“Advanced C Programming”

eDESD - 2021

Day 1

1. **Introduction to Linux , vi editor and C**
2. Data Types & Operators in C

1. Introduction to Linux, Vi and C

Introduction to Linux - Outline

- What is Unix?
- What is Linux?
- Linux Distributions
- Linux File System
- Shell and Basic Commands
- vim (vi improved) Editor (vi – visual interactive)

What is Unix?

- Initially, Named as “UNiplexed Information Computing System (UNICS)”
- Changed the name to “UNIX”
- Developed in 1969 at AT&T’s Bell Labs by
  - Ken Thompson - UNIX
  - Dennis Ritchie - C Language
  - Douglas Mcllroy - Pipes
- A multi-tasking and multi-user Operating System
  - You can have many users logged into a system simultaneously, each running many programs.
  - **00:00:00 Hours, Jan 1, 1970** is time zero for UNIX. It is also called as **epoch.**

Reference: http://en.wikipedia.org/wiki/History_of_Unix

UNIX

Ken Thompson (Sitting) and Dennis Ritchie (Standing) working together at a **PDP-11**

Ken Thompson and Dennis Ritchie

Reference: http://en.wikipedia.org/wiki/History_of_Unix

What is Linux?

- A clone of UNIX, Developed in **1991** by **Linus Torvalds,** a Finnish graduate student **(It was his personal project)**
- Inspired by and replacement of "**Minix** (Mini Unix by Tanenbaum for education)"
- Linus + Minix → Linux
- First kernel (v1.0) was released in **1994** (Under GNU general public license)
- Consist of
  - Linux Kernel
  - GNU (GNU is Not Unix) Software
  - Software Package management & Others

Linux is everywhere

- Originally developed for X86-32 Bit
- **Internet** was built around UNIX
- **Android & MAC OS** are developed based on Linux kernel
- Ported to other architectures.
  - IBM PowerPC
  - Mobile Phones -Nokia N810, **Google Nexus** (Ubuntu), etc.
  - Routers, GPS
  - **Robo**

Check **www.top500.org** to know the power of Linux

Why Linux is everywhere?

- Open, Free or Cheap
- Scalable and Portable
  - **Scalable** – In terms of processor count, Number of users, Memory size, I/O, Resource management etc..
  - **Portable** – It can work efficiently on anything from wristwatch to World's fastest Supercomputer
- Multiuser and multitasking
- Robust – A cluster or a Server can run for years without rebooting → Reliable

