

## Computing Systems - Reading List

**Module Name:** Introduction to an Operating System

**Lesson 1:** Understand working of Linux OS

**Recommended Reading Material**

**Brief:** Notes attached with this document

**Book:**

**Remote access link (Optional):**

**Duration:** 20 minutes

**Module Name:** Introduction to an Operating System

**Lesson 2:** Basic UNIX commands

**Recommended Reading Material**

**Brief:** Notes attached with this document

**Book:**

**Remote access link (Optional):**

**Duration:** 20 minutes

## Table of Content:

- Introduction (Computing Machine, Operating System, UNIX)
  - Getting started with LINUX
    1. Opening a Terminal Window
    2. Changing password
    3. UNIX treats everything as Files!!!
    4. Print Working Directory (pwd command)
    5. List the contents of current directory (ls command)
    6. Manual Pager (man command)
    7. Creating a Directory (mkdir command)
    8. Changing to a different Directory (cd command)
    9. Knowing date, time and calendar (date command and cal command)
    10. Additional Reading
- 

## Introduction

Computer is a programmable machine, capable of performing set of predefined operations. It consists of hardware and software. Hardware can loosely be defined as piece of metal when dropped makes sound. The bare piece of metal has no intelligence and to make the computing machine work, we need hardware as well as software.

As we know that a given computing machine has number of input/output devices, memories processor etc. These are valuable resources of the computing machines. These resources are to be efficiently utilized and the user has to be provided with the interface with which it can interact with the computer in easy manner. Operating system is software which acts as a resource manager and provides the user interface. Current operating system offers two different kinds of user interfaces. The UNIX operating system offers Command line interface as well as Graphical user interface. Here in this reading material, we would discuss some of the basic commands related with files and directory operations.

Computer has mainly two different types of memories i.e., volatile (the content of memory is lost when power is switched off; e.g. RAM) and non-volatile memory (the content of memory is preserved even after the removal of power supply. It is also known as persistent memory; e.g. Hard Disk). The information is usually stored in form of files in

persistent memory, whenever it is required; it is brought in main memory by the operating system. In the simplest form, the files can be defined as collection of related information. The related files can be stored in a folder/ directory. The various file and directory operations are discussed in following sections.

