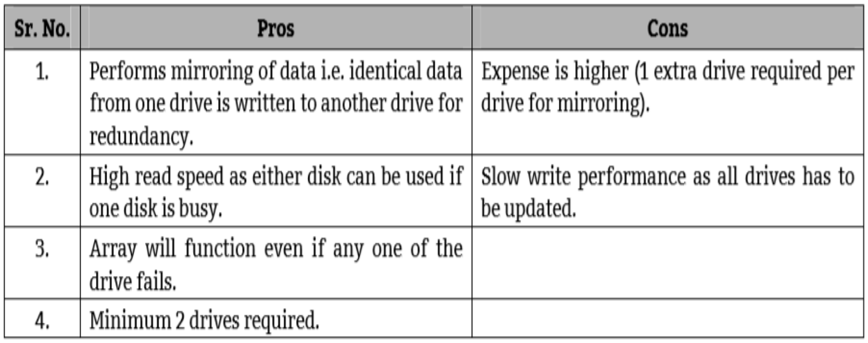
RAID 1 (Mirroring and Duplexing):

- This level performs mirroring of data in drive 1 to drive 2. It offers 100% redundancy as array will continue to work even if either disk fails. So organization looking for better redundancy can opt for this solution but again cost can become a factor.
- This level uses mirroring and data is duplicated on two drives. If either fails, the other continues to function until the failed drive is replaced.
- At the cost of 50% of available capacity, this level provides very high availability. Rebuild of failed drives is relatively fast. Read performance is good and write performance is fair compared to single drive read and write.
- A minimum of 2 drives is required. Whenever the need for high availability and vital data are involved, this level is a good candidate for use.



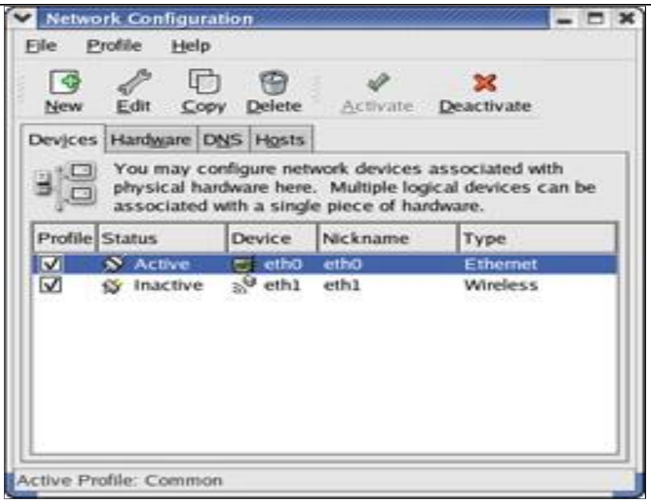
Example:

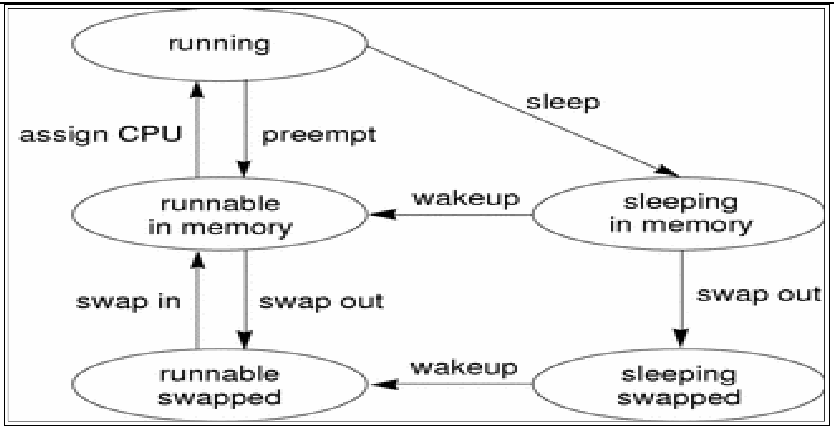
Calculation:
No. of Disk: 2
Size of each disk: 100GB
Usable Disk size: 100GB



