

**Birla Institute of Technology & Science, Pilani**  
**First Semester 2015-2016, Computer Programming [CS F111]**  
**Lab #1**

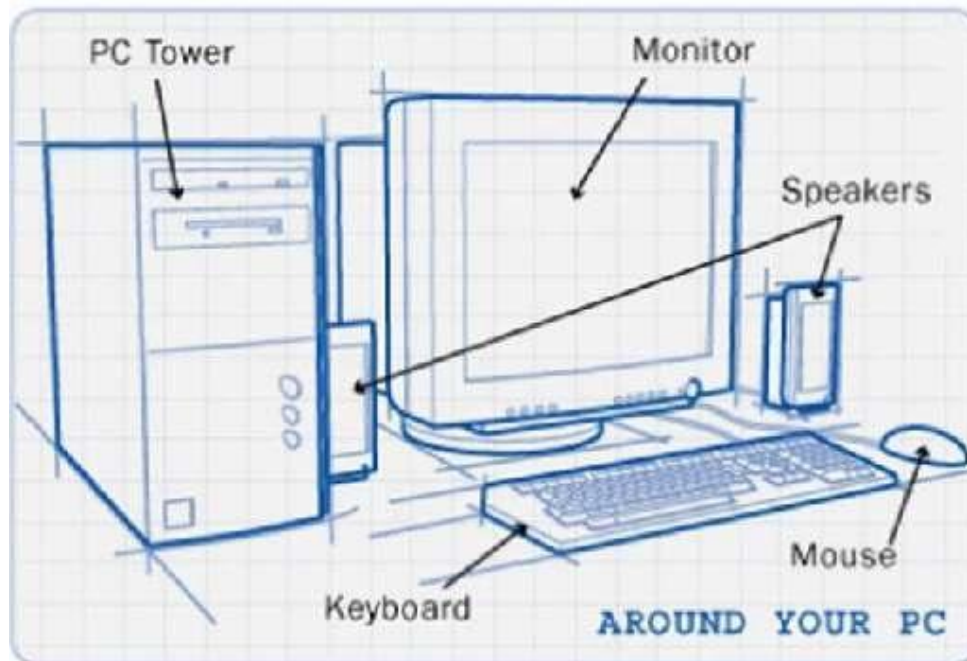
**Week #1**

**OBJECTIVES -**

- ✓ Introduction (Computing Machine, Operating System, UNIX)
- ✓ Getting started with UNIX
  - UNIX treats everything as Files!!!
  - Print Working Directory (pwd command)
  - List the contents of current directory (ls command)
  - Creating a Directory (mkdir command)
  - Changing to a different Directory (cd command)
  - File creation, display, and deletion.
  - Exercises
- ✓ Summary