## **Getting started with LINUX**

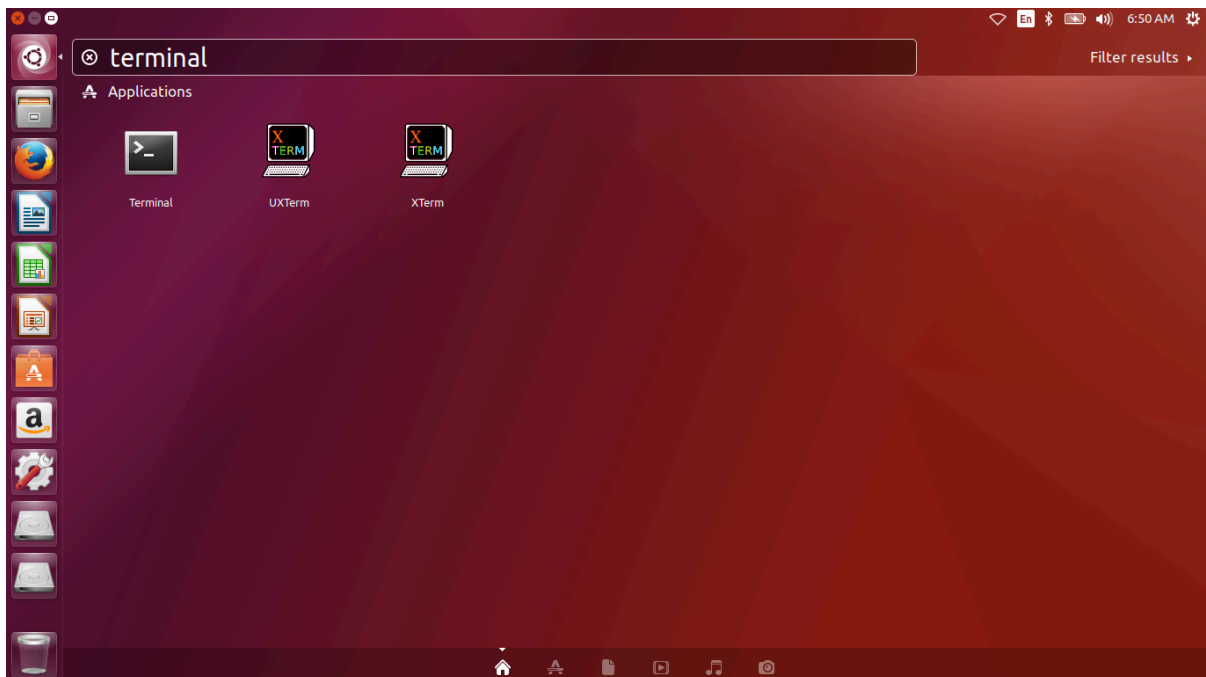
### **1. Opening a Terminal Window**

After logging in you will get a desktop screen similar to the one shown in Figure 1.



**Figure 1: Desktop of a Linux system after logging in**

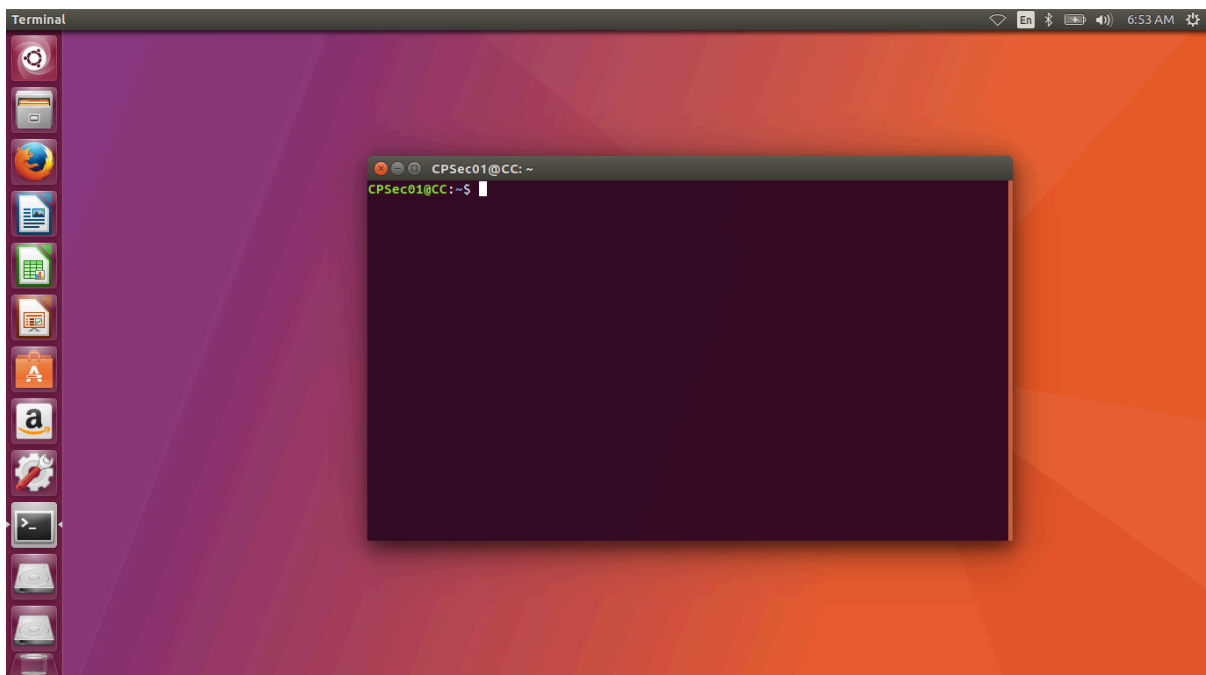
To open Terminal, either press Ctrl+Alt+T on your keyboard or go to Search by clicking the top left icon and search for Terminal. Figure 2 shows the method to open a terminal window.



**Figure 2: Method to open a Terminal Window**

In terminal Window you will find a prompt like this (as shown in Figure 3 and 4):

