Ineuron assignment -1

1.What is Linux? Linux is the kernel which communicate directly with Hw?

Ans-Linux is an open source operating system (OS). An operating system is the software that directly manages a system's hardware and resources, like CPU, memory, and storage. The OS sits between applications and hardware and makes the connections between all of your software and the physical resources that do the work.
Kernel: The base component of the OS. Without it, the OS doesn't work. The kernel manages the system's resources and communicates with the hardware. It's responsible for memory, process, and file management.

It mostly depends on the hardware. But typically a *nix kernel will interact with the hardware (read peripherals) using device drivers.

Device drivers are modules which let's the user access the hardware, say screen, safely and correctly. The driver writer has to read and understand the device' hardware manual and device a way to setup and interact with it.

Typically a driver code will use memory mapped hardware registers to setup and interact with the device and also setup a system call interface so that the user can access the device.

2.What is the difference between Linux and Unix? Linux is the kernel and unix is an OS?

Ans- From a user experience perspective, not very much is different! Much of the attraction of Linux was the operating system's availability across many hardware architectures (including the modern PC) and ability to use tools familiar to Unix system administrators and users.

Because of POSIX standards and compliance, software written on Unix could be compiled for a Linux operating system with a usually limited amount of porting effort. Shell scripts could be used directly on Linux in many cases. While some tools had slightly different flag/command-line options between Unix and Linux, many operated the same on both.

One side note is that the popularity of the macOS hardware and operating system as a platform for development that mainly targets Linux may be attributed to the BSD-like macOS operating system. Many tools and scripts meant for a Linux system work easily within the macOS terminal. Many open source software components available on Linux are easily available through tools like Homebrew.

The remaining differences between Linux and Unix are mainly related to the licensing model: open source vs. proprietary, licensed software. Also, the lack of a common kernel within Unix distributions has implications for software and hardware vendors. For Linux, a vendor can create a device driver for a specific hardware device and expect that, within reason, it will operate across most distributions. Because of the commercial and academic branches of the Unix tree, a vendor might have to write different drivers for variants of Unix and have licensing and other concerns related to access to an SDK or a distribution model for the software as a binary device driver across many Unix variants.

As both communities have matured over the past decade, many of the advancements in Linux have been adopted in the Unix world. Many GNU utilities were made available as add-ons for Unix systems where developers wanted features from GNU programs that aren't part of Unix. For example, IBM's AIX offered an AIX Toolbox for Linux Applications with hundreds of GNU software packages (like Bash, GCC, OpenLDAP, and many others) that could be added to an AIX installation to ease the transition between Linux and Unix-based AIX systems.

Proprietary Unix is still alive and well and, with many major vendors promising support for their current releases well into the 2020s, it goes without saying that Unix will be around for the foreseeable future. Also, the BSD branch of the Unix tree is open source, and NetBSD, OpenBSD, and FreeBSD all have strong user bases and open source communities that may not be as visible or active as Linux, but are holding their own in recent server share reports, with well above the proprietary Unix numbers in areas like web serving.

Where Linux has shown a significant advantage over proprietary Unix is in its availability across a vast number of hardware platforms and devices. The Raspberry Pi, popular with hobbyists and enthusiasts, is Linux-driven and has opened the door for an entire spectrum of IoT devices running Linux. We've already mentioned Android devices, autos (with Automotive Grade Linux), and smart TVs, where Linux has large market share. Every cloud provider on the planet offers virtual servers running Linux, and many of today's most popular cloud-native stacks are Linux-based, whether you're talking about container runtimes or Kubernetes or many of the serverless platforms that are gaining popularity.

One of the most revealing representations of Linux's ascendancy is Microsoft's transformation in recent years. If you told software developers a decade ago that the Windows operating system would "run Linux" in 2016, most of them would have laughed hysterically. But the existence and popularity of the Windows Subsystem for Linux (WSL), as well as more recently announced capabilities like the Windows port of Docker, including LCOW (Linux containers on Windows) support, are evidence of the impact that Linux has had—and clearly will continue to have—across the software world.

3.What is Linux Kernel? Is it legal to edit Linux Kernel? Yes, Linux kernel is free and can be modified by anyone.

Answer: Linux kernel is the main component of a Linux operating system (OS) and is the core interface between a computer's hardware and its processes. It communicates between the 2, managing resources as efficiently as possible.

The kernel is so named because—like a seed inside a hard shell—it exists within the OS and controls all the major functions of the hardware, whether it's a phone, laptop, server, or any other kind of computer.

Yes, Kernel is released under General Public Licence (GPL), and anyone can edit Linux Kernel to the extent permitted under GPL. Linux Kernel comes under the category of Free and Open Source Software (FOSS).

4.What is LILO? it is boot loader which permit to select which OS in a machine needs to be used

Answer: LILO (LInux LOader) is a boot loader (a small program that manages a dual boot) for use with the Linux operating system. Most new computers are shipped with boot loaders for some version of Microsoft Windows or the Mac OS. If a computer is to be used with Linux, a special boot loader must be installed. LILO is the most popular boot loader among users who employ Linux as their main, or only, operating system.