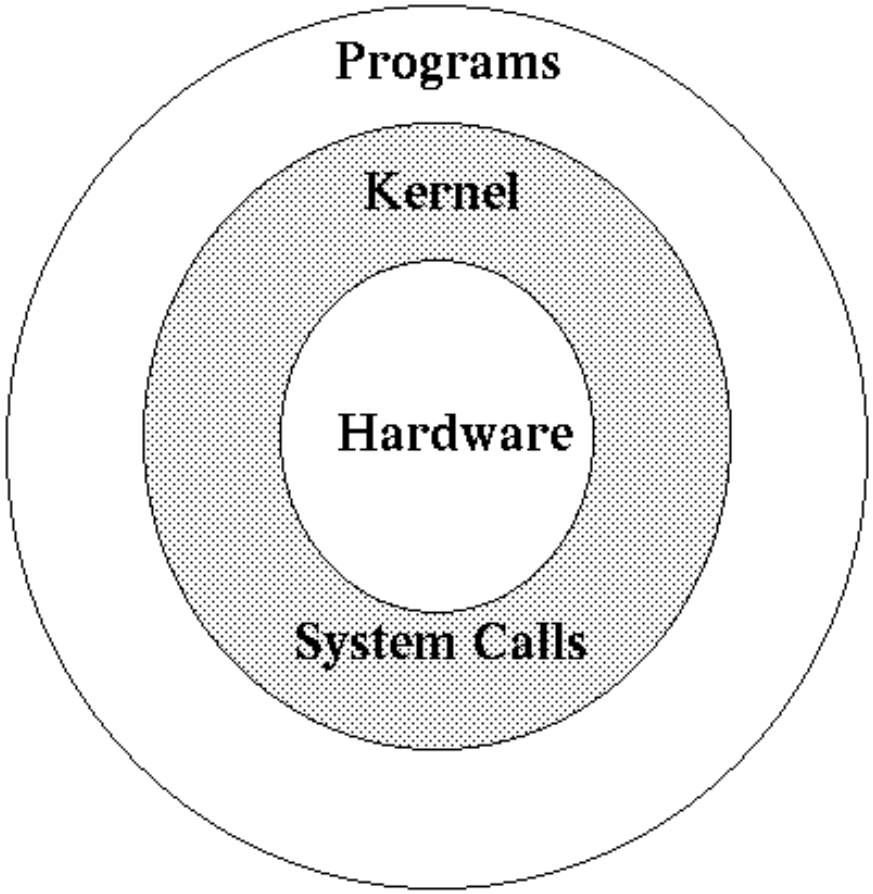
Linux Distributions

- 600+ Linux Distributions
  - RHEL (Commercial Support)
  - Fedora (Free, Majorly used for Desktop)
  - Ubuntu (Free, Majorly used for Desktop, From South Africa)
  - Slackware (One of the oldest, simple and stable)
  - CentOS (free RHEL, From England)
  - SuSe (Free and Commercial, From Germany)
  - Knoppix (first LiveCD distribution)

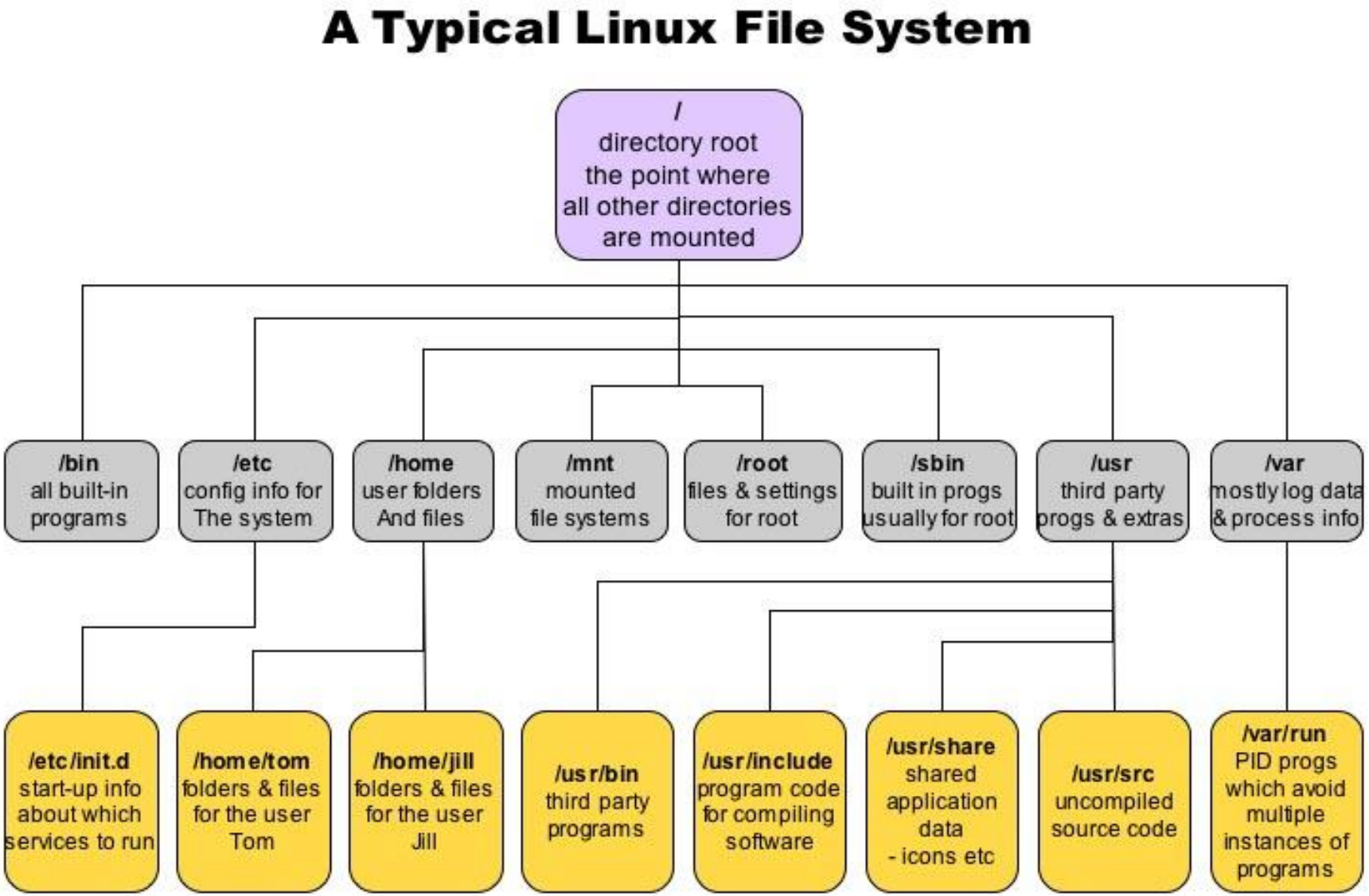
Which Linux Distribution…?

