**UNIX Shell**

1. **How will you remove all files in current directory? Including the files that are two levels down in a sub-directory.**

In Unix we have rm command to remove files and sub-directories. With rm command we have –r option that stands for recursive. The –r option can delete all files in a directory recursively.

It means if we our current directory structure is as follows:

My\_dir

->Level\_1\_dir

-> Level\_1\_dir ->Level\_2\_dir

-> Level\_1\_dir ->Level\_2\_dir->a.txt

With rm –r \* command we can delete the file a.txt as well as sub-directories Level\_1\_dir and Level\_2\_dir.

Command:

rm – r \*

The asterisk (\*) is a wild card character that stands for all the files with any name.

1. **What is the difference between the –v and –x options in Bash shell scripts?**

In a BASH Unix shell we can specify the options –v and –x on top of a script as follows:

#!/bin/bash -x –v

With –x option BASH shell will echo the commands like for, select, case etc. after substituting the arguments and variables. So it will be an expanded form of the command that shows all the actions of the script. It is very useful for debugging a shell script.

With –v option BASH shell will echo every command before substituting the values of arguments and variables. In –v option Unix will print each line as it reads.

In –v option, If we run the script, the shell prints the entire file and then executes. If we run the script interactively, it shows each command after pressing enter.

1. **What is a Filter in Unix command?**

In Unix there are many Filter commands like- cat, awk, grep, head, tail cut etc.

A Filter is a software program that takes an input and produces an output, and it can be used in a stream operation.

E.g. cut -d : -f 2 /etc/passwd | grep abc

We can mix and match multiple filters to create a complex command that can solve a problem.

Awk and Sed are complex filters that provide fully programmable features.

Even Data scientists use Unix filters to get the overview of data stored in the files.

1. **What is Kernel in Unix operating system?**

Kernel is the central core component of a Unix operating system (OS).

A Kernel is the main component that can control everything within Unix OS.

It is the first program that is loaded on startup of Unix OS. Once it is loaded it will manage the rest of the startup process.

Kernel manages memory, scheduling as well as communication with peripherals like printers, keyboards etc.

But Kernel does not directly interact with a user. For a new task, Kernel will spawn a shell and user will work in a shell.

Kernel provides many system calls. A software program interacts with Kernel by using system calls.

Kernel has a protected memory area that cannot be overwritten accidentally by any process.

1. **What is a Shell in Unix OS?**

Shell in Unix is a user interface that is used by a user to access Unix services.

Generally a Unix Shell is a command line interface (CLI) in which users enter commands by typing or uploading a file.

We use a Shell to run different commands and programs on Unix operating system.

A Shell also has a command interpreter that can take our commands and send these to be executed by Unix operating system.

Some of the popular Shells on Unix are: Korn shell, BASH, C shell etc.

1. **What are the different shells in Unix that you know about?**

Unix has many flavors of Shell. Some of these are as follows:

Bourne shell: We use sh for Bourne shell.

Bourne Again shell: We use bash to run this shell.

Korn shell: We can use ksh to for Korn shell.

Z shell: The command to use this is zsh

C shell: We use csh to run C shell.

Enhanced C shell: tcsh is the command for enhanced C shell.

**477.**

**What is the first character of the outp in ls –l command ?**

We use ls -l command to list the files and directories in a directory.

With -l option we get long listing format.

In this format the first character identifies the entry type. The entry type can be one of the following:

* Block special file
* Character special file
* Directory
* Symbolic link s Socket link
* FIFO
* Regular file

In general we see d for directory and - for a regular file.

1. **What is the difference between Multi-tasking and Multi-user environment?**

In a Multi-tasking environment, same user can submit more than one tasks and operating system will execute them at the same time.

In a Multi-user environment, more than one user can interact with the operating system at the same time.

1. **What is Command Substitution in Unix?**

Command substitution is a mechanism by which Shell passes the output of a command as an argument to another command. We can even use it to set a variable or use an argument list in a for loop.

E.g. rm `cat files\_to\_delete`

In this example files\_to\_delete is a file containing the list of files to be deleted. cat command outputs this file and gives the output to rm command. rm command deletes the files.

In general Command Substitution is represented by back quotes `.

1. **What is an Inode in Unix?**

An Inode is a Data Structure in Unix that denotes a file or a directory on file system. It contains information about file like-location of file on the disk, access mode, ownership, file type etc.

Each Inode has a number that is used in the index table. Unix kernel uses Inode number to access the contents of an Inode.

We can use ls -i command to get the inode number of a file.

1. **What is the difference between absolute path and relative path in Unix file system?**

Absolute path is the complete path of a file or directory from the root directory. In general root directory is represented by / symbol. If we are in a directory and want to know the absolute path, we can use pwd command.

Relative path is the path relative the current location in directory.

E.g. In a directory structure /var/user/kevin/mail if we are in kevin directory then pwd command will give absolute path as /var/user/kevin.

Absolute path of mail folder is /var/user/kevin/mail. For mail folder ./mail is the relative path of mail directory from kevin folder.

1. **What are the main responsibilities of a Unix Shell?**

Some of the main responsibilities of a Unix Shell are as follows:

1. Program Execution: A shell is responsible for executing the commands and script files in Unix. User can either interactively enter the commands in Command Line Interface called terminal or they can run a script file containing a program.
2. Environment Setup: A shell can define the environment for a user. We can set many environment variables in a shell and use the value of these variables in our program.
3. Interpreter: A shell acts as an interpreter for our scripts. It has a built in programming language that can be used to implement the logic.
4. Pipeline: A shell also can hookup a pipeline of commands. When we run multiple commands separated by | pipe character, the shell takes the output of a command and passes it to next one in the pipeline.
5. I/O Redirection: Shell is also responsible for taking input from command line interface (CLI) and sending the output back to CLI. We use >, <, >> characters for this purpose.

1. **What is a Shell variable?**

A Unix Shell variable is an internal variable that a shell maintains. It is local to that Shell. It is not made available to the parent shell or child shell.

We generally use lower case names for shell variables in C shell.

We can set the value of a shell variable by set command.

E.g. % set max\_threads=10

To delete a Shell variable we can use unset command.

To use a Shell variable in a script we use $ sign in front of the variable name.

E.g. echo $max\_threads