When a computer is powered-up or restarted with LILO installed in the usual manner, the basic input/output system (BIOS) performs some initial tests and then transfers control to the Master Boot Record (MBR) where LILO resides. The primary advantage of LILO is the fact that it allows for fast boot up of Linux when installed in the MBR. Its main limitation is the fact that not all computers tolerate modification of the MBR. In these situations, there are alternative approaches for using LILO, but it takes longer.

There are several boot loaders other than LILO that can be used to boot Linux into a computer's memory, such as LOADLIN (LOAD LINux) and GRUB (GRand Unified Bootloader).

5.What are the basic components of Linux? The kernel, X Server, Applications, Desktop environment

Answer: Every OS has component parts, and the Linux OS also has the following components parts:

- **Bootloader**. Your computer needs to go through a startup sequence called booting. This boot process needs guidance, and your OS is the software in control throughout the boot process. When you start your computer the bootloader for your operating system kickstarts the process.
- **OS Kernel**. You can call the kernel the part of the operating system which is the “closest” to your computing hardware as it is the part which controls the CPU, access to memory and any peripheral

devices. It is the “lowest” level at which your operating system works.

- **Background services**. Called “daemons” in Linux, these small applications act as servants in the background, ensuring that key functions such as scheduling, printing and multimedia function correctly. They load after you have booted up, or when you have logged into your computer.
- **OS Shell.** You need to be able to tell our operating system what to do, and this is the goal of the shell. Also known as the command line, it is a facility which lets you instruct your OS using text. However few people nowadays are familiar with command line code, and it once used to put people off using Linux. This changed because a modern distribution of Linux will use a desktop shell just like Windows.
- **Graphics server**. This provides a graphical sub-system that renders images and shapes on your computer monitor. Linux uses a graphical server called “X” or “X-server”.
- **Desktop environment**. You can’t interact with the graphical server directly. Instead you need software that can drive the server. This is called a desktop

environment in Linux and there are plenty of options including KDE, Unity and Cinnamon. A desktop environment is usually bundled with a number of applications including file and web browsers plus a couple of games.

- **Applications**. Obviously, the desktop environment which is bundled with your Linux OS or which you choose to install cannot cater for every application need, there are too many. Individual applications, however, can and there are thousands for Linux just like Windows and Apple's OS X has thousands of applications. Most Linux distros have app stores which help you find and install apps, for example Ubuntu Software which comes with Ubuntu.

It's worth noting that Ubuntu's application repository, the Ubuntu software center, is a great place to look around for Linux applications, both free to use and paid to use.

6.Which are the Shells used in Linux? It is called bash

Answer: If you now understand what a kernel is, what a shell is, and why a shell is so important for Linux systems, let’s move on to learning about the different types of shells that are available.

Each of these shells has properties that make them highly efficient for a specific type of use over other shells. So let us discuss the different types of shells in Linux along with their properties and features.

1. The Bourne Shell (sh)

Developed at AT&T Bell Labs by Steve Bourne, the Bourne shell is regarded as the first UNIX shell ever. It is denoted as sh. It gained popularity due to its compact nature and high speeds of operation.

This is what made it the default shell for Solaris OS. It is also used as the default shell for all Solaris system administration scripts. Start reading about shell scripting here.

However, the Bourne shell has some major drawbacks.

- It doesn’t have in-built functionality to handle logical and arithmetic operations.

- Also, unlike most different types of shells in Linux, the Bourne shell cannot recall previously used commands.
- It also lacks comprehensive features to offer a proper interactive use.

The complete path-name for the Bourne shell is /bin/sh and /sbin/sh. By default, it uses the prompt # for the root user and *$* for the non-root users.

2. The GNU Bourne-Again Shell (bash)

More popularly known as the Bash shell, the GNU Bourne-Again shell was designed to be compatible with the Bourne shell. It incorporates useful features from different types of shells in Linux such as Korn shell and C shell.

It allows us to automatically recall previously used commands and edit them with help of arrow keys, unlike the Bourne shell.

The complete path-name for the GNU Bourne-Again shell is /bin/bash. By default, it uses the prompt *bash-VersionNumber#* for the root user and *bash-VersionNumber$* for the non-root users.

3. The C Shell (csh)

The C shell was created at the University of California by Bill Joy. It is denoted as csh. It was developed to include useful programming features like in-built support for arithmetic operations and a syntax similar to the C programming language.

Further, it incorporated command history which was missing in different types of shells in Linux like the Bourne shell. Another prominent feature of a C shell is “aliases”.

The complete path-name for the C shell is /bin/csh. By default, it uses the prompt *hostname#* for the root user and *hostname%* for the non-root users.

4. The Korn Shell (ksh)

The Korn shell was developed at AT&T Bell Labs by David Korn, to improve the Bourne shell. It is denoted as ksh. The Korn shell is essentially a superset of the Bourne shell.

