1.List the contents of a directory and their attributes.

-w to check write permission

-a and operator

-gt greater than

-lt less than

Bc- basic commands

-l calling math library

-le less than or equal to

-o or operator

-f fuction

-ne is not equal to

-ge greater than or equal to

-le less than or equal to

-p allow to show i/p on screen

-sp not allow to show i/p on screen

-h explain file size in human friendly units (kb.mb.tb…)

-l display permission link owner

-A List all entries including those starting with periods (.), but excluding any. or ... entries.

-a lists all entries including those starting with a period (.).

-b displays non-printable characters as octal bytes with the form \ooo.

-C puts output into columns, sorted vertically; this is the default output format to the terminal.

-c uses the creation time of the file for sorting (-t) or displaying (-l).

-d does not display the contents of named directories, but show information on the directories themselves.

-F puts a / after each directory name, a \* after every executable file, a | after every FIFO file, a @ after every symbolic link and a = after every socket.

-f enables the -a option and disables the -C, -g, -l, -n, -o, -r, -s, and -t options. For each argument that is a directory, all directory entries are listed in the same order they are retrieved from the system (POSIX-compliant and UNIX systems only).

2.Long list the content with file, directory ownership, permissions,sizes, etc…

r- permission to read file

w- permission write the file

x- permission to execute the file

-s sort by file size

-t sort by modification time

- regular file

-C character special file

-d directory

l- symbolic file

n- network file

s- socket

3.Display the size of the file in human readable format.

r- permission to read file

w- permission write the file

x- permission to execute the file

4.Show all files and folders including hidden one

-A in order to show hidden files on linux

Ls – shows all files and folder

Ls -ls listing files and folders from largest size to last.

5.list directories recursively

ls -R : Use the ls command to get recursive directory listing on Linux

find /dir/ -print : Run the find command to see recursive directory listing in Linux

du -a . : Execute the du command to view recursive directory listing on Unix

6.Sort the files by size with largest at the top

-l long listing

-s sort by file size,largest first

--sort=size sort by size instead of file name

-r reverse order while sorting

-h human readable output with -l and -s option print like size 1k,6gb,4tb etc.

7.Sort the files by last time modified displaying the newest first.

Ls -ltu list files based on the last access time the newest files will be listed first.

-u Sort by access time, newest first

Ls -lt

l-long listing formart

t-sort by modification time newest first

ls -ltr files in reverse order, i.e oldest files first

ls -ltc -c sort by last modification time

8.Diplay the location of a program/command, where it is installed.

Display ls command location along with man page path:  
**whereis ls , which**

9.Which command is used to switch directory from one to another

/: Moves to the root directory (forward slash)

..: Moves to the parent directory (two periods without a space)

~: Moves to the logged-in user's home directory (tilde)

-: Moves to the previous directory (hyphen)

10.List all the environment variables set for the current shell environment

Path /user/home/editor/uid/term/shell

11.Did you notice something in the output of "env" command?

Evn[option]..[-][name-value]…[command..[args]…]

Set each name to value in the environment and run command.

12.Which command is used to print the text or any variables value in the Console/Terminal?

Set command

Envprint command

13.Print the value of the env variable "PATH" on the console

echo "$PATH"  
printf "%s\n", $PATH

14.Is linux a case-sensitive operating system?

It is an open-source operating system.It is free.The file names are case-sensitive.A monolithic kernel is used in this operating system.It is more efficient in comparison to Windows OS.A ‘forward slash’ is used to separate the directories.It provides higher level of security in comparison to windows.It is widely used by programmers, mostly for developmental, gaming, and hacking purposes.

15.Is, "Ls" same as "ls"?

Ls is the name of a Linux executable file in the /bin directory that when executed, lists the visible files in the directory specified as argument. However, if no directory is specified, ls lists the files in the current working directory. Ls, like most other executable files or commands, has a different behavior of execution when a flag is passed to it. ls -l is a Linux command that will list all “visible” files in current directory in long format and sorted in descending and alphabetical order.

16.Display your currently logged in user

* W

17.how do you change the currrently logged in user to another user?

* su - user2name

18.Which command is used to leave a shell environment that you are currently logged in to?

exit command in linux is used to exit the shell where it is currently running. It takes one more parameter as[n] and exits the shell with a return of status[n]. If n is not provided, then it simply returns the status of last command that is executed.