**INTRODUCTION**

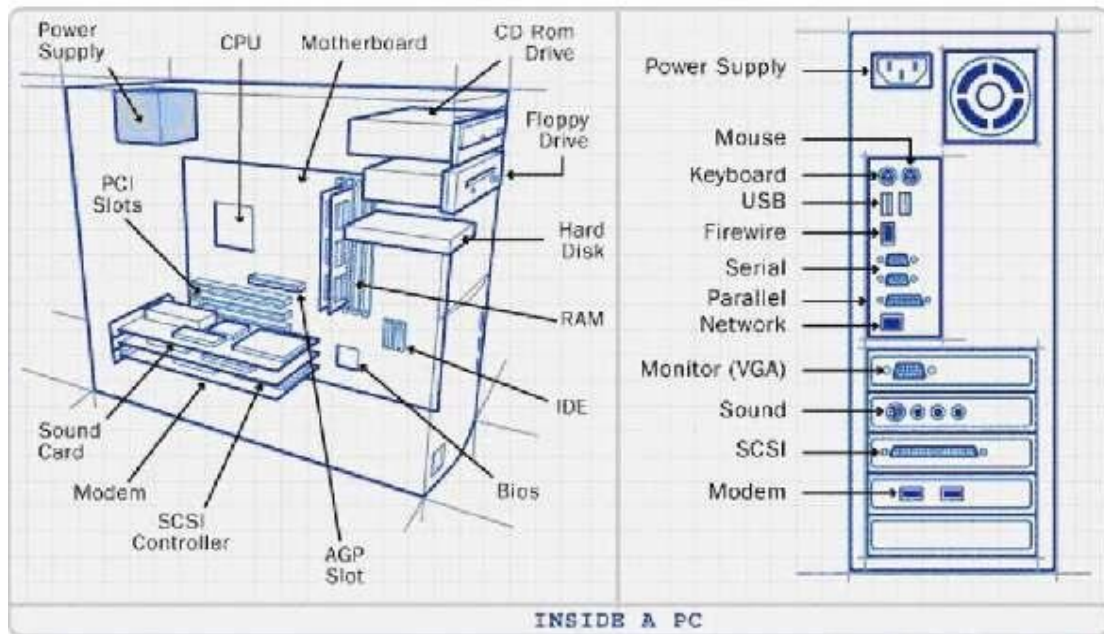
A *computer* is a machine, capable of performing a set of predefined operations. It is *programmable* i.e. it can be programmed to perform - arbitrarily long - sequences of operations chosen from predefined set. Its basic components - referred to as the hardware – provide for the predefined operations and its programmability is achieved using “software” i.e. instructions that tell the hardware to perform a sequence of these operations. A block diagram of a typical personal computer (PC) is shown below in Figure 1 with its typical hardware components marked.



**Figure 1**

**Exercise 1.1 (Trivial):** Spot the corresponding physical parts in the computer in front of you. [Some parts – such as the speaker may be missing.]

The PC tower is just a collection of various components connected together and ergonomically packaged into a box. The internals of a typical PC tower are shown in **Figure2.**



**Figure2**

### **Operating System**

The operating system is software that manages the hardware and helps you to interact with the computer. There are several Operating Systems available on the market today for PCs. Some of the most popular ones are:

1. Windows (Windows XP, Windows Vista, and Windows 7) produced by Microsoft.
2. Unix/Linux (AIX produced by IBM, Fedora produced by RedHat, HP-UX produced by HP etc.)
3. Mac OS (produced by Apple).

We will be using UNIX/LINUX for our lab work.

### **What is UNIX?**

UNIX is an operating system originally developed at AT&T Bell Laboratories. It became very popular among the scientific, engineering, and academic communities due to its *multi-user* and *multi-tasking* environment, flexibility and *portability*, *electronic mail* and *networking* capabilities, as well as the numerous programming, text processing and scientific utilities available [If you do not understand few words here, don't worry].

Unix does not refer to a single operating system but to a family of operating system i.e. there are many flavors (aka. variants, types, or implementations) of Unix available. Linux is a modern day variant developed originally by Linus Torvalds and several implementations of Linux are supported today by the open source community. Although based on a core set of Unix commands, different flavors have their own unique commands and features, and are designed to work with different types of hardware.

The following is some of the well-known Unix flavors,

1. Ubuntu (Linux variant)
2. Fedora (Linux variant)
3. Debian (Linux variant)
4. Free BSD (One of the old Unix variants still available)

We will be using **Fedora v8** in our lab.

## **GETTING STARTED WITH FEDORA**

Linux systems have two basic modes for a system to run in:

1. *Text console mode*, in which user interaction happens through commands typed using the keyboard and the display is text only.
2. *Graphical mode*, in which the display is rich in pictures, frames, windows, and interactive elements (such as menu items) and in which the user interaction happens through choices made using mouse clicks.

