**Identity access management:**

It is designed to make sure that right people can access what they need while keeping your systems and data secure from unauthorized users.

**Identity:**identity will changes all the time when the changes will occur.

**Management:**Managing all the identities from the centralized resources into a single resource.

**Why idm:** for each and every small company especially who are trying to break into a competitive industry,it can be tempting to put all IT companies like IDM.

It is a time consuming process requires a dedicate IT staff and usually involves an financial investment.

**Birthright access:** the person who belongs to a staff category they will be granted to login account and access to email as birthright.

**Entities:** Entities have to authenticate themselves to service provider inorder to use their services

An entity provides personally identifiable information that unquiely identifiess it to an service provider

**Reconciliation**: syncing of the information

**Trusted reconciliation:** the person whom you belive is called trusted resource.if we receive the information from trusted resource is called trusted reconciliation

**Target reconciliation:** if we receive the information from the targeted resource is called target reconciliation.

**Identity and access provisioning life cycle:**

**Provisioning:**providing the information to the target system to provide an account.

**Review**:checking the accounts periodically,disable the inactive accounts,check for excercive and creeping privilages.

**Revocation:**disable the account as soon as an employee leaves in an organisation.

**Protocols:** A protocol is a set of rules to govern the data transfer between the devices. The rules are used For compressing the data,For sending device to indicate that it has finished sending a message, For receiving device to indicate that it has received a message.

### 1. Transmission control Protocol (TCP)

This is a **communication protocol** that computers uses to communicate over a network.

### 2. Internet Protocol (IP)

Internet protocol is **addressing protocol**. It is always used together with TCP. IP addresses of packet, routes them through different nodes and networks until it reaches its final destination.

### 3. Internet Address Protocol (IP Address)

This is the address that identifies a computer on a network using TCP/IP. This address is usually supplied by a Internet Service Provider.

### 4. Post office Protocol (POP)

This is used to **receive incoming E-mail** .

### 5. Simple mail transport Protocol (SMTP)

This protocol is used for **sending and distributing outgoing E-Mail** .

### 6. File Transfer Protocol (FTP)

This is a system that allows users to **transfer files** from one computer to another computer. Files that can be transfered may include program files, text files and multimedia files ect. This method of file transfer is faster than that using HTTP.

### 7. Hyper Text Transfer Protocol (HTTP)

HTTP is used to transfer a hyper text between two or more computers. Hyper text is the text that is coded using the language called HTML.   
8. Ethernet

Ethernet is a most popular protocol used for LAN communication. It transfer the information in digital packets.

### 9. Telnet

Telnet is a set of rules used to connect one computer to another computer. The process of this connection is called as **remote login**.

**Port:**

A port number is the logical address of each application or process that uses a network or the Internet to communicate. A port number uniquely identifies a network-based application on a computer. Each application/program is allocated a 16-bit integer port number.

The range of port numbers from 1024 to 49151 are the Registered, also known as User ports.

A port number primarily aids in the transmission of data between a network and an application.

The port numbers in the range from 0 to 1023 are the well-known, also known as System ports.

commonly used Internet/network services are allocated with global port numbers such as Port Number 80 for HTTP, 23 for Telnet and 25 for SMTP.

20,21-FTP,22-secureshell,23-telenet,25-SMTP,53-DNS,67,68-Dynamic host configuration protocol,80-HTTP,110-POP3,

1. nameserver,67-bootstrap,69-trival file transfer

**Directory:**A tree structure it will store the data.

**LDAP**:lightweight directory access protocol.

**Azure Active** directory:store the all user information.

**OPERATING SYSTEM:**

it is a powerful, and usually large, program that controls and manages the hardware and other software on a computer.

All computers and computer-like devices require operating systems, including your laptop, tablet, desktop, smartphone, smartwatch, and router.

**Examples are**

Laptops, tablets, and desktop computers all run operating systems that you've probably heard of. Some examples include versions of Microsoft Windows (like [Windows 10](https://www.lifewire.com/windows-10-2626217), Windows 8, Windows 7, Windows Vista, and Windows XP), [Apple's macOS](https://www.lifewire.com/what-is-macos-4691239) (formerly OS X), [Chrome OS](https://www.lifewire.com/google-chrome-os-1616448), BlackBerry Tablet OS, and [flavors of Linux,](https://www.lifewire.com/unix-flavors-list-4094248) an open-source operating system.