19.How do you reboot the system?

1. To reboot the Linux system from a terminal session, sign in or “su”/”sudo” to the “root” account.
2. Then type sudo reboot to reboot the box.
3. Wait for some time and the Linux server will reboot itself

20.How do you shutdown the system?

Sudo shutdown

Shutdown[option]…time[message]

Sudo Shutdown -c (for cancelling shutdown)

Sudo shutdown -h now(it will directly shutdown your system)

21.Display all the major running processes in the system.

The **ps** (process statuses) command produces a snapshot of all running processes.

Both the htop and top command display the same information when listing your Linux processes, but the former offers user-friendly features that are great for everyday process management.

* ps command — outputs a static view of all processes.
* top command — displays the real-time list of all running processes.
* htop command — shows the real-time result and is equipped with user-friendly features.

22.Understand the output of command used in above question of displaying processes, explain the meaning of each column and what data it displays?

The **ps** (process statuses) command produces a snapshot of all running processes.

Both the htop and top command display the same information when listing your Linux processes, but the former offers user-friendly features that are great for everyday process management.

* ps command — outputs a static view of all processes.
* top command — displays the real-time list of all running processes.
* htop command — shows the real-time result and is equipped with user-friendly features.

23.Display the name of the system kernel

* -o

24.display the kernel release number

* -r

25.display teh machine type of the current kernel

* Uname -r

26.Display the name of the operating system that the kernel is running on

* uname

27.Display all info that uname command can show.

The command uname display information about the system.

Uname -a/uname -s/uname -n/uname -r/ uname -p/uname -m/ uname -v

28.Display the name of directory that you are currently pointing to

* cwd/cd

29.change the current directory to another directory that you have in your system.

Cd[directory]

>pwd

>cd

>pwd

30.Go up one directory

$ cd /usr/share/backgrounds/

backgrounds:$ goUp 2

usr:$

31.Return to last directory

-cd goes back to previous directory.

32.change the current directory to home(logged in user's) directory

mv /home/myusername /home/myusername-old

ln -s /var/www/html /home/myusername

33.How to check all the command used from the prompt (Command History)

Command [number] history 10

$ history

34.In which file the history of commands are stored in?

~/.bash\_history

35.How many lines of history does the system keep and from where you can change it?

36.How can you modify bash's history behaviour

To customize bash command history, we will have to make changes in the ~/.bashrc file. To edit the ~/.bashrc file

$ nano ~/.bashrc

Once you are done with modifying the file, use Ctrl+O and Ctrl+X to save and close the nano editor.

Then run the following command to apply the modifications:

$ source ~/.bashrc

37.Display all the commands entered so far, now, try to run a particular command from the history list without typing that command.

* Use Get-Command to see all the command entered so far

NOTE:- TAB key is your friend when it comes to command completion and having long file and directory names autocompleted at the bash prompt for you. JUST BE LAZY AND USE TAB KEY FOR AUTO COMPLETION ;-)

38.What are the different types of shell and where are they used and how do we use them?

The shell provides you with an interface to the UNIX system. It gathers input from you and executes programs based on that input. When a program finishes executing, it displays that program's output.

the prompt, $, which is called command prompt, is issued by the shell. While the prompt is displayed, you can type a command.

The shell reads your input after you press Enter. It determines the command you want executed by looking at the first word of your input. A word is an unbroken set of characters. Spaces and tabs separate words.

* Bourne shell ( sh)
* Korn shell ( ksh)
* Bourne Again shell ( bash)
* POSIX shell ( sh)

The different C-type shells follow:

* C shell ( csh)
* TENEX/TOPS C shell ( tcsh)

39.What is the difference between login shell and non-login shell?

A Login shell is created after a successful login of user. For example, when you login t a Linux system via terminal, SSH or switch to user with “su -” command.

When a login shell start, it runs a set of predefined script to configure shell environment. To identify the login shell, run the below command on terminal.

$ echo $0

Non Login Shell is the shell, which is started by the login shell. For example, A shell which you started from another shell or started by a program etc.

A non login shell executes the following script to set the shell environment.