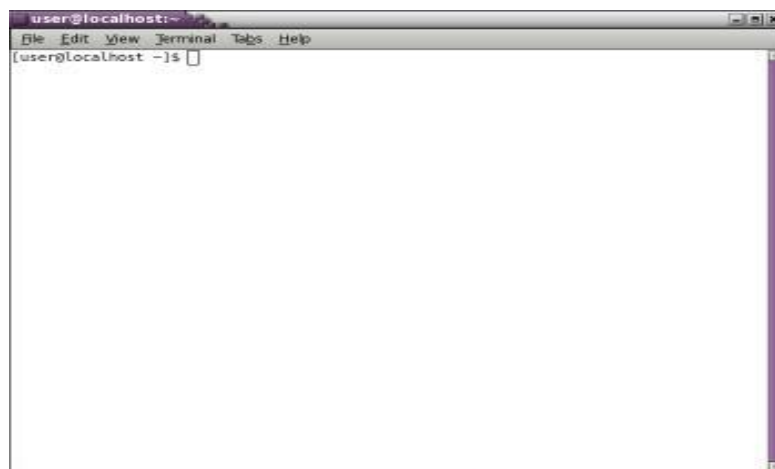
**We will be primarily using Text console mode.**

Your Desktop will look as in **Figure 5 (the Graphical mode)**



**Figure 5.**

Go to **Application** on top of the Desktop and click on **System Tools**. Then click on **Terminal**. You will see a window as in **Figure6**, on which you will learn /practice / program throughout this course by typing text commands.

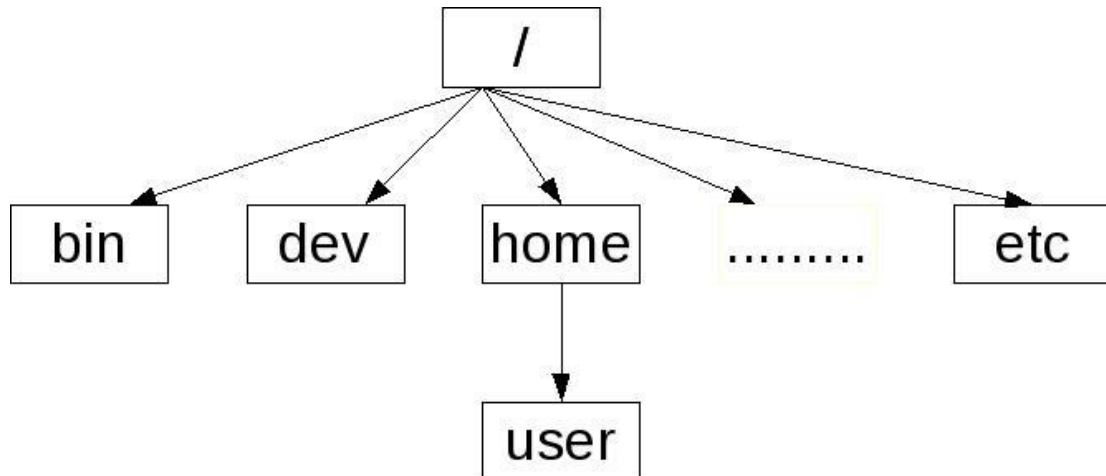


**Figure 6**

## UNIX treats everything as Files!!!

Everything in UNIX is either a *file* or a *process* (don't bother to understand the term process at this stage, not a big deal anyway). A file is a (logical) collection of data irrespective of where it is stored (in a hard disk, a CD ROM, a memory stick etc.). Contents of a file may be a sequence of characters (in which the file is referred to as a text file) or a sequence of bits (0s and 1s) in which case the file is referred to as a binary file.

All the files are grouped together in an (inverted) tree-like structure. The root (denoted by '/') is at the top. A collection of files is referred to as a directory in Unix as it forms an index for the collection of files. In Unix each directory is also stored as a file and therefore a directory may contain other directories (each of which is stored as a file).



Top-level Directory Structure in UNIX systems

Once you are logged in, you will **most likely be** in the directory named by user. But how do I verify this???