- Depends on user requirements
  1. Slackware
  2. Ubuntu
  3. Fedora
  4. RHEL
  5. CentOS

UNIX/Linux Structure



Linux File System



Details of File System

/	→	root directory
/boot	→	files for booting system
/etc	→	configuration files
/bin	→	important system binaries
/sbin	→	contains system admin programs(super user)
/usr	→	user applications
/lib	→	dynamic libraries
/home	→	user home directories
/root	→	super user home dir
/var	→	contains variable data constantly generated when system is running
/dev	→	device files

Files, Directories and Inodes

- **File:** A file represents a sequence of bytes.
  - Each file will have a name
  - Special characters are allowed but need to be used carefully
- **Directory:** A directory represents a list of files.
  - **A directory is also a file** which contains the list of files containing in it.
  - Every directory and file will be listed in its parent directory
- **Inode:** An inode (Index Node) contains information about a file (metadata) – File permissions, UID, GID, Size, Time Stamp etc.
  - The information about all the files will be maintained in a table called **"Inode Table"**

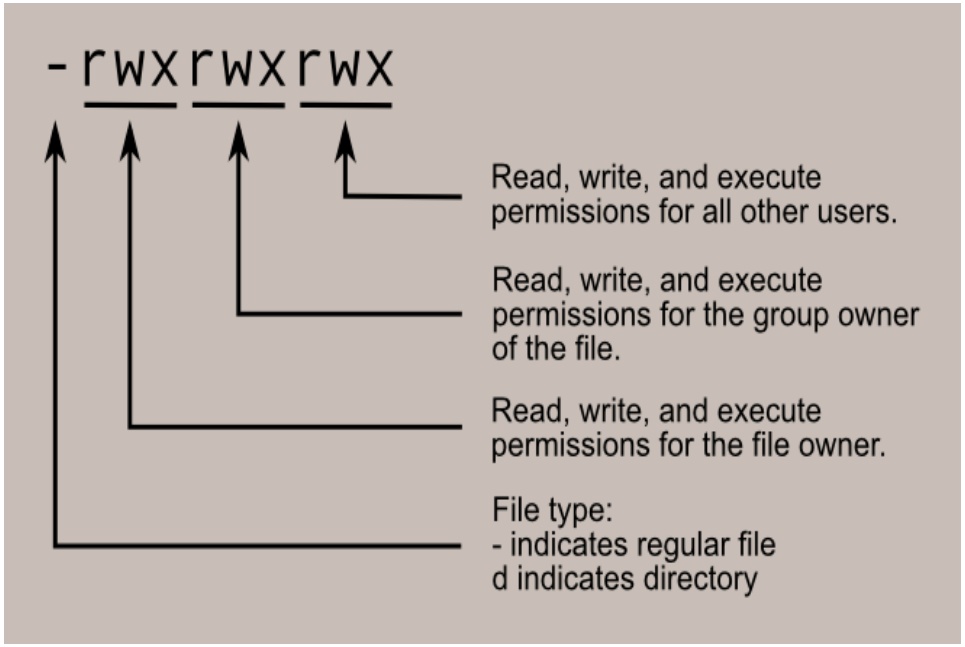
Users and Groups

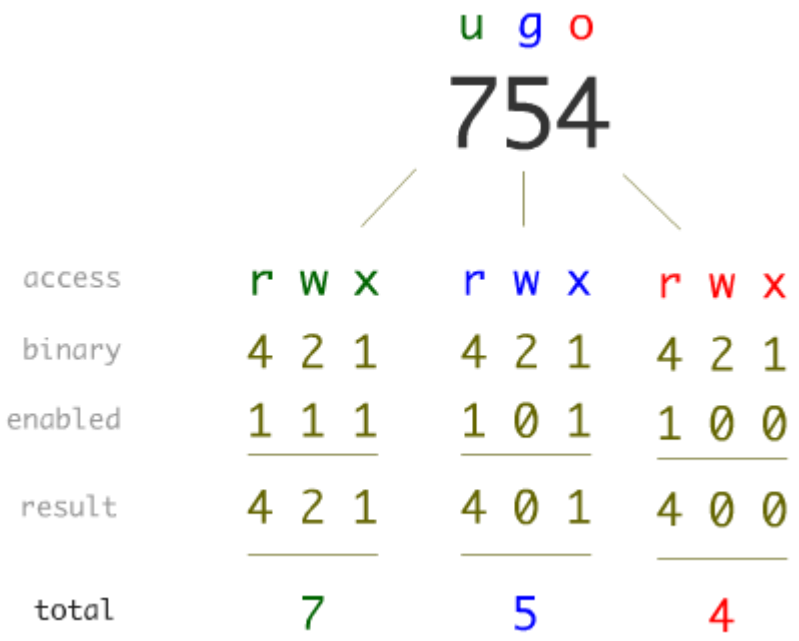
- **Users:** Users can be either people or accounts which exist to use specific applications of Linux.
  - Each user will be given a unique user ID (UID).
  - A **root user** will also be present and he has all the administrave privileges.
- **Groups:** Users can be tied together into groups for a common purpose.
  - Example: climate, ssdh, acts
  - Each group is associated with a group ID (GID).