Operating system is software that is required in order to run application programs and utilities.

**How Operating System Work**

When you turn on the computer, the operating system program is loaded into the main memory. This program is called the kernel. Once initialized, the system program is prepared to run the user programs and permits them to use the hardware efficiently. different types of hardware configurations from thousands of vendors and accommodates thousands of different *I/O* devices like [printer](https://ecomputernotes.com/fundamental/input-output-and-memory/what-is-a-printer-and-what-are-the-different-types-of-printers" \o "Printers are Output devices used to prepare permanent Output devices on paper." \t "https://ecomputernotes.com/fundamental/disk-operating-system/_self)s, disk drives, scanners and cameras.

Types:

1. ****Batch Operating System –****  
   This type of operating system does not interact with the computer directly. There is an operator which takes similar jobs having same requirement and group them into batches. It is the responsibility of operator to sort the jobs with similar needs.
2. ****Time-Sharing Operating Systems –****  
   Each task is given some time to execute, so that all the tasks work smoothly. Each user gets time of CPU as they use single system. These systems are also known as Multitasking Systems. The task can be from single user or from different users also.
3. ****Distributed Operating System –****  
   These types of operating system is a recent advancement in the world of computer technology and are being widely accepted all-over the world and, that too, with a great pace. These are referred as ****loosely coupled systems**** or distributed systems.
4. ****Network Operating System –****  
   These systems run on a server and provide the capability to manage data, users, groups, security, applications, and other networking functions. These type of operating systems allow shared access of files, printers, security, applications, and other networking functions over a small private network.
5. ****Real-Time Operating System –****  
   These types of OS serves the real-time systems. The time interval required to process and respond to inputs is very small. This time interval is called ****response time.****

**Two types:**

****Hard Real-Time Systems:****  
These OS are meant for the applications where time constraints are very strict and even the shortest possible delay is not acceptable.

****Soft Real-Time Systems:****  
These OS are for applications where for time-constraint is less strict.

Unix:

## What is Unix ?

The Unix operating system is a set of programs that act as a link between the computer and the user.

The computer programs that allocate the system resources and coordinate all the details of the computer's internals is called the **operating system** or the **kernel**.

Users communicate with the kernel through a program known as the **shell**. The shell is a command line interpreter; it translates commands entered by the user and converts them into a language that is understood by the kernel.

There are various Unix variants available in the market. Solaris Unix, AIX, HP Unix and BSD are a few examples.

Unix versions:

The main concept that unites all the versions of Unix is the following four basics −

**Kernel** − The kernel is the heart of the operating system. It interacts with the hardware and most of the tasks like memory management, task scheduling and file management.

**Shell** − The shell is the utility that processes your requests. When you type in a command at your terminal, the shell interprets the command and calls the program that you want. The shell uses standard syntax for all commands. C Shell, Bourne Shell and Korn Shell are the most famous shells which are available with most of the Unix variants.

**Commands and Utilities** − There are various commands and utilities which you can make use of in your day to day activities. **cp**, **mv**, **cat** and **grep**, etc. are few examples of commands and utilities. There are over 250 standard commands plus numerous others provided through 3rd party software. All the commands come along with various options.

**Files and Directories** − All the data of Unix is organized into files. All files are then organized into directories. These directories are further organized into a tree-like structure called the **filesystem**.

02-09-2020

Shell script:

**A shell script is a [computer program](https://en.wikipedia.org/wiki/Computer_program" \o "Computer program) designed to be run by the [Unix shell](https://en.wikipedia.org/wiki/Unix_shell" \o "Unix shell), a [command-line interpreter](https://en.wikipedia.org/wiki/Command-line_interpreter" \o "Command-line interpreter).**

**which allows you to enter commands for execution by the system.**

**When a terminal window is opened on a linux computer, it starts the shell program which presents an interface to enter commands. This interface is known as the command line interface.**

**When a command is entered, it is executed by the shell and the output is displayed on the screen.**

//the shell can also execute commands stored in a file. This mode of execution is known as ****shell scripting****,

**We can perform Typical operations performed by shell scripts include file manipulation, program execution, and printing text**

Shell script can also be used to provide linkage for already existing programs.

list of commands executed sequentially.