Sr. No.	Pros	Cons
1.	Performs mirroring of data i.e. identical data from one drive is written to another drive for redundancy.	Expense is higher (1 extra drive required per drive for mirroring).
2.	High read speed as either disk can be used if one disk is busy.	Slow write performance as all drives has to be updated.
3.	Array will function even if any one of the drive fails.	
4.	Minimum 2 drives required.	

	e	**Write procedure to configure TCP/IP settings in a network.**	**4 M**
	Ans	• Linux includes a network configuration tool that enables you to configure various network interfaces on your system for TCP/IP networking. • You can run the network configuration tool to add a new network interface or to alter information such as name servers and hostnames. • Log in as root and from the GNOME desktop, select Main Menu>Programs>System>Network Configuration to run the Network Configuration tool. • The Network Configuration tool displays a tabbed dialog box	**Any suitable procedure: 4 M**

		 Network Configuration File Profile Help New Edit Copy Delete Activate Deactivate Devices Hardware DNS Hosts You may configure network devices associated with physical hardware here. Multiple logical devices can be associated with a single piece of hardware. Profile Status Device Nickname Type ☑ Active eth0 eth0 Ethernet ☑ Inactive eth1 eth1 Wireless Active Profile: Common • **Devices:** Lets you add a new network interface, specify the IP address of the interface, and activate the interface. This information is stored in various files in the /etc/sysconfig directory. • **Hardware:** Lets you add a new hardware device such as an Ethernet card, modem, or an ISDN device. You can then provide information such as interrupt request (IRQ) and I/O port numbers, and DMA channels for the device. • **DNS:** Lets you enter the hostname for your system and the IP addresses of name servers. The name server addresses are stored in the /etc/resolv.conf file. The host name is stored in the HOSTNAME variable in the /etc/sysconfig/ network file. • **Hosts:** Shows you the current contents of the /etc/hosts file and lets you add, remove, or edit entries. To configure the network interfaces, you need to assign IP addresses to each interface. If you are running a private network, you may use IP addresses in the range 192.168.0.0 to 192.168.255.255	
5		**Attempt any TWO of the following :**	**12 M**
	a	**With Suitable diagram, describe state transition diagram.**	**6 M**
	Ans		



An active process is normally in one of the five states in the diagram. The arrows show how the process changes states.

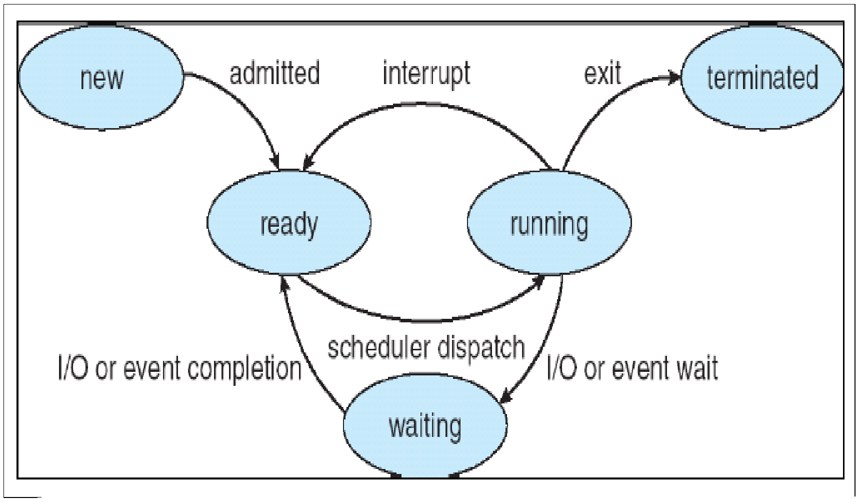
- A process is running if the process is assigned to a CPU. A process is removed from the running state by the scheduler if a process with a higher priority becomes runnable. A process is also pre-empted if a process of equal priority is runnable when the original process consumes its entire time slice.
- A process is runnable in memory if the process is in primary memory and ready to run, but is not assigned to a CPU.
- A process is sleeping in memory if the process is in primary memory but is waiting for a specific event before continuing execution. For example, a process sleeps while waiting for an I/O operation to complete, for a locked resource to be unlocked, or for a timer to expire. When the event occurs, a wakeup call is sent to the process. If the reason for its sleep is gone, the process becomes runnable.
- When a process' address space has been written to secondary memory, and that process is not waiting for a specific event, the process is runnable and swapped.
- If a process is waiting for a specific event and has had its whole address space written to secondary memory, the process is sleeping and swapped.