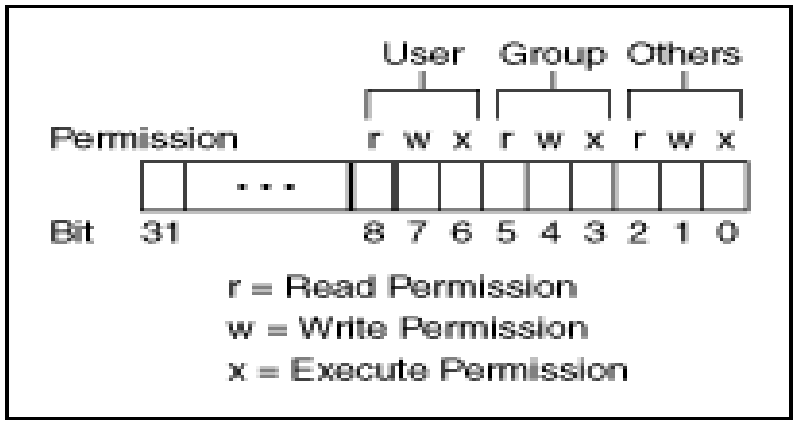
Access Permissions

- **File Permissions:** There are 3 permissions for any file → r, w, x.
  1. Read (r) - Indicates that a given category of user can read a file.
  2. Write (w) - Indicates that a given category of user can write to a file.
  3. Execute (x)- Indicates that a given category of user can execute the file.
- **Directory permissions:**
  1. Read (r) - The direcotry can be read.
  2. Write (w) - The directory can be updated, renamed or deleted.
  3. Execute (x)- Operations can be perfomred on the files of the directories. This bit is also called as search bit, it indicates whether you are permitted to search files under that directory
- **Categories of users:** All of these three permissions are assigned to three categories of users – User (U), Group(G), Others(O)

Access Permissions...