**File Manipulation-programs that read and write them as in the form of files and directories**

**Program execution- a program when executed a special instance is provided by the system to the process**

**Printing text-**

## **Types of Shells**

There are two major types of shells in Unix. These are:

### **Bourne Shell**

This is the default shell for version 7 Unix. The character $ is the default prompt for the bourne shell. The different subcategories in this shell are Korn shell, Bourne Again shell, POSIX shell etc. Syn:bsh

### **C Shell**

It is a command processor that is run in a text window. The character % is the default prompt for the C shell. File commands can also be read easily by the C shell

Syn:csh

List of coomands:

## Files and Directories

These commands allow you to create directories and handle files.

Given below is the list of commands in Files and Directories.

|  |  |
| --- | --- |
| **Sr.No.** | **Command & Description** |
| 1 | **cat**  Displays File Contents |
| 2 | **cd**  Changes Directory to dirname |
| 3 | **chgrp**  Changes file group |
| 4 | **chmod**  Changes permissions |
| 5 | **cp**  Copies source file into destination |
| 6 | **file**  Determines file type |
| 7 | **find**  Finds files |
| 8 | **grep**  Searches files for regular expressions |
| 9 | **head**  Displays first few lines of a file |
| 10 | **ln**  Creates softlink on oldname |
| 11 | **ls**  Displays information about file type |
| 12 | **mkdir**  Creates a new directory dirname |
| 13 | **more**  Displays data in paginated form |
| 14 | **mv**  Moves (Renames) an oldname to newname |
| 15 | **pwd**  Prints current working directory |
| 16 | **rm**  Removes (Deletes) filename |
| 17 | **rmdir**  Deletes an existing directory provided it is empty |
| 18 | **tail**  Prints last few lines in a file |
| 19 | **touch**  Updates access and modification time of a file |

## Manipulating data

The contents of files can be compared and altered with the following commands.

Given below is the list of commands in Manipulating data.

|  |  |
| --- | --- |
| **Sr.No.** | **Command & Description** |
| 1 | **awk**  Pattern scanning and processing language |
| 2 | **cmp**  Compares the contents of two files |
| 3 | **comm**  Compares sorted data |
| 4 | **cut**  Cuts out selected fields of each line of a file |
| 5 | **diff**  Differential file comparator |
| 6 | **expand**  Expands tabs to spaces |

|  |  |
| --- | --- |
| **Sr.No.** | **Command & Description** |
| 1 | **ftp**  File transfer program |
| 2 | **rcp**  Remote file copy |
| 3 | **rlogin**  Remote login to a Unix host |
| 4 | **rsh**  Remote shell |
| 5 | **tftp**  Trivial file transfer program |
| 6 | **telnet**  Makes terminal connection to another host |
| 7 | **ssh**  Secures shell terminal or command connection |
| 8 | **scp**  Secures shell remote file copy |
| 9 | **sftp**  Secures shell file transfer program |

|  |  |
| --- | --- |
| **Sr.No.** | **Command & Description** |
| 1 | **chfn**  Changes your finger information |
| 2 | **chgrp**  Changes the group ownership of a file |
| 3 | **chown**  Changes owner |
| 4 | **date**  Prints the date |
| 5 | **determin**  Automatically finds terminal type |
| 6 | **du**  Prints amount of disk usage |
| 7 | **echo**  Echo arguments to the standard options |
| 8 | **exit**  Quits the system |
| 9 | **finger**  Prints information about logged-in users |
| 10 | **groupadd**  Creates a user group |
| 11 | **groups**  Show group memberships |
| 12 | **homequota**  Shows quota and file usage |
| 13 | **iostat**  Reports I/O statistics |
| 14 | **kill**  Sends a signal to a process |
| 15 | **last**  Shows last logins of users |
| 16 | **logout**  Logs off Unix |
| 17 | **lun**  Lists user names or login ID |
| 18 | **netstat**  Shows network status |
| 19 | **passwd**  Changes user password |
| 20 | **passwd**  Changes your login password |
| 21 | **printenv**  Displays value of a shell variable |
| 22 | **ps**  Displays the status of current processes |
| 23 | **ps**  Prints process status statistics |
| 24 | **quota -v**  Displays disk usage and limits |
| 25 | **reset**  Resets terminal mode |
| 26 | **script**  Keeps script of terminal session |
| 27 | **script**  Saves the output of a command or process |
| 28 | **setenv**  Sets environment variables |
| 30 | **stty**  Sets terminal options |
| 31 | **time**  Helps time a command |
| 32 | **top**  Displays all system processes |
| 33 | **tset**  Sets terminal mode |
| 34 | **tty**  Prints current terminal name |

03-09-2020

Login: at first weather the user is an existing user then the person have to login directly with the user name and password and the it will take us directly to the desktop

Pwd:present working directory

/(slash):root directory of the file

USR:unix system resource

Files:

**storing** large amounts of information.

All data in Unix is organized into files. All files are organized into directories.