  If a machine does not have enough primary memory to hold all its active processes, that machine must page or swap some address space to secondary memory.
- When the system is short of primary memory, the system writes individual pages of some processes to secondary memory but leaves those processes runnable. When a running process, accesses those pages, the process sleeps while the pages are read back into primary memory.
- When the system encounters a more serious shortage of primary memory, the system writes all the pages of some processes to secondary memory. The system marks the pages that have been

Diagram: 2 M; Description: 4 M

written to secondary memory as swapped. Such processes can only be scheduled when the system scheduler daemon selects these processes to be read back into memory.

OR



During the lifespan of a process, its execution status may be in one of four states: (associated with each state is usually a queue on which the process resides)

New: The process being created is available in the new state. It is the new state because the system is not permitted it to enter the ready state due to limited memory available in the ready queue. If some memory becomes available, then the process from the new state will go to ready state.

Ready State: The process which is not waiting for any external event such as I/O operation and which is not running is said to be in ready state. It is not in the running state because some other process is already running. It is waiting for its turn to go to the running state.

Running State: The process which is currently running and has control of the CPU is known as the process in running state. In single user system, there is only one process which is in the running state. In multiuser system, there are multiple processes which are in the running state.

Blocked State: The process is currently waiting on external event such as an I/O operation is said to be in blocked state. After the completion of I/O operation, the process from blocked state enters in the ready state and from the ready state when the process turn will come it will again go to running state.

		Terminated / Halted State: The process whose operation is completed, it will go the terminated state from the running state. In halted state, the memory occupied by the process is released.	
	b	**) Describe Chmod command with example.**	**6 M**
	Ans	In Unix-like operating systems, the chmod command sets the permissions of files or directories. On Unix-like operating systems, a set of flags associated with each file determines who can access that file, and how they can access it. These flags are called file permissions or modes, as in "mode of access." The command name chmod stands for "change mode." It restricts the way a file can be accessed. In general, chmod command take the form: chmod options permissions file name If no options are specified, chmod modifies the permissions of the file specified by file name to the permissions specified by permissions. Permissions defines the permissions for the owner of the file (the "user"), members of the group who owns the file (the "group"), and anyone else ("others"). There are two ways to represent these permissions: with symbols (alphanumeric characters), or with octal numbers (the digits 0 through 7). **Syntax :** chmod [reference][operator][mode] file... **Example:** chmod u=rwx,g=rx,o=r myfile The letters u, g, and o stand for "user", "group", and "other". The equals sign ("=") means "set the permissions exactly like this," and the letters "r", "w", and "x" stand for "read", "write", and "execute", respectively. The commas separate the different classes of permissions, and there are no spaces in between them. The equivalent command using octal permissions notation: *chmod 754 myfile* Here the digits 7, 5, and 4 each individually represent the permissions for the user, group, and others, in that order. Each digit is a combination of the numbers 4, 2, 1, and 0: 4 stands for "read", 2 stands for "write", 1 stands for "execute", and 0 stands for "no permission."	**Description: 4 M;** **Example 2 M**
	c	**Write a shell script to accept a number from user check whether number is even or odd.**	**6 M**

	Ans	# !/usr/bin/ksh
		
		echo "Input number"
		read number < /dev/tty
		if [$(($number % 2)) -eq0]
		then
		  echo "$number is even"
		else
		  echo "$number is odd"
		fi
		