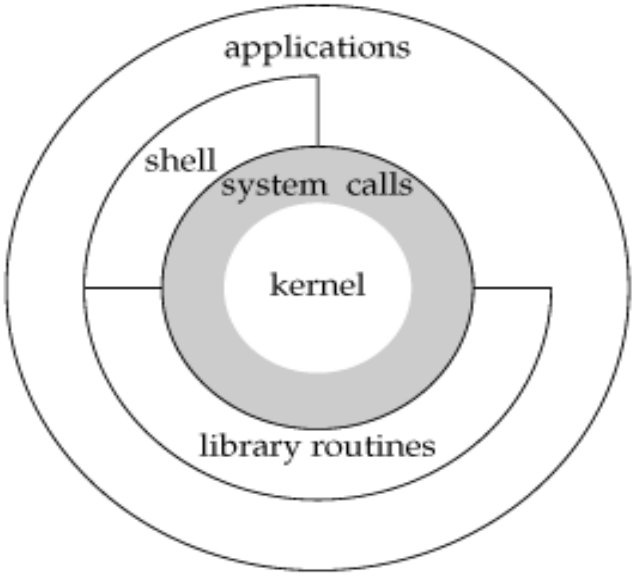


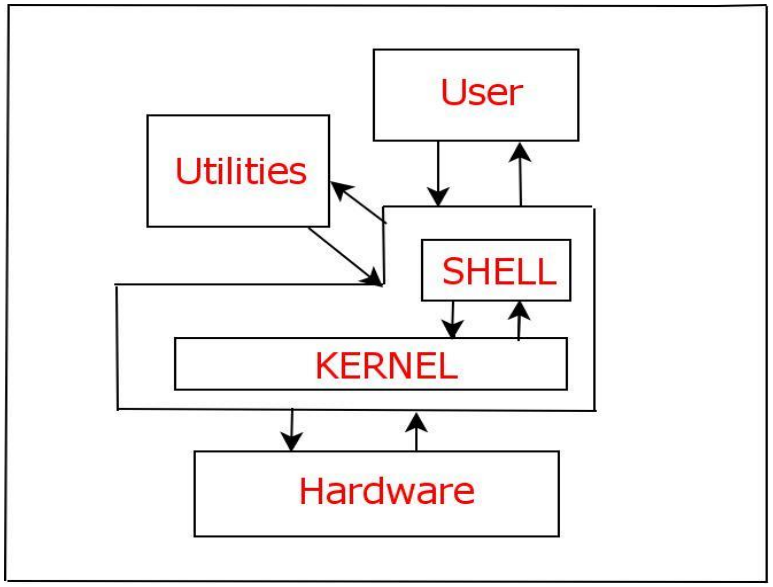
File Types

File Type	Symbol	Created by	Removed by
Regular file	-	Editors, cp, etc..	rm
Directory	d	mkdir	rmdir, rm -f
Character device file	c	mknod	rm
Block device file	b	mknod	rm
UNIX domain socket	s	socket(2)	rm
Named pipe	p	mknod	rm
Symbolic link	l	ln -s	rm

File Types Table

What is Linux Shell?





- The **kernel** sits on top of the hardware and is the core of the OS; **it receives tasks from the shell and performs them**
- The **shell** is the command line interface through which the **user interacts with the OS**. Most commonly used shell is “bash”
- Linux has a kernel and one or more shells – bash, csh, ksh etc.
- When you login to the system the default shell will be given to the user. Bash is the default shell for Linux (We can change the default shell)
- GUI of Linux is in fact an application program works on the shell.

Shell commands

- Shell Command - A **command is a program** which interacts with the kernel to provide the environment and perform the functions called by the user.
- A command can be: **a built-in shell command**; an executable shell file, known as a **shell script**; or a source compiled, object code file.

- Examples: ls, mkdir, pwd, cd
- Command Structure:

  $ command <options> <arguments>

- Multiple commands can be executed one by one using a single line by separating them with semicolon.

  $ ls; mkdir my_dir; ls

Shell Programming

- Shell Script - A series of commands can be written into a text file and execute that file. This is called as shell script.
- The first line of a shell script must be "#!/bin/Shell-Name"
- In order to execute a shell script we need to give execute permission (x) to that file using "chmod" comand.

  chmod +x users.sh

- Example:

```
#!/bin/bash
date
who
ps -ax
```

How to explore?

Man pages

- Manpage
  - $ man ls
  - $ man 2 mkdir
  - $ man man

How to explore…?

Man page sections

1. User-level commands and apps - /bin/mkdir
2. System calls - int mkdir(const char *, …);
3. Library calls - int printf(const char *, …);
4. Device drivers and network protocols - /dev/tty
5. Standard file formats - /etc/hosts
6. Games and demos - /usr/games/fortune
7. Misc. files and docs - man 7 locale
8. System admin. commands - /sbin/reboot

Shell commands

1. File management
2. System information
3. Process and Job management
4. Network
5. Searching
6. System start and stop
7. Compression and decompression
8. Miscellaneous

Shell commands…

1. File management

ls	- directory listing (ex: ls a* , ls *.c)
ls -al	- formatted listing with hidden files
cd dir	- change directory to dir
cd	- change to home
pwd	- show current directory
mkdir dir	- create a directory dir
rm file	- delete file
rm -r dir	- delete directory dir
rm -f file	- force remove file
rm -rf dir	- force remove directory dir *
cp file1 file2	- copy file1 to file2
cp -r dir1 dir2	- copy dir1 to dir2; create dir2 if it doesn't exist

Shell commands…

1. File management …

mv file1 file2	- rename or move file1 to file2. If file2 is an existing dir, moves file1 into directory file2
touch file	- create or update file
cat > file	- places standard input into file
more file	- output the contents of file
head file	- output the first 10 lines of file
tail file	- output the last 10 lines of file
tail -f file	- output the contents of file as it grows, starting with the last 10 lines

ls > dir-file-list.txt - redirecting the command output into file

Shell commands…

1. File management …

The single most useful command…….

$ man man

Who am I?

$ whoami

Where am I?

$ cd

$ pwd

Shell commands…

1. File management …

What is here?

$ ls

That's all? Or still more files??

$ ls -a

A file or a directory?

$ ls -F

Want to see the details of all the files/dirs?

$ ls -l

Shell commands…

1. File management …

Want to go to a different directory?