These directories are organized into a tree-like structure called the file system.

At the very top of the file system is a directory called “root” which is represented by a “/”.

**Directories or Files and their description –**

* **/ :** The slash / character alone denotes the root of the filesystem tree.
* **/bin :** Stands for “binaries” and contains certain fundamental utilities, such as ls or cp, which are generally needed by all users.
* **/boot :** Contains all the files that are required for successful booting process.
* **/dev :** Stands for “devices”. Contains file representations of peripheral devices and pseudo-devices.
* **/etc :** Contains system-wide configuration files and system databases.
* **/home :** Contains the home directories for the users.
* **/lib :** Contains system libraries, and some critical files such as kernel modules or device drivers.
* **/media :** Default mount point for removable devices, such as USB sticks, media players, etc.
* **/mnt :** Stands for “mount”. Contains filesystem mount points. These are used, for example, if the system uses multiple hard disks or hard disk partitions. It is also often used for remote (network) filesystems, CD-ROM/DVD drives, and so on.
* **/proc :** procfs virtual filesystem showing information about processes as files.
* **/root :** The home directory for the superuser “root” – that is, the system administrator.
* **/tmp :** A place for temporary files.
* **/usr :** Originally the directory holding user home directories,its use has changed.
* **/usr/bin :** This directory stores all binary programs distributed with the operating system not residing in /bin, /sbin or (rarely) /etc.
* **/usr/include :** Stores the development headers used throughout the system. Header files are mostly used by the **#include** directive in C/C++ programming language.
* **/usr/lib :** Stores the required libraries and data files for programs stored within /usr or elsewhere.
* **/var :** A short for “variable.” A place for files that may change often – especially in size, for example e-mail sent to users on the system, or process-ID lock files.
* **/var/log :** Contains system log files.
* **/var/mail :** The place where all the incoming mails are stored. Users (other than root) can access their own mail only. Often, this directory is a symbolic link to /var/spool/mail.
* **/var/spool :** Spool directory. Contains print jobs, mail spools and other queued tasks.
* **/var/tmp :** A place for temporary files which should be preserved between system reboots.

**Ordinary files –**  contains data, text, or program instructions.

* Used to store your information,(like text,images)
* Always located within/under a directory file.
* Do not contain other files.
* In long-format output of ls -l, this type of file is specified by the “-” symbol.

1. **Directories –** Directories store both special and ordinary files.

If you have 10 files in a directory, there will be 10 entries in the directory. Each entry has two components.  
(1) The Filename  
(2) A unique identification number for the file or directory (called the inode number)

· All files are descendants of the root directory, ( named / ) located at the top of the tree.

In long-format output of ls –l , this type of file is specified by the “d” symbol.

**3. Special Files –** Used to represent a real physical device such as a printer, tape drive or terminal, used for Input/Output (I/O) operations. **Device or special files** are used for device Input/Output(I/O) on UNIX and Linux systems. They appear in a file system just like an ordinary file or a directory.

**4. Pipes –** UNIX allows you to link commands together using a pipe. The pipe acts a temporary file which only exists to hold data from one command until it is read by another.A Unix pipe provides a one-way flow of data.For example: **who | wc -l**

In long-format output of ls –l , named pipes are marked by the “p” symbol.

**5. Sockets –** A Unix socket (or Inter-process communication socket) is a special file which allows for advanced inter-process communication. A Unix Socket is used in a client-server application framework. .

In long-format output of ls -l, Unix sockets are marked by “s” symbol.

**6. Symbolic Link –** Symbolic link is used for referencing some other file of the file system.Symbolic link is also known as Soft link. It contains a text form of the path to the file it references. In long-format output of ls –l , Symbolic link are marked by the “l” symbol (that’s a lower case L).