Besides supporting everything that would be supported by the Bourne shell, it provides users with new functionalities. It allows in-built support for arithmetic operations while offereing interactive features which are similar to the C shell.

The Korn shell runs scripts made for the Bourne shell, while offering string, array and function manipulation similar to the C programming language. It also supports scripts which were written for the C shell. Further, it is faster than most different types of shells in Linux, including the C shell.

The complete path-name for the Korn shell is /bin/ksh. By default, it uses the prompt *#* for the root user and *$* for the non-root users.

5. The Z Shell (zsh)

The Z Shell or zsh is a sh shell extension with tons of improvements for customization. If you want a modern shell that has all the features a much more, the zsh shell is what you're looking for.

7.What is Swap Space? it is the space used in the disk instead of the memory

Answer: A computer has a sufficient amount of physical memory but most of the time we need more so we swap some memory on disk. Swap space is a space on a hard disk that is a substitute for physical memory. It is used as virtual memory which contains process memory images. Whenever our computer runs short of physical memory it uses its virtual memory and stores information in memory on disk. Swap space helps the computer's operating system in pretending that it has more RAM than it actually has. It is also called a swap file. This interchange of data between virtual memory and real memory is called swapping and space on disk as "swap space".

Virtual memory is a combination of RAM and disk space that running processes can use. **Swap space** is the **portion of virtual memory** that is on the hard disk, used when RAM is full.

Swap space can be useful to computers in various ways:

- It can be used as a single contiguous memory which reduces I/O operations to read or write a file.

- Applications that are not used or are used less can be kept in a swap file.
- Having sufficient swap files helps the system keep some physical memory free all the time.
- The space in physical memory which has been freed due to swap space can be used by OS for some other important tasks.

Operating systems such as Windows, Linux, etc systems provide a certain amount of swap space by default which can be changed by users according to their needs. If you don’t want to use virtual memory you can easily disable it all together but in case if you run out of memory then the kernel will kill some of the processes in order to create a sufficient amount of space in physical memory. So it totally depends upon the user whether he wants to use swap space or not.

8.What is the difference between BASH and DOS? in bash / is a directory while in DOS it is \. Bash is case sensitive

Answer: Bash is a powerful command shell and scripting language developed from the Bourne shell long used on UNIX systems and which conforms to the same POSIX standard as Korn Shell. However Bash implementations can be found on other operating systems now. Bash supports array variables, shell functions and very flexible substitution. It is actively developed and new features are added regularly.

By DOS I assume you mean the original Microsoft command shell. It only runs on DOS It has hardly changed since version 3.3 which came out about 30 years ago. It's substitution features are clumsy and weak, it's really just a launcher for other DOS binaries. Trying to write DOS scripts is a nightmare. If you are scripting for Microsoft platforms use powershell instead. Or Bash.

9.What command would you use to check how much memory is being used by Linux? free command can be used

Answer: On linux, there are commands for almost everything, because the gui might not be always available. When working on servers only shell access is available and everything has to be done from these commands. So today we shall be checking the commands that can be used to check memory usage on a linux system. Memory include RAM and swap.

It is often important to check memory usage and memory used per process on servers so that resources do not fall short and users are able to access the server. For example a website. If you are running a webserver, then the server must have enough memory to serve the visitors to the site. If not, the site would become very slow or even go down when there is a traffic spike, simply because memory would fall short. Its just like what happens on your desktop PC.

1. free command

The free command is the most simple and easy to use command to check memory usage on linux.

2. /proc/meminfo

The next way to check memory usage is to read the /proc/meminfo file. Know that the /proc file system does not contain real files. They are rather virtual files that contain dynamic information about the kernel and the system.

3. vmstat

The vmstat command with the s option, lays out the memory usage statistics much like the proc command.

4. top command

The top command is generally used to check memory and cpu usage per process. However it also reports total memory usage and can be used to monitor the total RAM usage. The header on output has the required information.

5. htop

Similar to the top command, the htop command also shows memory usage along with various other details.

10.Explain file permission in Linux.

Answer: Every file and directory in your UNIX/Linux system has following 3 permissions defined for all the 3 owners discussed above.

Read: This permission give you the authority to open and read a file. Read permission on a directory gives you the ability to lists its content.

Write: The write permission gives you the authority to modify the contents of a file. The write permission on a directory gives you the authority to add, remove and rename files stored in the directory. Consider a scenario where you have to write permission on file but do not have write permission on the directory where the file is stored. You will be able to modify the file contents. But you will not be able to rename, move or remove the file from the directory.

Execute: In Windows, an executable program usually has an extension ".exe" and which you can easily run. In Unix/Linux, you cannot run a program unless the execute permission is set. If the execute permission is not set, you might still be able to see/modify the program code(provided read & write permissions are set), but not run it.

r = read permission

w = write permission

x = execute permission

– = no permission