$ cd path_of_dest_dir

Want to go to your parent dir?

$ cd ..

Want to jump into your home?

$ cd

Want to go to root of entire machine?

$ cd /

Shell commands…

1. File management …

Want to create a new directory?

$ mkdir dir_name

Want to create a directory tree (In the current directory)?

$ mkdir -**p** eDESD/c/day1/linux

Want to copy the contents of one file to another file?

$ cp file1 file2

Want to copy a directory to another directory?

$ cp –r dir1 dir2

Shell commands…

1. File management …

chmod *octal file* – change the permissions of *file* to *octal*, which can be found separately for user, group, and world by adding:

- 4 – read (r)
- 2 – write (w)
- 1 – execute (x)

Examples:

chmod 777 – read, write, execute for all

chmod 755 – rwx for owner, rx for group and world

For more options, see **man chmod**.

Shell commands…

2. System Information

date	– Show the current date and time
cal	– Show this month's calendar
uptime	– Show current uptime
w	– Display who is online
whoami	– Who you are logged in as
finger user	– Display information about user
uname -a	– Show kernel information
cat /proc/cpuinfo	– CPU information
cat /proc/meminfo	– Memory information
df	– Show disk usage
du	– Show directory space usage
free	– Show memory and swap usage

Shell commands…

3. Process and Job management

ps – Display your currently active processes

top – Display all running processes

kill *pid* – Kill process id *pid*

killall *proc* – Kill all processes named *proc* (use with extreme caution)

bg – Lists stopped or background jobs; resume a stopped job in the background

fg – Brings the most recent job to foreground

Shell commands…

4. Network commands

ifconfig – List IP addresses for all devices on the local machine

ping *host* – Ping ***host*** and output results

wget *file* – Download ***file***

wget -c *file* – Continue a stopped download

Shell commands…

5. Searching

grep *pattern files* – Search for *pattern* in *files*

grep -r *pattern dir* – Search recursively for *pattern* in *dir*

command* | grep *pattern – Search for *pattern* in the output of *command*

locate *file* – Find all instances of *file*

find / -name *filename* – Starting with the root directory, look for the file called *filename*

locate *filename* – Find a file called *filename*

which *filename* – Show the subdirectory containing the executable file called *filename*

grep *TextStringToFind /dir* – Starting with the directory called ***dir***, look for and list all files containing *TextStringToFind*

Shell commands…

6. System stop and start

shutdown -h now – Shutdown the system now and do not reboot

shutdown -r 5 – Shutdown the system in 5 minutes and reboot

shutdown -r now – Shutdown the system now and reboot

reboot – Stop all processes and then reboot - same as above

Shell commands…

7. Compression and uncompression

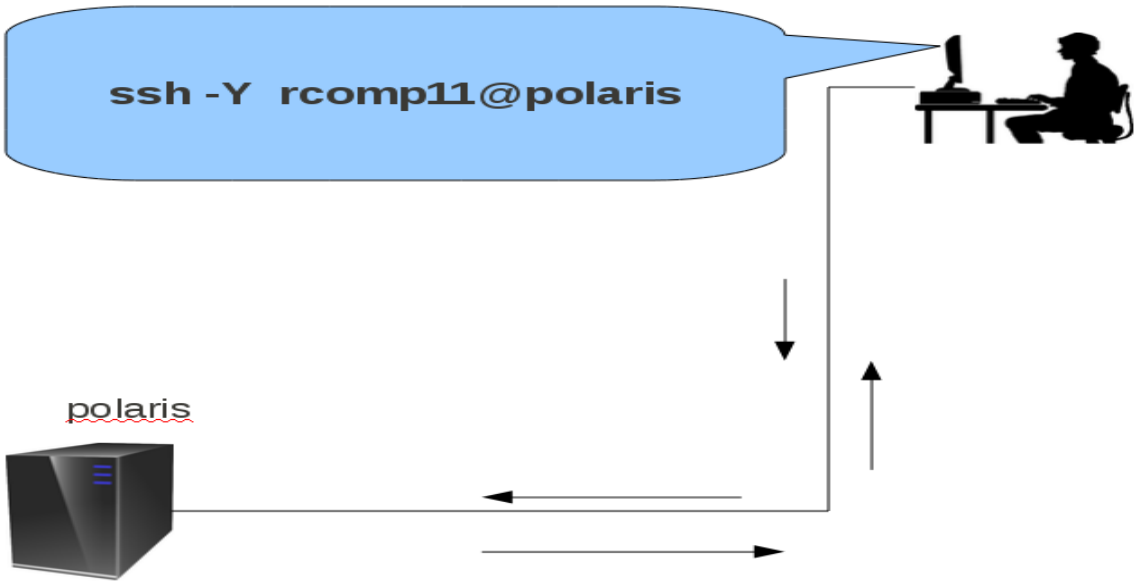
tar cf *file.tar files*	– Create a tar named ***file.tar*** containing ***files***
tar xf file.tar	– Extract the files from ***file.tar***
tar czf *file.tar.gz files*	– Create a tar with Gzip compression
tar xzf *file.tar.gz*	– Extract a tar using Gzip
tar cjf file.tar.bz2	– Create a tar with Bzip2 compression
tar xjf *file.tar.bz2*	– Extract a tar using Bzip2
gzip *file*	– Compresses ***file*** and renames it to ***file.gz***
gzip -d *file.gz*	– Decompresses ***file.gz*** back to ***file***