Dot matrix file:

Help:man command

Ls -lr:

Long listing files:

## **Long Listing Of Files**

More information about a file, such as its size, can be obtained by using the long listing option of the **ls** command:

|  |  |
| --- | --- |
| *Type:* | **ls -l** |

This produces output in the following format:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *-rw-r--r--* | 1 | abc2 | cc\_user | 41 | Apr 12 | 09:14 | chapter1 |
| *-rw-r--r--* | 1 | abc2 | cc\_user | 11 | Apr 14 | 10:22 | chapter2 |
| *-rw-r--r--* | 1 | abc2 | cc\_user | 18 | May 15 | 15:13 | chapter3 |

where the eight fields represent:

|  |  |
| --- | --- |
| *-rw-r--r--* | the protection associated with the file |
| 1 | number of links to the file |
| abc2 | owner |
| cc\_user | group |
| 41 | byte size of the file |
| Apr 12 | creation date |
| 09:14 | creation time |

Change permission:chmod

04-09-2020

Umask:

the settings of a mask that controls how file permissions are set for newly created files.

umask command is invoked without any arguments, it will display the current mask

The output will be in either [octal](https://en.wikipedia.org/wiki/File_permissions" \l "Numeric_notation" \o "File permissions) or [symbolic](https://en.wikipedia.org/wiki/File_permissions" \l "Symbolic_notation" \o "File permissions) notation, depending on the OS

Syntax:$umask

0022(only executed by the admin)

File permission mode:

There are two modes are there:

**Absolute Mode –** Use numbers to represent file permissions. When you change permissions by using the absolute mode, you represent permissions for each triplet by an octal mode number. Absolute mode is the method most commonly used to set permissions.

**Symbolic Mode –** Use combinations of letters and symbols to add permissions or remove permissions.

0-no permissions(---)

1-execute(--x)

2-write(-w-)

3-write,read only(-wx)

4-read(r--)

5-Read and execute(r-x)

6-Read and write(rw-)

7-Read, write, and execute(rwx)

---------------------------

05-09-2020

Login process:

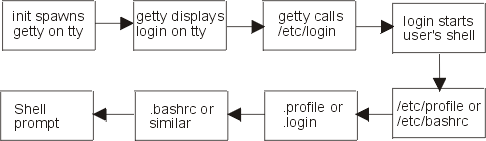
**login is a builtin aredefault command which is used in the corn shell (csh).**

**the functionality of login is hidden from the user. The login command runs automatically upon connection.**

**Steps for login process:**

* **At first the user has to enter with his user name and password.the os confirms weather the user name and the password are correct.**
* **If the username and the password was correct it will automatically create a shell.**

/etc/rc are the init process (script stored in the file **/etc/inittab.** It also controls autonomous processes required by any particular system.

* commands to run and then execute the commands.
* The starting process in the unix is called getty(it will take all the complete login process in the unix)
* The getty will do the internal process like enter username and the password these are the basic details which is going to be filled by the user
* The password which was typed and it will be in the hidden format It will not displayed on the screen.
* **[/etc/passwd](https://www.linuxnix.com/2011/06/linux-password-file-explained-detail.html)** and [/etc/shadow](https://www.linuxnix.com/2011/06/linux-shadow-file-explained-detail.html) are the commands used for checking the credentials,
* After that it reads **/etc/motd** file for displaying content banner message.
* the default files are read the shell reads **user specific login files .profile** or **.login .**
* if password matches it will initiates user properties else getty will terminate login process and re-initiates once again with new login: prompt.
* In the unix/linux falvours . If user failed to enter correct password for three consecutive times,
* the getty process read the user properties like username, UID, GID, home directory,
* getty process reads /etc/profile file for shell related to the information given by the user.
* Now the PS1 prompt is ready for the user to execute their commands.
* [](https://3.bp.blogspot.com/-Z20oRyKsatg/U51eHBXLdQI/AAAAAAAAAOc/G-IIwcnGZlw/s1600/login.png)
* -------------------------(other )-----------------------------------------------------------------------
* **[Init](https://www.tutorialspoint.com/unix_commands/init.htm" \l "3)** is the command and then it is the parent of all processes. Its create processes from a script stored in the file **/etc/inittab** . This file usually has entries which cause **init** to spawn(release) **getty** on each line that users can log in. It processes by any particular system.
* **The getty will displays the login process like**user has to enter with his user name and password**(** getty will do the internal process like enter username and the password these are the basic details which is going to be filled by the user)
* The getty will calls the /etc/login Once user enters his login name getty starts **/ets/login**, this in-turn will prompt for user password.
* The password which was typed and it will be in the hidden format It will not displayed on the screen.
* It will automatically create a shell by its default if he a valid user
* There are default files like **/etc/profile** or **/etc/bashrc are the commands**
* **t**he shell reads **user specific login files .profile** or **.login .**
* At last stage it reads shell specific configuration files **(.bashrc, .bash\_profile** etc. for **bash shell**) of that user which it gets on the users home directory.
* And then finally it will display the shell prompt
* **/etc:**is a hirarchy contains the confiruration files,control the operation of the program
* **.bashrc-**is a file that execute the user logins
* **.bash\_profile** is the personal initialization of the file for configuring the user environment
* **.profile** is the linux commands comes under the system startup file
* .login-initilizes the user environment from the user database from the command line