### Print Working Directory (Command Name: *pwd*)

Type *pwd* at the prompt

```
[user@localhost:~] $ pwd ↵  
/home/user
```

You get the path name of your *home directory*. A path is a sequence of edges (or lines) from the root to a specific file or directory. A pathname is a collection of names of the directories encountered in the path ending in the specific file or directory. For example in the above case: the path starts at /, then goes via home to user. The '/' character is also used to separate the directory names in the path.

### What is Home Directory???

When you first login, your current working directory is your home directory. Your home directory has the same name as user.

You would like to find out what is inside your home directory.

### List the contents of the current directory (Command Name: *ls*)

Do you see the following *prompt*?

```
[user@localhost:~] $
```

Type `ls` at the prompt

```
[user@localhost:~] $ ls ↵
```

You will get the prompt, it will list nothing. You will immediately see the prompt back

```
[user@localhost:~] $
```

Your home directory is empty!!! It's empty because you didn't create anything. But how can we make sure, that my home directory has nothing hidden? We can still find it out.

```
[user@localhost:~] $ ls -a ↵
```

```
.      ..      .bash_history  .bash_profile  .bashrc      .kde .lessht
```

Observe the result of our previous command `ls -a` : all the files listed as result start with a **DOT (.)**. These files are hidden files. (Do not fiddle with these files now!!!). Also observe the first two results. A Single dot **(.)** in UNIX means the current directory and two consecutive dots **(..)** means the parent of the current directory.

**[Note:** *There is a space between the command name `ls` and the option `-a`. Most commands in UNIX have options and optional arguments. Each option is usually preceded by a space and then a hyphen (-)* **End of Note]**

Type the following commands, and observe the outcomes.

```
[user@localhost:~] $ ls .
```

```
[user@localhost:~] $ ls ..
```

### Creating a Directory (Command Name: `mkdir`)

We will create a subdirectory within our home directory to hold the files we will be creating and using during the lab hours of CP. We name the new directory as 'exercises'.

The command is as follows

```
[user@localhost:~] $ mkdir exercises ↵
```

You will immediately get back the prompt, without any other messages, as

```
[user@localhost:~] $
```

But, how do we make sure that if the directory named 'exercises' is created or not??? We have `ls` command for it.

```
[user@localhost:~] $ ls ↵
```

```
exercises
```

***How do we go to the directory created??? (See next section)***

### Changing to a different directory (Command Name: `cd`)

The command `cd directory` means change the current working directory to '*directory*'. The current working directory may be thought of as the directory you are in, i.e. your current position in the file-system tree. We want to change the current working directory to **exercises**. We do it as follows

```
[user@localhost:~] $ cd exercises ↵
```

```
[user@localhost:~/exercises] $
```

## EXERCISE-1

### 1. Try the following commands now (and try to understand the results too)

- `ls`
- `ls ~`
- `ls ~/exercises`
- `ls ./..`
- `ls ~/..`

### 2. Create two subdirectories test1 and test2 under exercises

*Observe the two prompts above.*

- Before you enter the command you can see a `~` symbol towards the end of the prompt
- Once you entered the command you can find the `~` symbol is replaced by 'exercises'. 'exercises' is the present working directory. Then what does the `~` symbol mean?
- *`~` symbol stands for your home directory*

## EXERCISE-2

Consider you are in your home directory. Change to the exercises directory.

```
[user@localhost:~] $ cd exercises
```

Now create a new directory **exercise1** in **exercises** directory

```
[user@localhost:~/exercises] $ mkdir exercise1
```

Create a directory inside **exercise1** with a name **exercise2**

```
[user@localhost:~/ exercises] $ mkdir exercise1/exercise2
```

Go to **exercise2**

```
[user@localhost:~/ exercises] $ cd exercise1/exercise2
```

Create a directory inside **exercise2** with a name **test**

```
[user@localhost:~/exercises/exercise1/exercise2] $ mkdir test
```

Try to execute the following commands and observe the results

```
(a)[user@localhost:~/exercises/exercise1/exercise2] $ ls .
```

```
(b)[user@localhost:~/exercises/exercise1/exercise2] $ ls ../..
```

```
(c)[user@localhost:~/exercises/exercise1/exercise2] $ ls ../../..
```

Recall that `.` indicates present working directory and `..` indicates the parent of present directory.