**CPSec01@Z – 111:~\$**



**Figure 3: A terminal Window**



Figure 4: A terminal Window

## 2. Changing password of your account (*passwd*)

Type *passwd* at the prompt

```
CPSec01@Z – 111:~$ passwd
```

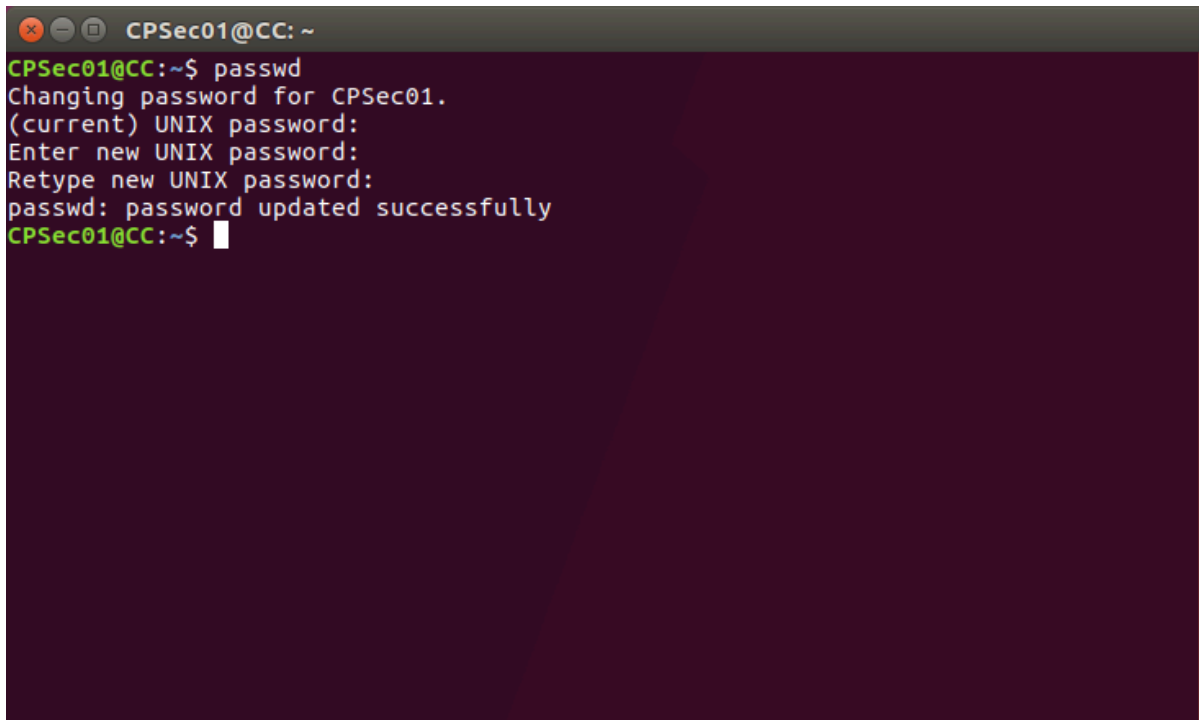
Changing password for CPSec01.

(Current) UNIX password: **<Enter your current password here>**

Enter new UNIX password: **<Enter your new password here>**

Retype new UNIX password: **<Enter your new password again here>**

**passwd: password updated successfully**



```
CPSec01@CC: ~  
CPSec01@CC:~$ passwd  
Changing password for CPSec01.  
(current) UNIX password:  
Enter new UNIX password:  
Retype new UNIX password:  
passwd: password updated successfully  
CPSec01@CC:~$
```