* Non login shell first executes ~/.bashrc
* Then ~/.bashrc executes /etc/bashrc
* /etc/bashrc calls the scripts in /etc/profile.d

40.How do we start login shell and non-login shell?

* The login shell is started after successfully login the /bin/login file, by reading the /etc/password file. And Non login shell is started by a program without a login. In case the program just passes the shell executable.

41.What happens when you start a login shell (which files are read and used and Why)?

When Bash is invoked as an interactive login shell, or as a non-interactive shell with the --login option, it first reads and executes commands from the file /etc/profile, if that file exists. After reading that file, it looks for ~/.bash\_profile, ~/.bash\_login, and ~/.profaile, in that order, and reads and executes commands from the first one that exists and is readable. The --noprofile option may be used when the shell is started to inhibit this behavior.

42.What happens when you start a non-login shell (Which files are read and used and Why)?

When an interactive shell that is not a login shell is started, Bash reads and executes commands from ~/.bashrc, if that file exists. This may be inhibited by using the --norc option. The --rcfile *file* option will force Bash to read and execute commands from *file* instead of ~/.bashrc.

So, typically, your ~/.bash\_profile contains the line

43.What are Shell Configuration Files, why do we need it?

Shell configuration files are executed automatically when you log in and. out of a shell. They initialize and configure a shell upon login and. perform cleanup operations upon logout. A configuration file, often shortened to config file, defines the parameters, options, settings and preferences applied to operating systems (OSes), infrastructure devices and applications in an IT context. Software and hardware devices can be profoundly complex, supporting myriad options and parameters.

44.Explain the Order of file usage from the system/user's home directory when user logs in to the System.

45.What are Shell Variables, list major shell variables and what do they represent?

A shell variable is a variable that is available only to the current shell. In contrast, an environment variable is available system wide and can be used by other applications on the system.

A shell is the operating system's command interpreter. It processes the commands entered on the command line or read from a shell script file.

* **Environment variables –** Variables that are exported to all processes spawned by the shell. Their settings can be seen with the env command. A subset of environment variables, such as PATH, affects the behavior of the shell itself.
* **Shell (local) variables –** Variables that affect only the current shell. In the C shell, a set of these shell variables have a special relationship to a corresponding set of environment variables. These shell variables are user, term, home, and path. The value of the environment variable counterpart is initially used to set the shell variable

46.How we see all our env variables?

Printenv commands- print all or part of environment

Env commands- display all exported environment or run a program in a modified environment

Set command – list the name and value of each shell variable

47.How we see all env variables in alphabical order?

Exploring environment variables is straightforward using the get-childitem cmdlet. To see a complete list of environment variables,

Get-childitem env:\*

Get-childitem env:\* |

Sort-object key |

more

48.What Format does the env var and its values are stored?

The Global environment variables of your system are stored in /etc/environment.  
Any changes here will get reflected throughout the system and will affect all users of the system. Also, you need a Reboot, for any changes made here to take effect.

User level Environment variables are mostly stored in .bashrc and .profile files in your Home folder. Changes here only affect that particular user. Just close and open the terminal for configuration changes to take place.

49.How do you create your own varible?

You can set your own variables at the command line per session, or make them permanent by placing them into the ~/.bashrc file, ~/.profile, or whichever startup file you use for your default shell. On the command line, enter your environment variable and its value as you did earlier when changing the PATH variable.

me=dave

my\_boost=linux

him=popeye

his\_boost=Spanish

this\_year=2021

to see the value held in variable use echo command.

Echo $my\_name

Echo $my\_boost

Echo $this\_year

50.How do you start a new bash shell?

51.Difference between Local/Shell variables to Global Variable

|  |  |  |
| --- | --- | --- |
| Scope | It is declared inside a function. | It is declared outside the function. |
| Value | If it is not initialized, a garbage value is stored | If it is not initialized zero is stored as default. |
| Lifetime | It is created when the function starts execution and lost when the functions terminate. | It is created before the program's global execution starts and lost when the program terminates. |
| Data sharing | Data sharing is not possible as data of the local variable can be accessed by only one function. | Data sharing is possible as multiple functions can access the same global variable. |
| Parameters | Parameters passing is required for local variables to access the value in other function | Parameters passing is not necessary for a global variable as it is visible throughout the program |
| Modification of variable value | When the value of the local variable is modified in one function, the changes are not visible in another function. | When the value of the global variable is modified in one function changes are visible in the rest of the program. |
| Accessed by | Local variables can be accessed with the help of statements, inside a function in which they are declared. | You can access global variables by any statement in the program. |
| Memory storage | It is stored on the stack unless specified. | It is stored on a fixed location decided by the compiler. |

52.Making a variable accessible from other the shell in the system.