## LONG LISTING OF A DIRECTORY

1. In this section you will learn various options which can be used with **ls** command to provide various file related information.

**a) ls** command when used with **-a** option lists all the files in your directory including those that start with a dot(`.`), such files are hidden files. Execute the following command –

```
[user@localhost:~/ Lab2]$ ls -a
```

(Please give a space between **ls**, and **-a**), observe the output

**b) ls** command when used with **-l** option displays [Unix file types](#), permissions, number of [hard links](#), owner, group, size, date, and filename. Execute the following command –

```
[user@localhost:~/ Lab2]$ ls -l
```

(Please give a space between ls, and -l), observe the output

## **GENERAL PURPOSE UNIX COMMANDS**

1. The **clear** command- Used to clear the screen.
2. The **date** command – displays today's date. Execute the following command and observe the result. You will see today's date.

```
[user@localhost:~/ Lab2]$ date
```

3. The **who** command – displays all the users who are currently logged in **your system** (online). Execute the following command and observe the result.

```
[user@localhost:~/ Lab2]$ who
```

4. The **who am i** command – displays your own identity. Execute the following command and you will be able to see output as your own user identity.

```
[user@localhost:~/ Lab2]$ who am i
```

5. The **cal** command – It is a standard program on [UNIX](#) and [UNIX](#)-like operating systems that prints an [ASCII](#) calendar of the given month or year. Execute the following commands and see the output.

a. **[user@localhost:~/ Lab2]\$ cal 2010 (give space between cal, 2010)**

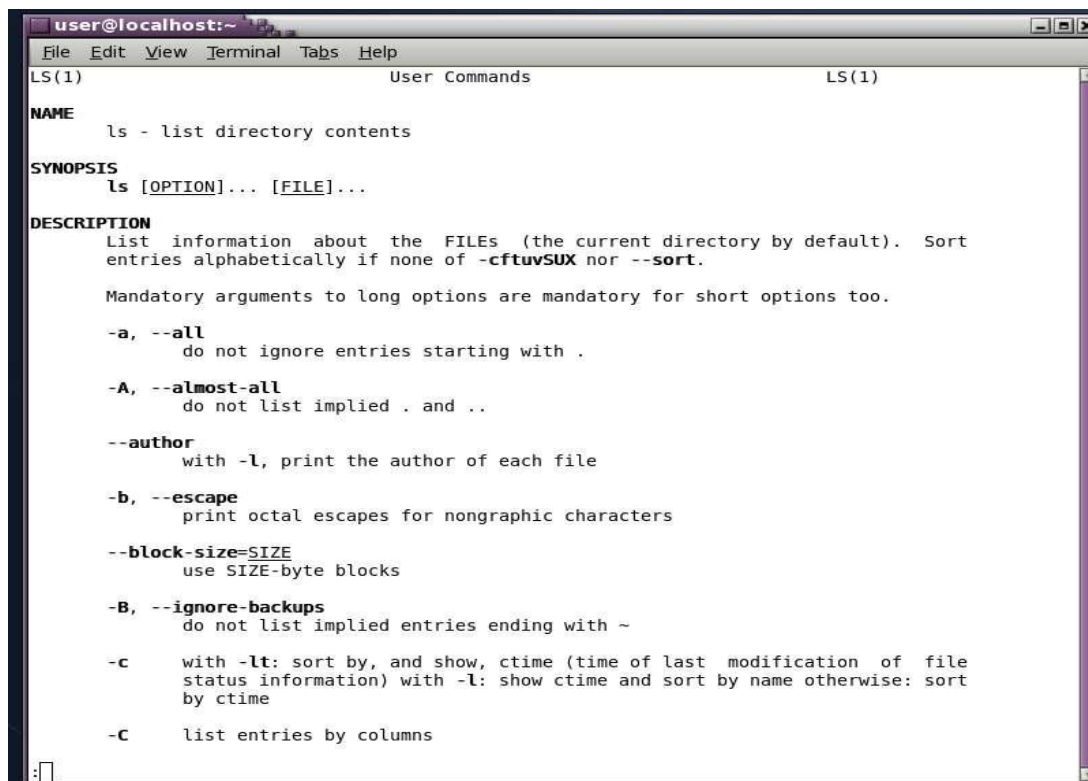
b. **[user@localhost:~/ Lab2]\$ cal 2 2010 (give space between cal, 2, and 2010)**

