Introduction to Linux

Dr. Asif Uddin Khan

What OS does?

- It loads itself into the memory-Booting.
- It loads the user program in the memory.
- It loads the data required by the program in the memory.
- Interprets the program instruction one at a time.
- Sends instruction to the output device to display the result on the screen.
- Uses the resources like memory, input and output devices properly and efficiently.

Major OS Functions

- User Interface
- Job/Task Management
- Data Management
- Device Management
- Security

Types of OS

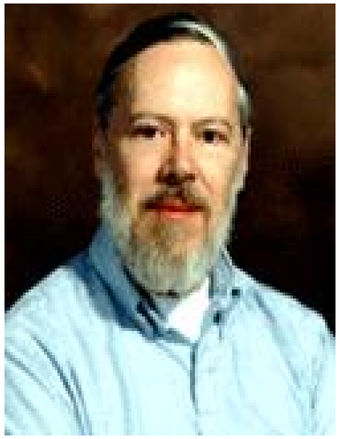
- Single User OS
- Multi User OS
- Multi Tasking OS
- Multi Processing
- Real Time OS

Linux OS

- Linux is a free Unix-like operating system Developed under the GNU General Public License
- It is a multi-user, multitasking OS with support for Networking
- The source code for Linux is freely available to everyone.

History of Development!

- The development of Linux started from UNIX.
- UNIX is a trade mark of AT & T, originally developed by Ken Thompson, Dennis Ritchie and others back in 1970s running on mainframes and microcomputers.
- It is a multitasking and multi user OS.
- Unix is mostly used in servers and web servers.
- It is written with 'C' language
- It is a foundation for many OS like Linux, Novell etc.


Dennis Ritchie, 1941

Ken Thompson, 1943

History continues

- Linux OS uses Unix as the base and gives further more facilities and applications.
- It was first released by its inventor Linus Torvalds in 1991.
- In Linux GUI is made having Unix as its core.
- Linux is the ultimate Unix.

Linux Distributions

- Several Companies and Organizations began gathering and packaging Linux software together in to useable forms called distributions
- Now there are almost more than 350 distributions available around the world!
- The most popular distributions are
  – Red Hat, SuSe, Debian, Slackware, Mandrake etc.

Features of LINUX

- Multi-user
  - Multiple users can logged into and work on the system at the same time.
  - Each user will have their own private area in the system where they can customize their environments for use.
- Multitasking
  - Can have many programs running at the same time for the user
  - Also Linux runs many of the system processes in the background called Daemon processes, which remain ready for ever to serve the request of a user.
- Graphical User Interface.
- Hardware Support
  - Linux supports all most all well known hardware.
- Networking Support

Linux Commands

What is a Command?

- A command is an instruction telling a computer to do something.

Linux Commands

A command prompt looks like

```
[cs0203@linuxserver cplab23]$
```

- cs0203: user name
- linuxserver: machine name
- cplab23: current working directory

Guidelines

- All commands must always be entered in small case letters
- Between command name and the options there must be a space
- The options are usually preceded by a ‘-’ sign
- Two or more options may be combined like
- ls –a -l as ls –al
- To repeat a command up-down arrow keys may be used

Basic Linux Commands

- **date**
  - prints current date and time
  - Syntax:

```
[cs0203@linuxserver cpp]$ date
```

  - O/P: Tue Nov 22 15:24:12 IST 2005
- **cal**
  - Prints calander of current year
  - Syntax:

```
[cs0203@linuxserver cpp]$cal 11 2005
```

  - O/P:

```
November 2005
Su  Mo  Tu  We  Th  Fr  Sa
         1   2   3   4   5
 6   7   8   9  10  11  12
13  14  15  16  17  18  19
20  21  22  23  24  25  26
27  28  29  30
```

Basic Linux Commands

- **who (who am i)**
  - Displays the users logged in to the system
  - Syntax

```
[cs0203@linuxserver cpp]$who
```

  - O/P:

```
ee05183  pts/278     Nov 22 15:23 (192.168.5.8)
et05428  pts/286     Nov 22 15:29 (192.168.5.9)
ee05410  pts/285     Nov 22 15:29 (192.168.5.10)
```

- **pwd**
  - Displays present working directory
  - Syntax:

```
[cs0203@linuxserver cpp]$pwd
```