Search:

Whereis:any services if we want to check the depended files then we can use the whereis command.

Find: we can search the specific file by the name are the extension

It is not a global search

Some of the **whereis parameters :**

-b:search for the only binary files

-m:search for only manual sections

-s:search for only sources

-u:search for unusual entries

-B:change or otherwise limit the places where searches for binary files are located

-M:change or otherwise limit the places where whereis search for manual sections(store the manual files)

-S:change or otherwise limit the places where whereis search for sources

Find:Use find from the command line to locate a specific file by name or extension

**Find commands parameters: we can use the particular files for finding the destination**

//The -O1 optimization is the default and forces find to filter based on filename before running all other tests.

**-01**-search based on the file name first(if I give imoortant data bydefault it will check the important of the file)

**-02-**filename first,thenfile-type(if I want to search the file based on the extensuin)

**-03-**allow to find the automatically reorder the search is based on the efficient use of resources and likehood for success(automatically it will set the destination of ur file where you can find it)

**-maxdepth X-**search the current directory as well as all subdirectories Xlevels deep(we can definr the limit for all subdirectories)

**-iname-**search without regard for test cases(case sensitivie eill not happen)

**-not-**return only results that does not match the testcase

-t**ype f-**search for particular files

**-type d-**search for directories

07-09-2020

Vi editor:

The original code for vi was written by [Bill Joy](https://en.wikipedia.org/wiki/Bill_Joy" \o "Bill Joy) in 1976,

* The vi editior (short for visual editor) is a **text editor** which is available on almost all unix/inux systems
* Vi has no menus but instead used combination of keystrokes inorder to accomplish the commands
* You can use **vi editor to edit**(add,change and delete text) **an existing file** or to **create a new file** from scratch.you can **use this editor** to just **read a text file**
* While working with the vi editor we should know two modes
* **Command mode:** performs the adminstrative tasks such as ssaving a file,executing commands,moving cursor,cutting and pasting lines or words and searching and replacing.in this mode what ever you typed it is interpreted
* **Insert mode:**enables u to insert text into the file.what ever the input we are giving it is interpreted as input and finally it is put on the file
* Escape mode:Line Mode is invoked by typing a colon [:],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.
* The vi always starts in command mode
* To enter text, you must be in insert mode
* To come in inside mode,you simply type **i**
* to get out of insert mode press the ESC key which will put u into a command mode
* Hint:if you are not sure which mode you are in if we oress ESC twice and then you will be in the command mode
* Vi commands are case sensitive
* We create a file in the vi editor
* We give the vi filename as roopa and then enter
* You can get the file name “roopa”[new file]
* Starting a file
* ***vi filename:*** *Creates a new file if it already not exist, otherwise opens existing file.*
* ***vi -R filename*** *: Opens an existing file in read only mode.*
* ***view filename*** *: Opens an existing file in read only mode.*

**Moving within a File(Navigation):**  
To move around within a file without affecting text must be in command mode (press Esc twice). Here are some of the commands can be used to move around one character at a time.