		exit 0
			Accept numbers: 2 M; checking even odd of numbers: 4 M

6		**Attempt any TWO of the following :**	**12 M**
	a	**Describe the procedure of designing a firewall.**	**6 M**
	Ans	**Step 1: Retrieve the Iptables firewall:** Iptables is pre-installed on almost every Linux distribution. You can use this command to retrieve the package: **sudo apt-get install iptables** **Step 2: Discover what Iptables is already configured to do by default:** **Run the iptable L command** **Step 3: You can decide to modify the existing rules or instead start afresh:** To start afresh, run this command **iptables-F** **Step 4: Decide which firewall ports to close:** First block all lines of attack by running the following commands: Block XMAS Packets: **iptables -A INPUT -p tcp –tcp-flags ALL ALL -j DROP** Block null packets: **iptables -A INPUT -p tcp –tcp-flags ALL NONE -j DROP** Block syn-flood packets: **iptables -A INPUT -p tcp ! –syn -m state –state NEW -j DROP** **Step 5: Decide which firewall ports to leave open:** Here are some ports you could decide to leave open: For outgoing connections: • 80/tcp for HTTP • 53/udp for DNS • 443/tcp for HTTPS (secured HTTP) • 21/tcp for FTP (File Transfer Protocol) • 465/tcp for SMTP (send emails) • 25/tcp for Insecure SMTP • 22/tcp for SSH (secure connection from computer to computer) • 993/tcp&udp for IMAP (receive emails) • 143/tcp&udp for Insecure IMAP	**6 steps : 1 M each or Relevant Description of procedure : 6 M**

		• 9418/tcp for GIT (version control system) For Incoming connections: • 993/tcp&udp for IMAP (receive emails) • 143/tcp&udp for Insecure IMAP • 110/tcp for POP3 (old way to receive emails) • 22/tcp for SSH (secure connection from computer to computer) • 9418/tcp for GIT (version control system) **Step 6: Save your firewall configuration** Type the following command to save the settings you've configured and restart your firewall: **iptables -L -n** **iptables-save | sudo tee /etc/sysconfig/iptables** **service iptables restart**	
	b	**Write a procedure to perform the operation for managing the users / groups.**	**6 M**
	Ans	**Creating a user with a default setting:** A user can be added by running the *useradd* command at the command prompt. # useraddabc # passwdabc Changing password for user anirban. New password: Retype new password: passwd: all authentication tokens updated successfully. **Specifying a user's full name when creating a user:** A systems administrator can use the –c option with *useradd* to specify the user's full name, as shown below: # useradd -c "Anil Bhai Choudhury" abc **Creating a user with the UID:** You can create a user with a custom UID with the –u option, as follows: # useradd -u 1036 abc **Adding a user to a primary group and supplementary group:** A systems administrator can specify a primary group and a supplementary one by specifying the *–g* and *–G* option, respectively. # useradd -g "head" -G "faculty" abc **Locking and unlocking a user:** A super user can lock and unlock a user account. To lock an account, one needs to invoke passwd with the -l option. # passwd -l abc Locking password for user abc. passwd: Success The *–u* option with *passwd* unlock an account	**Any 3 Commands for users 1 M each; Any 3 Commands for managing group 1 M each**

```
# passwd -uabc
Unlocking password for user abc.
passwd: Success
```

Changing a user name: The –l option with the usermod command changes the login (user) name, as shown below:

```
# usermod -l “pqr” abc
```

Removing a user: Combining userdel with the –r option drop a user and the home directory associated with that user, as shown below:

```
# userdel -r pqr
```

Linux group

Linux group is a mechanism to organise a collection of users. Like the user ID, each group is also associated with a unique ID called the GID (group ID).

Creating a group with default settings: To add a new group with default settings, run the groupadd command as a root user, as shown below:

```
# groupaddmygroup
```

Changing the group’s name: To change the group’s name, run the groupmod command with the -n option as a super user, as shown below:

```
# groupmod -n mynewgroupmygroup
```

Creating a group with a specified GID: To explicitly specify the GID of a group, execute the groupadd command with the –g option, as follow:

```
# groupadd -g 1200 manager
```

Deleting a group: Before deleting a primary group, delete the users of that primary group. To delete a group, run the groupdel command with the group name, as shown below:

```
# groupdelmynewgroup
```

c **Write a shell script to accept length and breadth of rectangle from user. Calculate and display area, perimeter, of entered values using choice entered by user. (Hint : Use case statement)** **6 M**

Ans

```
#!/bin/bash
# GNU bash, version 4.3.46

echo "Enter Length of Rectangle: "
read length
echo "Enter Breadth of Rectangle: "
read breadth

echo "Which operation you want to perform? 1: area 2: perimeter"
read ch
case $ch in
        1) res=`echo $length \* $breadth | bc`
```

Accept values: 2 M; use of case statement: 2 M; Calculating and displaying result: 2 M

		;; 2) res=` echo 2 \* $length \* $breadth | bc` ;; esac echo "Result is $res" exit 0	