  - O/P: /home/cs0230/cpp

Basic Linux Commands

- **ls**
  - Lists files and folders of current directory
  - O/P:

```
squid.conf    fontforge-20040529-1.i386.rpm     squid.new
    squid.conf  golden collection mukesh pics
    squidGuard.log      indlinux tcp
```

- **ls -l**
  - Long listing of files and folders of current directory
  - O/P:

```
-rw-r--r--   1 cs01 cs01     2474 Oct  1 17:25 maze.cpp
drwxrwxr-x   4 cs01 cs01     4096 Oct 25 11:19 linux
```

- **ls –al**
  - List all files in the current working directory in long listing format showing permissions, ownership, size, and time and date stamp

Getting Help

- **man <command>**
  - Displays the system manual
  - Command man displays information on man
  - Syntax :

```
[cs0203@linuxserver cpp]$man ls
```

- **info <command>**
  - Displays information in a different format
  - Command info displays information on man
  - Syntax:

```
[cs0203@linuxserver cpp]$info date
```

What is a File?

Any named collection of information stored on a disk. Application programs and documents are examples of files. You make a file when you create information (such as text or graphics) using a program, give the material a name and save it on a disk.

What is a Directory?

It is a special kind of file that contains others files and directories. Directories can be nested to any depth. Some software may refer to directories and subdirectories by other names, such as, folders, lockers, file drawer, etc.

Creating a file

- **touch <file name>**
  - Creates an empty file
  - Syntax:

    [cs0203@linuxserver cpp]$touch test.txt

- **cat**
  - Used to create and display a file.
  - cat <file name> : displays file contents

    Syntax:**[cs0203@linuxserver cpp]$cat firstfile.txt**
  - cat **>** <file name>: creates a file and save with **ctrl+d**

    Syntax:[cs0203@linuxserver cpp]$cat > firstfile.txt
  - cat file1 file2 **>** file3 : concatenates contents of files file1 & file2 & store it in file3(if it is not exist then newly created otherwise content of file3 is overridden)

    Syntax:[cs0203@linuxserver cpp]$cat file1 file2 > file3
  - Cat file1 file2 >> file3 :concatenates content of files ,if file3 exist &contents some information then new content is appended.

    Syntax:**[cs0203@cat file1 file2 >>file3**

File related commands

- **mkdir <dir name>**
  - To create directory(s)
  - Syntax:

```
[cs0203@linuxserver cpp]$mkdir kiit
```

- **cd <dir name>**
  - To change directory
  - Syntax:

```
[cs0203@linuxserver cpp]$cd kiit
```

- **file <file name>**
  - Display file type
  - Syntax:

```
[cs0203@linuxserver cpp]$file firstfile
```

What is Path?

Path is the specification of a file or directory in a hierarchical file system using pathname separators ("/" in Unix)between directories. The pathname tells the shell where in the directory tree to find a file. Files may be referred to by **absolute pathname** (also called full or complete pathname) or **relative pathname**.

Absolute path is a path relative to the root directory, the topmost node of a hierarchical file system. Its first character must be a pathname separator, e.g., the Unix forward slash . absolute pathname is the full specification of a path beginning with the root directory and including all directory levels the system passes through to locate the file.

relative pathname is the location of the file or directory relative to the directory in which the user is currently located (the current working directory).

File commands contd.

- **cp <srcfile> <destfile>**
  - Copies srcfile to destfile
  - If destination file is present then overwrites
  - Else creates a new file with name destfile
  - Syntax:

```
[cs0203@linuxserver cpp]$cp test.c  test1.c
```

- **clear**
  - **Clears the screen**
  - Syntax:

```
[cs0203@linuxserver cpp]$clear
```

- **shutdown**
  - **Shuts the system down.**
  - Syntax:

```
[cs0203@linuxserver cpp]$shutdown
```

File commands contd.