***Commands and their Description***

* **k** : Moves the cursor up one line.
* **j** : Moves the cursor down one line.
* **h** : Moves the cursor to the left one character position.
* **l** : Moves the cursor to the right one character position.
* **0 or |** : Positions cursor at beginning of line.
* **$** : Positions cursor at end of line.
* **W** : Positions cursor to the next word.
* **B** : Positions cursor to previous word.
* **(** : Positions cursor to beginning of current sentence.
* **)** : Positions cursor to beginning of next sentence.
* **H** : Move to top of screen.
* **nH** : Moves to nth line from the top of the screen.
* **M** : Move to middle of screen.
* **L** : Move to bottom of screen.
* **nL** : Moves to nth line from the bottom of the screen.
* **colon along with x** : Colon followed by a number would position the cursor on line number represented by x.

**Control Commands(Scrolling):** There are following useful commands which can used along with **Control Key**:

**Commands and their Description:**

* **CTRL+d** : Move forward 1/2 screen.
* **CTRL+f** : Move forward one full screen.
* **CTRL+u** : Move backward 1/2 screen.
* **CTRL+b** : Move backward one full screen.
* **CTRL+e** : Moves screen up one line.
* **CTRL+y** : Moves screen down one line.
* **CTRL+u** : Moves screen up 1/2 page.
* **CTRL+d** : Moves screen down 1/2 page.
* **CTRL+b** : Moves screen up one page.
* **CTRL+f** : Moves screen down one page.
* **CTRL+I** : Redraws screen.

**Editing and inserting in Files(Entering and Replacing Text):** To edit the file, we need to be in the insert mode. There are many ways to enter insert mode from the command mode.

* **i** : Inserts text before current cursor location.
* **I** : Inserts text at beginning of current line.
* **a** : Inserts text after current cursor location.
* **A** : Inserts text at end of current line.
* **o** : Creates a new line for text entry below cursor location.
* **O** : Creates a new line for text entry above cursor location.
* **r** : Replace single character under the cursor with the next character typed.
* **R** : Replaces text from the cursor to right.
* **s** : Replaces single character under the cursor with any number of characters.
* **S** :Replaces entire line.

**Deleting Characters:** Here is the list of important commands which can be used to delete characters and lines in an opened file.

* **X** Uppercase: Deletes the character before the cursor location.
* **x** Lowercase : Deletes the character at the cursor location.
* **Dw** : Deletes from the current cursor location to the next word.
* **d^** : Deletes from current cursor position to the beginning of the line.
* **d$** : Deletes from current cursor position to the end of the line.
* **Dd** : Deletes the line the cursor is on.

**Copy and Past Commands:** Copy lines or words from one place and paste them on another place by using the following commands.

* **Yy** : Copies the current line.
* **9yy** : Yank current line and 9 lines below.
* **p** : Puts the copied text after the cursor.
* **P** : Puts the yanked text before the cursor
* *-----------------------------*

*09-09-2020*

*Tar files:*

tar stands for tape archive.

Modes

.tar -cvf : for creating tar file

tar -cvf : for creating tar file

tar-xvf : for extracting from tar files.

|  |
| --- |
| tar stands for tape archive. |
| it has three modes for different operations : |
| i.tar -cvf : for creating tar file. |
| ii..tar -cvf : for creating tar file |
| iii. tar-xvf : for extracting from tar files. |

*Df-f*

*Df-k*

*Du-sk*

*Quota*

*Tar -xvfrc*

*10-09-2020*

*Shell scripting:*

****Shell Scripting**** is an open-source computer program designed to be run by the Unix/Linux shell. Shell Scripting is a program to write a series of commands for the shell to execute

What is shell:

****Shell**** is a UNIX term for an interface between a user and an operating system service. Shell provides users with an interface and accepts human-readable commands into the system and executes those commands which can run automatically and give the program’s output in a shell script.

Components:

two components are -

* Kernel
* Shell

A Kernel is at the nucleus of a computer. It makes the communication between the hardware and software possible. While the Kernel is the innermost part of an operating system, a shell is the outermost one.

## Types of Shell

There are two main shells in Linux:

****1****. The ****Bourne Shell****: The prompt for this shell is $ and its derivatives are listed below:

* POSIX shell also is known as sh
* Korn Shell also knew as sh
* ****B****ourne ****A****gain ****SH****ell also knew as bash (most popular)

****2.**** ****The C shell****: The prompt for this shell is %, and its subcategories are:

* C shell also is known as csh
* Tops C shell also is known as tcsh

## How to Write Shell Script in Linux/Unix

****Shell Scripts**** are written using text editors. open a new file to begin typing a shell script or shell programming,

We can create the shell scripting by

1. ****Create a file**** ****using**** a ****vi**** editor(or any other editor).  Name  script file with ****extension .sh****
2. ****Start**** the script with ****#! /bin/sh****
3. Write some code.
4. Save the script file as filename.sh
5. For ****executing**** the script type ****bash filename.sh****

"#!" is an operator called shebang, if we use"#! /bin/sh" the script gets directed to the bourne-shell.

