**LINUX FUNDAMENTALS**

**Generic format of printf:** int printf(const char \*\_format,…);

 f**ormat**: A string that specifies how subsequent arguments (if any) are converted for output. It can include:

* Text to be printed as is.
* Format specifiers that begin with %, indicating how to format the corresponding argument.

 **...**: The ellipsis indicates that printf can take a variable number of arguments, which correspond to the format specifiers in the format string.

**COMPILATION PROCESS:**

* Source program🡪pre-processing(pp)🡪Compiler(ccl)🡪Assembler(as)🡪Linker(ld)🡪exe
* The source program will be in the .c format. **Ex:** hello.c
* At the time of pre-processing the processer handles directives that start with # symbol and form transformations and return the file in .i format. **Ex:**hello.i (modified source program)
* At the time of compilation process it will check the syntax errors and return the file in .s format. **Ex**: hello.s (assembly program)
* At the assembler, the assembler converts the assembly code to binary code and return it in the .o format. **Ex:** hello.o (Relocatable object programs)
* At the linker it combines one or more object files into one executable file and gets executed and the final format is .obj or .exe

**GCC---**gcc is a compiler

**Multhi-threading:** Multiple threads will be running at a time in the same process.

**Multi-processing:** Muttiple processors are running different tasks at a time.

**Muthi-tasking:** Multiple tasks running in a computer or OS at a time.

**LINUX:**Linux is an open-source operating system kernel, which means it’s the core part of an operating system that manages hardware and software resources.

**STRUCTURE OF THE OS:**

There are 3 parts in the OS:

1.kernal

2.shell

3.user

**1.KERNAL:**

* Kernal is the core part of the operating system that manages hardware, memory, processes, and system calls.
* The linux kernal handles communication between the software and the hardware.

**2.SHELL:**

* It acts as a interface between kernal and the user.

**BOOTING PROCESS:**

* The process of turning on the computer and the OS.
* It is the first step that involves switching the power ON. ...
* BIOS: Power On Self Test. It is an initial test performed by the BIOS. ...
* Loading of OS. In this step, the operating system is loaded into the main memory. ...
* System Configuration. ...
* Loading System Utilities. ...
* User Authentication.

**RUN LEVELS:** Unix system has several modes of operation called system states.(or run levels).

* 0 : shutdown state
* 1 : administrative state
* S or s : single user state
* 2 : multi-user
* 6 : stop & rebot state
* The states are passed to the init program.

**FILES:**

* A file a collection of data.
* Devices are also treated as special files.
* UINX stores all files in an identical manner.
* Internally each file is assigned a unique identification number called Inode (Information Node).

**SWITCH CASE:**

**How switch is internally operated?**

* In switch all the cases are converted into the integer constants.
* Switch case use **BINARY SEARCH** format for executing that’s why it is faster than the else if statement.
* IN UNIX OR LINUX EVERYTHING WILL BE CONVERTED INTO THE BINARY FORMAT THAT’S WHY IT IS FASTER THEN THE WINDOWS.

**Unix recognizes the following files:**

* Regular, Directory, Executable, Symbolic files, Device specific files, Named pipe.

**FILE NAMING CONVENTIONS:**

* Maximum file length depends upon the kernal configuration. In our system it is 255.
* No concept of Primary and Secondary name.
* Case Sensitive
* Name may contain alphabets, digits, dots and underscores.
* Embedded space and tab names are not allowed.

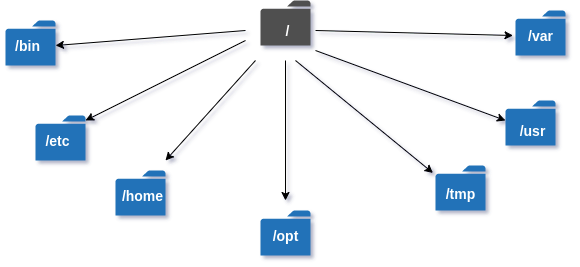
**Commands for Creating a file :**

* **TOUCH:** By using touch command we can create a file.
* **Ex:** touch test.txt or touch /path\_to\_directory/file\_name (for creating in a specific directory)
* **LS—LIST:**
* Ls command lists the contents of the directory and confirm that file was created.
* **CAT:**
* Cat command is used to show the contents of the file if the file exists. If the file does not exists the it will create the file. **Ex:** cat test1.txt
* **cat > test2.txt** --------The operater > indicates place the input into the test2 instead of displaying into the screen.

**DIRECTORIES IN LINUX:**

* A directory is a container that is used to contain folders and files.

Structure of directory:



/bin: binary or executable programs

/etc: system configuration files

/home: home directory. (default current directory)

/opt: optional or third-party software

/tmp: temporary space,typically cleared on reboot

/usr: user related programs

/var: log files

/lib: it contains kernal modules and an shared library

/lost+found: it is used to find recovered bits of corrupted files

**FILES AND INODES:**

A file consists of exactly one inode, and zero or more data blocks. An inode is a structure used to maintain information about the file.

**LINKS IN LINUX:**

* A link in linux is a pointer to a file. Like pointers in any programming languages, links in linux are ponters pointing to a file or a directory.

Creating links is a kind of shortcuts to access a file.

1.Hard Link

2.Soft Link

**1.Hard link:** A hard link adds an additional pathname to reference a single file.

* Each hard linked file is assigned the same inode value as the original, therefore they reference the same file location.
* **ls -l commad** shows all the links with the link column shows number of links.
* Removing any link, just reduces the link count, but doesn’t affect other links.
* The **rm command** decrements the link count.

**Syntax:** $ ln [original filename] [link name]

**2.Soft link or Symbolic link:** A symbolic points to another file.

* A soft link is similar to the **file shortcut** feature which is used in windows operating system.
* Each soft linked file contains a separate inode value that points to the original file.
* If the original file is deleted then it is not possible to access the other files.
* Soft links contains the path for original file and not the contents.

**Syntax:** $ ln -s [original filename] [link name]