- **mv**
  - moves/renames a file or directory
  - mv <oldfile> <newfile> : renaming
  - mv <file> <path/file>: moves to path directory
  - mv <file1> <path/file2>: moves with renaming
  - Syntax:

```
[cs0203@linuxserver cpp]$mv test.txt  test1.c
```

- **rm <file>**
  - Removes/deletes a file
  - Syntax:

```
[cs0203@linuxserver cpp]$rm test.c
```

- **rmdir <dir>**
  - Removes/deletes an empty directory
  - Syntax:

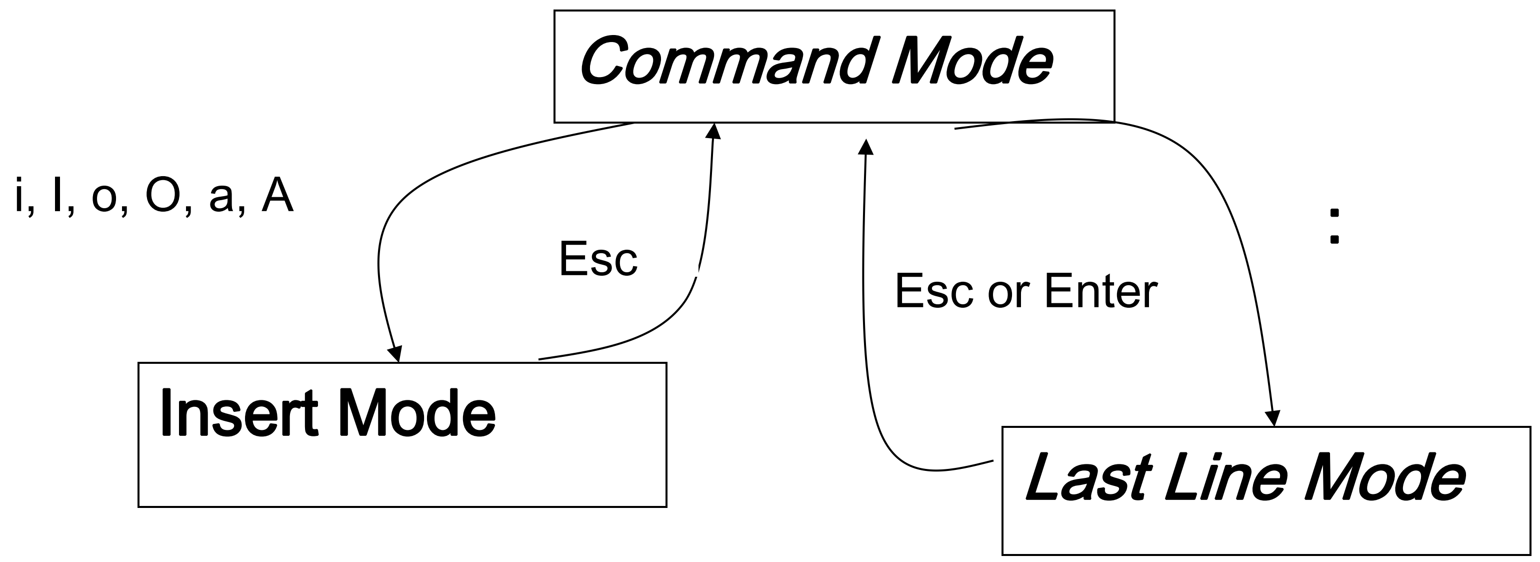
```
[cs0203@linuxserver cpp]$rmdir  cpp
```

Some VI/VIM Commands

INTRODUCTION TO VI EDITOR

- vi is a screen-oriented text editor written by Bill Joy in 1976 .
- The name vi is derived from the shortest unambiguous abbreviation for the command visual in ex.
- vi editor works in 3 different modes

  >> Command Mode
  >> Insert Mode
  >> Last Line Mode


Command Mode
i, I, o, O, a, A
Esc
Esc or Enter
:
Insert Mode
Last Line Mode

Command to invoke Vi editor

```
[cs0203@linuxserver cpp]$vi <filename>
```

e.g

```
[cs0203@linuxserver cpp]$vi firstcprog.c
```

- If the file firstcprog.c exists then this command open the file otherwise it will be created.

- when vi editor starts it will be in command mode. To type we must have to go to Insert mode.

Operation on Vi Editor

✓Delete Operation

- First move the cursor so that it covers the first character of the group you want to delete, then type the desired command as below
- **x** Delete only the current character
- **D** Delete to the end of the line
- **db** Delete from the current character to the beginning of the current word
- **de** Delete from the current character to the end of the current word
- **dd** Delete/cut the current line
- **dw** Delete/cut from the current character to the beginning of the next word
- **[n]dd** delete/cut n lines from the current line.