Shell commands…

8. Miscellaneous

adduser *accountname*	– Create a new user call ***accountname***
passwd *accountname*	– Give ***accountname*** a new password
su	– Log in as superuser from current login
exit	– Stop being superuser and revert to normal user

How to connect to Linux system?



ssh - is a protocol and a program
-Y - option that requests ssh "forward X traffic"
Rcomp11 - name of the user
Polaris - name of the remote computer

Quiz - 1

1. List the hidden files in the current directory.
2. Search for a string “DESD” in a file “desd.c”.
3. Display disk usage statistics in GBs.
4. Which command is used to change the file permissions of a file?

Vi editor

- The VI editor is the most popular and classic text editor in the Linux family. Below, are some reasons which make it a widely used editor –

  1) It is available in almost all Linux Distributions

  2) It works the same across different platforms and Distributions

vi Command mode:

- The vi editor opens in this mode, and it only **understands commands**
- In this mode, you can, **move the cursor and cut, copy, paste the text**
- This mode also saves the changes you have made to the file
- **Commands are case sensitive.** You should use the right letter case

Reference: https://www.guru99.com/the-vi-editor.html)

Vi editor

❖ **vi Editor Insert mode:**

- This mode is for inserting text in the file.
- You can switch to the Insert mode from the command mode **by pressing 'i' on the keyboard**
- Once you are in Insert mode, any key would be taken as an input for the file on which you are currently working.
- To return to the command mode and save the changes you have made you need to press the Esc key

How to use vi editor

To launch the VI Editor -Open the Terminal (CLI) and type
vi <filename_NEW> or <filename_EXISTING>
And if you specify an existing file, then the editor would open it for you to edit. Else, you can create a new file.

Reference: https://www.guru99.com/the-vi-editor.html)

Vi editing basic commands

- i - Insert at cursor (goes into insert mode)
- a - Write after cursor (goes into insert mode)
- A - Write at the end of line (goes into insert mode)
- ESC - Terminate insert mode
- u - Undo last change
- U - Undo all changes to the entire line
- dd - Delete line
- 3dd - Delete 3 lines.
- cw - Change word
- x - Delete character at the cursor
- r - Replace character

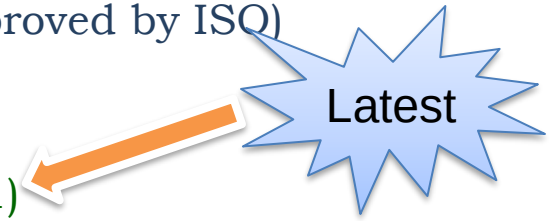
Saving and Closing the file

- Shift+zz - Save the file and quit
- :w - Save the file but keep it open
- :q - Quit without saving
- :wq - Save the file and quit

Reference: https://www.guru99.com/the-vi-editor.html)

Overview of C

- C is a general-purpose, procedural, computer programming language developed in between **1969** and **1973** by **Dennis M. Ritchie** at the Bell Telephone Laboratories to develop the UNIX operating system
- UNIX OS, C compiler have been written in C
- C standards
  - 1989 - ANSI C (C89)
  - 1990 - C90 (Same as C89, but approved by ISO)
  - 1999 - C99
  - 2011 - **C11** (Approved in Dec-2011)



- New features in C11 – Type generic macros, Anonymous structures, Atomic operations, Bounds-Checked functions etc.