53.Show the real life use case of exporting a variable

Export -p

-p:to view all the exported variable on current shell.

-f:it must be used if the name refers to function.

Name[=value]:you can assign value before exporting the syntax

$ export name[=value]export [-f] [-n] [name[=value] ...] or export -

54.Convert the above script file into a command, The file should run with "myappp" instead of "myapp.sh"

55.What is Globbing? Explain in depth with examples?

Bash does not support native regular expressions like some other standard programming languages. The Bash shell feature that is used for matching or expanding specific types of patterns is called globbing. Globbing is mainly used to match filenames or searching for content in a file. Globbing uses wildcard characters to create the pattern.

Example: Suppose, you want to search those text filenames whose names are 4 characters long and extension is .txt. You can apply globbing pattern by using ‘?’ four times to do this task.

Find out the list of all files and folder of the current directory.

$ ls -l

$ ls -l ????.txt

56.List all entries with extension ".sh"

* Use ls will give you all the list of files and to find the type of the file just use ‘file’ command.

57.List all entries with numbers in it.

58.List all entries that starts with a character and ends with a number

59.List all entries that name length more than 5 characters

60.What is Quoting? and Why do we need it?

Quoting is used to remove the special meaning of certain characters or words to the shell. Quoting can be used to disable special treatment for special characters, to prevent reserved words from being recognized as such, and to prevent parameter expansion.

In Linux Shell, many special characters have their own special meanings. Sometimes they are used to perform an action while other times they are just used as a character, so the quoting mechanism performs this task it makes us use them in whatever way we want to

61.Write few(minimum 3) unique examples that shows, how a particular problem is solved using Quoting.

62.How do you find a particular files/directories based on a particular search criteria?

HINT:- look for commands -> locate, find and whereis

* To find a particular files in linux use ‘locate’ command which serves to find a files on file system.
* Also you can use ‘find’ command to find different files, directories and perform subsequent operation on that.
* And we can use ‘whereis’ command is used in linux to find the location of a file of source/binary file of a command and manuals section for a specified file

63.Write major difference between locate, find and whereis?

* Locate command is used to find a file on a file system. It searches through prebuild local database of the file on filesystem.
* Where as the find command is able to find recursively search for any given path for various files.
* And the whereis command is used find a file through a given path.

64.How Globbing is different from locate, find and whereis?

Globbing is basically used to find a filename using a special character called wildcards with a single filename.

65.Explain the Linux File System.

A linux file system is a structured collection of files on a disk drive or a partition. A partition is a segment in a memory and contains some specific data. In our machine, there can be various partition of memory. Generally every partition contain a file memory.

66.Explain absolute and Relative Paths

* A relative path is defines as a path related to present working directly(pwd). It starts with current directory and never starts with a/.
* A absolute path is defined as specifying location of a file or directory from a root directory.

67.What are the different ways of creating a File in linux System? Write an example of each and the difference between them.

* We can use touch command to create a file in linus ( $ touch filename).
* Also we can create a file a file in linux using cat command ($ cat > filename).
* Echo command is also used to create a file in linux ($ echo “This is the text” > filename).
* Redirectional operator when used along with linux command directs a output toward particular destination file ( $ grep This file.txt > new\_file1.txt)
* Vim command is used to edit a file in linux ($ vim filename)

68.In how many ways we can delete the files from linux system? write an example of each and teh difference between them.

* To remove files from directory use rm command ($ rm filename)
* Unlike command is also used to delete a file from linux ( $ unlike filename)

69.Archiving files using linux command, write a command to archive set of files from linux commands.

* Tar command is used in linux to archive files ( tar [options] [archive-file] [file or directory to be archeived]).

70.Extract the archived files from the above step.

* To extract a file from archive use command -xvf ( $ tar xvf file.tar).