6. The **man** command- provides reference information on topics, such as commands, subroutines, and files. The **man** command provides one-line descriptions of commands specified by name. The **man** command also provides information on all commands whose descriptions contain a set of user-specified keywords. This is the command which you will be using to get any information about inbuilt functions while writing C programs.

Example: Type the following

```
[user@localhost:~]$man ls
```

**You will see the following**



```
user@localhost:~  
File Edit View Terminal Tabs Help  
LS(1) User Commands LS(1)  
  
NAME  
ls - list directory contents  
  
SYNOPSIS  
ls [OPTION]... [FILE]...  
  
DESCRIPTION  
List information about the FILES (the current directory by default). Sort  
entries alphabetically if none of -cftuvSUX nor --sort.  
  
Mandatory arguments to long options are mandatory for short options too.  
  
-a, --all  
do not ignore entries starting with .  
  
-A, --almost-all  
do not list implied . and ..  
  
--author  
with -l, print the author of each file  
  
-b, --escape  
print octal escapes for nongraphic characters  
  
--block-size=SIZE  
use SIZE-byte blocks  
  
-B, --ignore-backups  
do not list implied entries ending with ~  
  
-c  
with -lt: sort by, and show, ctime (time of last modification of file  
status information) with -l: show ctime and sort by name otherwise: sort  
by ctime  
  
-C  
list entries by columns
```



Which tells you details how to use ls command. Letter or string starting with '-' are optional with ls command.

**Use Ctrl z to come out of manual page.**

So what you understand here is, the structure of UNIX command is

**<command name><space><option1><space><option2>.....**

So, play with man command to understand UNIX commands in detail. Try out the options you can use with the commands you have already learnt. What happens when you type **man man ?**

## **File Related Operations & Commands**

### **INTRODUCTION**

In this session we will learn how to create files, display its contents and file maintenance commands. Apart from this we will learn some more UNIX commands.

### **CREATE and DISPLAY THE FILE**

First create a directory named Lab1 using mkdir command. Now go to directory Lab1 using cd command and start working on files as shown below. Type the following commands in sequence

**Step1:[user@localhost:~/ Lab1]\$ cat > first.txt**

This is CP first lab and I am creating the file

Learning commands is fun

Hard to memorize,

Practice makes me perfect

**Step 2: Press** Ctrl + d

**Step 3: [user@localhost:~/ Lab1]\$ ls**

You will see file called first.txt.

**Step4: [user@localhost:~/ Lab1]\$ clear**

**Step5:[user@localhost:~/ Lab1]\$ cat first.txt**

You will see whatever you typed previously on the screen.

**Step6:[user@localhost:~/ Lab1]\$ Similarly create another file called second.txt with some new text.**

**Step 7:[user@localhost:~/ Lab1]\$ cat first.txt second.txt**

You will see the contents of the two files one after the other.

[The command named cat – short for concatenate – concatenates contents files named as arguments and displays their contents on the display. If no argument is given the input is taken from the standard input (keyboard). In the first step, the output of cat is then redirected (by the '>' operator) to the file named first.txt. Ctrl+d is the End-of-File character.]

### **Logging out of your computer**

- (a) Type the command **logout**. The command window will be closed.
- (b) Go to the **System** menu at the top of the desktop and click **Logout user**. You should see the login screen if you are successfully logged out.