Operation on Vi Editor

✓Copy & Paste operation

- **yy**

  Yank(copy) the current line.
- yw

  Yank (copy) to the start of the next word.
- **[n]yy**

  Yank (copy n lines from the current line.

  Example: 3yy
- p

  Paste the specified buffer after the current cursor position or line.
- P

  Paste the specified buffer before the current cursor position or line.

❑u   undo last change in a file

❑U   undo **all** changes to the current line

Inserting New Text (Change mode from Command to Insert)

A

Append at the end of the current line.

I

Insert from the beginning of a line.

O

(letter oh) Enter *insert* mode in a new line above the current cursor position.

a

Enter *insert* mode, the characters typed in will be inserted after the current cursor position. A count inserts all the text that had been inserted that many times.

i

Enter *insert* mode, the characters typed in will be inserted before the current cursor position. A count inserts all the text that had been inserted that many times.

o

Enter *insert* mode in a new line below the current cursor position.

Saving and Quitting

ZZ	(in Command Mode) Exit the editor, saving if any changes were made.
:w	(in Last Line Mode) Save the current document
:q	(in Last Line Mode) Exit the editor if no modification is made.
:wq	(in Last Line Mode) Exit the editor, saving if any changes were made.
:q!	(in Last Line Mode) Exit the editor, without saving if any changes were made.
:wq!	(in Last Line Mode) Exit the editor, overrides the files saving if any changes were made.

Miscellaneous

- echo Prints the typed word on the terminal.
- tail Displays the last ten lines of a file
- tar Compresses a directory
- tar -xvf to extract the compressed file
- sudo super user
- install used to install a package using **apt-get**, **apt**

Example: sudo apt-get install vim

Sample Answers(SA):

C Program to Print "Hello World"

```
#include<stdio.h>
int main()
{
printf("Hello World\n");
return 0;
}
```

O/P:

```
Hello World
```

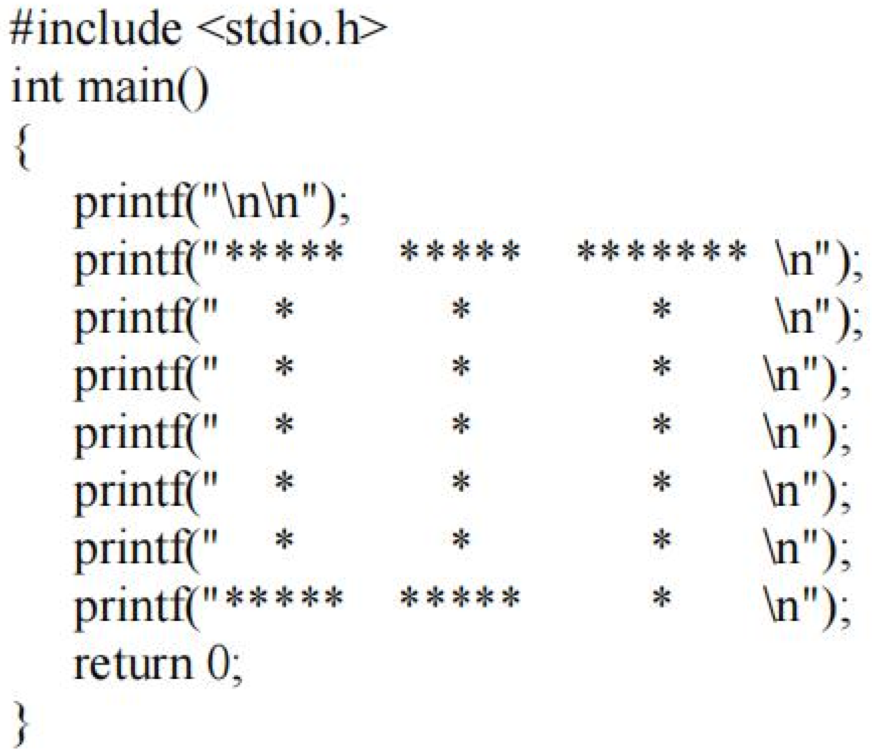
Sample Answers(SA):
C Program to Print Welcome Message

```
#include<stdio.h>
int main()
{
printf("\n Welcome to C Programming Laboratory\n");
return 0;
}
```