11-09-2020

Looping:

Untill loop:

#!/bin/sh

a=10

until [ $a -lt 10 ]do

echo $a

a=`expr $a + 1`done

BREAK:

#!/bin/sh

a=0

while [ $a -lt 10 ]do

echo $a

if [ $a -eq 5 ]

then

break

fi

a=`expr $a + 1`done

Upon execution, you will receive the following result −

0

1

2

3

4

5

NEASTED FOR LOOP:

[Live Demo](http://tpcg.io/NcYiyO" \t "https://www.tutorialspoint.com/unix/_blank)

#!/bin/sh

for var1 in 1 2 3do

for var2 in 0 5

do

if [ $var1 -eq 2 -a $var2 -eq 0 ]

then

break 2

else

echo "$var1 $var2"

fi

donedone

CONTINUE:

#!/bin/sh

NUMS="1 2 3 4 5 6 7"

for NUM in $NUMSdo

Q=`expr $NUM % 2`

if [ $Q -eq 0 ]

then

echo "Number is an even number!!"

continue

fi

echo "Found odd number"done

Upon execution, you will receive the following result −

Found odd number

Number is an even number!!

Found odd number

Number is an even number!!

Found odd number

Number is an even number!!

Found odd number

11-09-2020

11-09-2020

**ifs:**

IFS IFS stands for Internal Field Separator.

we can assign separator to the IFS like IFS=any separator like , or . etc

Based on the input and separator it will divide the input and print it in the next line

**set** By using set command we can set value of the variable .Eg:set a=hi

**read only** When we give a variable as read only we can not change the value of that variable.Ex.a=10 read only a.We can only print that but we can not change that value.It acts just a constant**.**

**unset** Once we set the variable as unset then we can not use that one.

**Basic Commands**

**B -** To go to the previous word.

G-To go the first line.2G-to go to the second line

0- to go the Starting of the line

H - To go the top of the file

!vi - To go to the top of the file

14-09-2020 Loops in unix for(initialization;condition;updation)

do

statements

done

until[condition]

do

stmts

done

Input redirection Suppose if you want to pass the input from one file to script then this it is called input redirection

syntax:

Create any file with data without any spaces.

while read VAL

do

case $VAL in

1)echo "this is a";;

2)echo "this is b";;

esac

done < Demo(filename)

Output redirection If you want to dsipaly the output of one into another file then that is called output redirection.

syntax:

./filename.sh > another\_file\_name

and display the output by cat another\_file\_name

Suppose if you want to display the error in the file as a output then use $./filname.sh > another\_file\_name 2>&1

Funtions in unix 1)function without parameters and without return type.

syntax:function\_name () {

statements

}

function\_name

2)with arguements and without return type

syntax:function\_name () {

echo "$1"

echo "$2"

}

function\_name arg1 arg2

3)function with arguements and return type

syntax:function\_name () {

echo "$1"

echo "$2"

return var

}

function\_name arg1 arg2

echo var

15-09-2020

echo -n \ :To print the output in a single along with the spaces.

heredoc command :A block of code or text which can be redirected to the command script or interactive program is called here document or HereDoc.

syntax:

command << somtext

any text

sometext

crontab command :This command will decides the scheduling of the processes.

syntax: m h dom mon dow command

m for minutes:range 0 to 59

h for hour:range 0 to 23

dom for day of month:range 0 to 31

dow for day of week:range 0 to 6

command means any unix command

16-09-2020

heredoc command There are some set programs or processes where it requires user interaction in order to complete the process.But we can automate that process by using this heredoc.

For example if we connect to the database then we need to perform some operations like displaying the list of tables,or create view etc.We can automate these process by usnig heredoc in my

mysql.

Suppose if you want to transfer the file from one system to another system then we can also automate this one by using sftp protocol.

17-09-2020 heredoc command How to create a directory and file on other machines and how to delte those files.And how to go to .ssh directory and sftp.