Reference: http://en.wikipedia.org/wiki/C_(programming_language)

Applications of “C”

- Operating Systems
- Language Compilers
- Assemblers
- Text Editors
- Device Drivers
- Modern Programs
- Databases
- Lang. Interpreters
- Utilities
- Libraries

Applications developed using C

- Google Chrome, Firefox browsers
- Unix/Linux
- MySQL
- MS Office
- Mozilla Thunderbird (Mail client)
- Winamp
- VC++ Compiler
- JVM
- And many more……

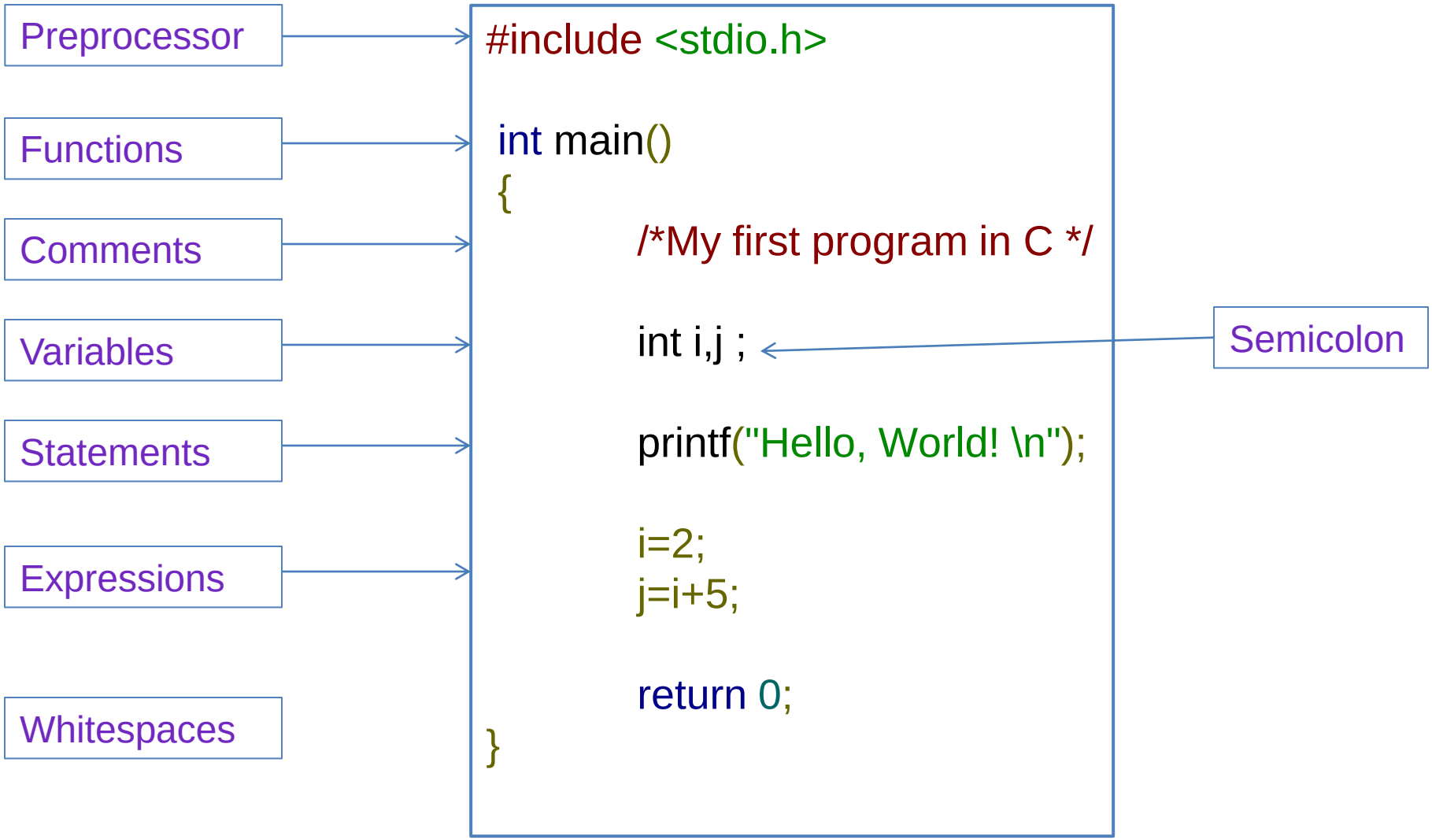
Most widely used & popular system programming Language

Start Working with C

Requirements:

- **Editor:** vi (or) vim (or) gedit
- **Compiler**: gcc or icc
- **OS:** Linux
- Run Time Environment
- Syntax of C

Components of a C Program



Compiling and running C programs

Source Files: hello.c
Executables: hello

```
gcc  hello.c   -o  hello
```

Execution:

```
./hello
```

Source Files: main.c hello.c
Executables: main

```
gcc  main.c hello.c   -o  main
```

Execution:

```
./main
```

THANK YOU