O/P:

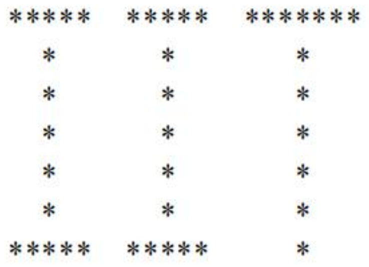
```
Welcome to C Programming Laboratory
```

WAP to display “IIT” using the character ‘*’



```
#include <stdio.h>
int main()
{
    printf("\n\n");
    printf("*****    *****    *******  \n");
    printf("  *        *         *      \n");
    printf("  *        *         *     \n");
    printf("  *        *         *     \n");
    printf("  *        *         *     \n");
    printf("  *        *         *     \n");
    printf("*****    *****       *     \n");
    return 0;
}
```

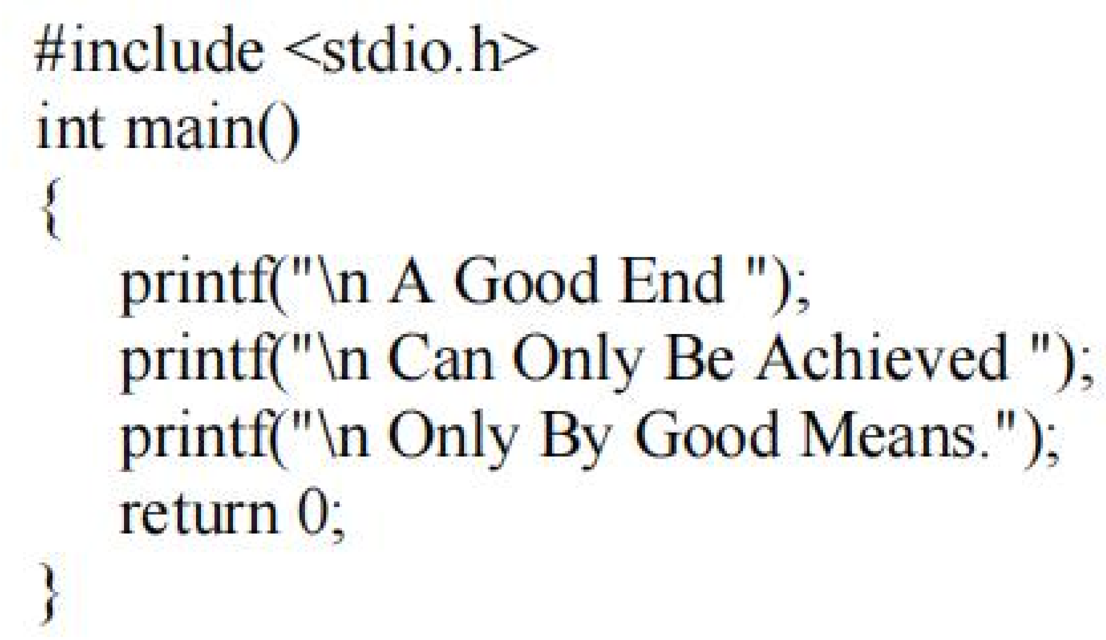
O/P:



```
*****   *****   *******
  *       *        *
  *       *        *
  *       *        *
  *       *        *
  *       *        *
*****   *****      *
```

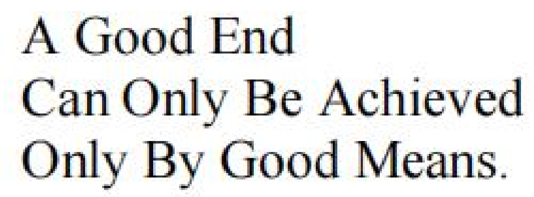
- WAP to display the following message by using multiple printf statement.
- A Good End
- Can Only Be Achieved
- Only By Good Means.

Program Code:



```
#include <stdio.h>
int main()
{
    printf("\n A Good End ");
    printf("\n Can Only Be Achieved ");
    printf("\n Only By Good Means.");
    return 0;
}
```

Output:



```
A Good End
Can Only Be Achieved
Only By Good Means.
```

- WAP to display the following message by using single printf statement.
- A Good End
- Can Only Be Achieved
- Only By Good Means.

Program Code:

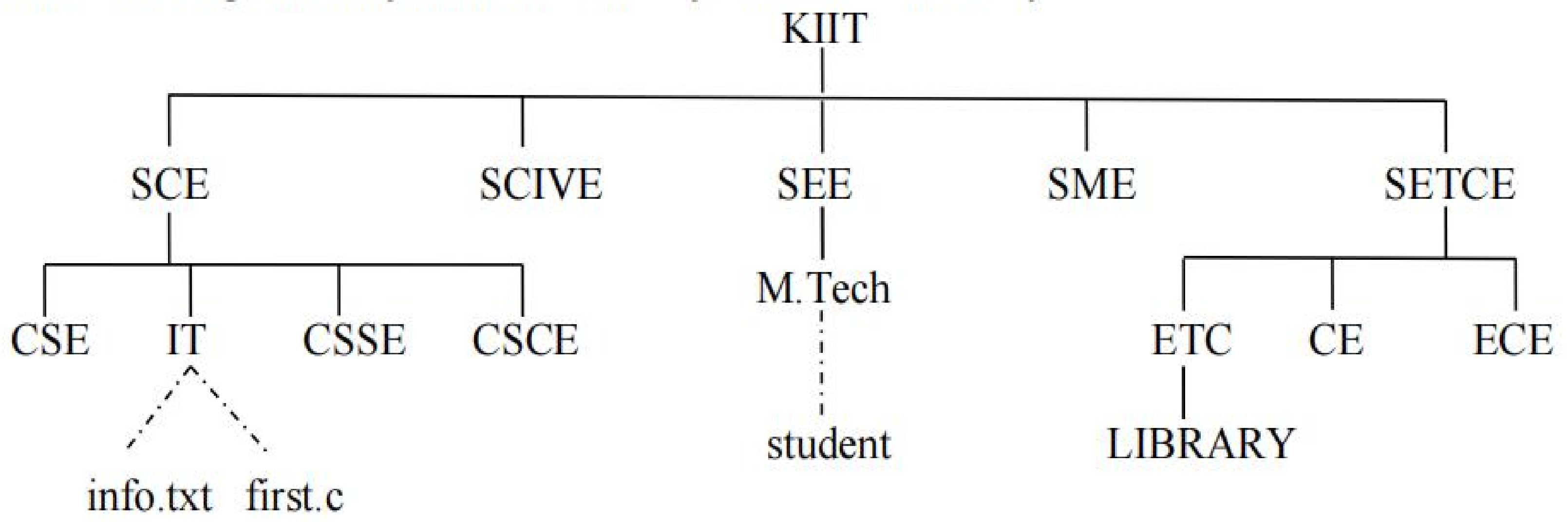
```
#include <stdio.h>
int main()
{
    printf("\n A Good End\nCan Only Be Achieved\nOnly By Good Means.");
    return 0;
}
```

Output:

```
A Good End
Can Only Be Achieved
Only By Good Means.
```

Lab Assignments(LA)

LA1.1 First create a sub-directory named as your roll number under your home directory. Then create the following directory structure under your rollno directory.



N.B. The names under solid lines are assumed as directories and dotted lines as file names.

Do the following operations

a) Create the file names under the directories as mentioned in the figure and write some relevant data into the files.

b) Rename the file info.txt as itstudentsdata.txt.

c) Copy the file first.c into the directory CE with the same name.

d) Copy the file first.c into the directory SME with a new name as hello.c.

e) Transfer the file student into the directory SCIVE and check whether transferred or not.

Lab Assignments(LA)

LA1.2 WAP to display “KIMS” using the character ‘#’.
LA1.3 WAP to display the following message by using multiple printf statement.

If The End Is Good,
Then It Is Good,
Whatever Be The Means.

LA1.4 WAP to display the message of LA 1.3 by using single printf statement.

Home Assignments(HA)

HA1.1 WAP to print your BIO-DATA (Name, Regd.no”, Branch, JEE Rank, Gender, Phone no., Address etc.) using printf statement.

Thank You

Any Question!