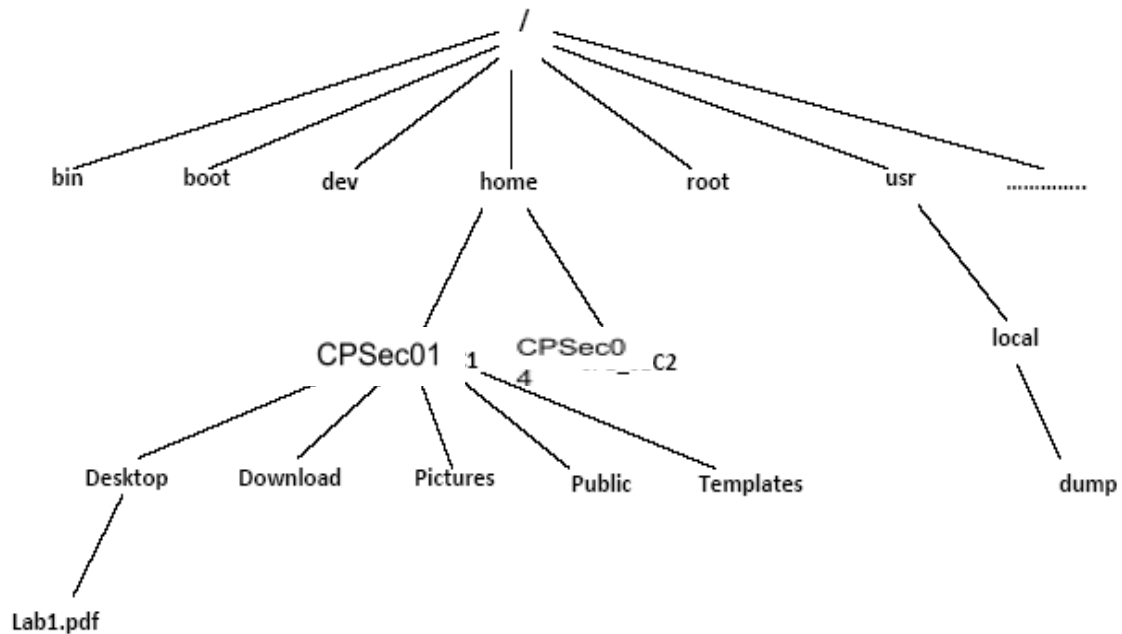
Figure 5: execution of passwd command

### 3. 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 collection of data. They are created by users using text editors, running compilers etc. **Examples of files:**

- a. A document (report, essay etc.)
- b. The text of a program written in some high-level programming language (C, Pascal, Java, etc...) instructions comprehensible directly to the machine and incomprehensible to a casual user, for example, a collection of binary digits (an executable or binary file);

All the files are grouped together in the directory structure. The file-system is arranged in a hierarchical structure, like an inverted tree as shown in Figure 6. The top of the hierarchy is traditionally called **root** (written as a slash / )



**Figure 6: Example of a typical Linux File System**

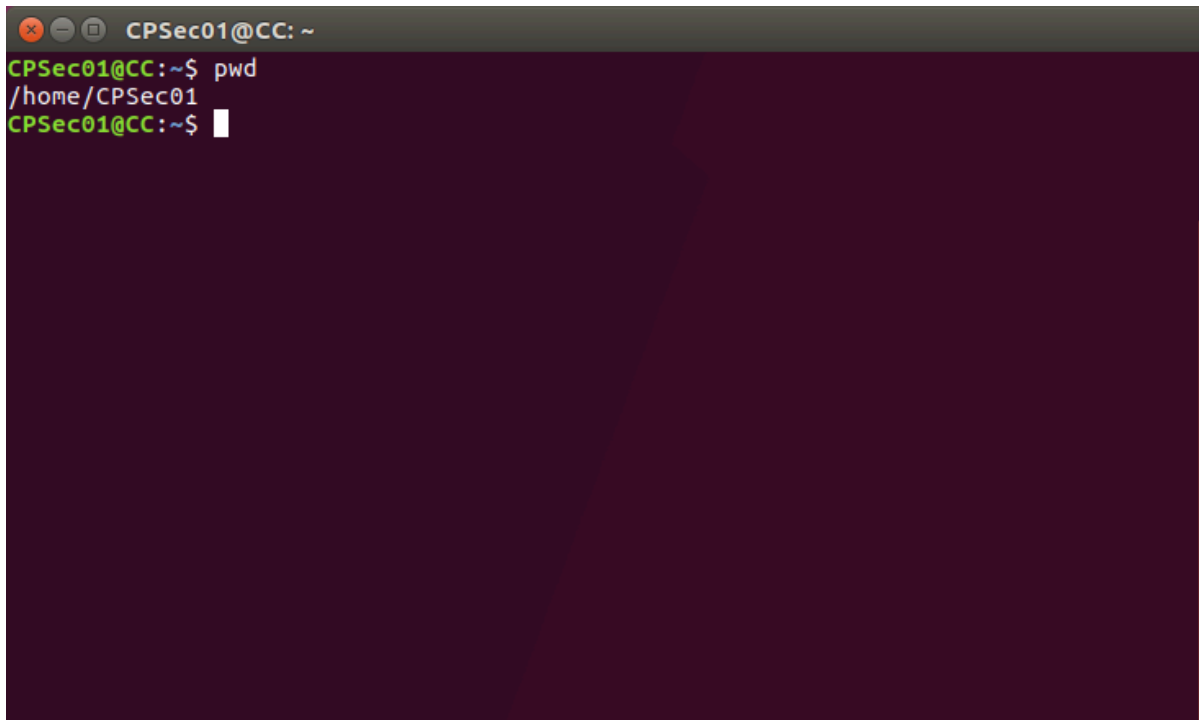
Once you open a terminal window, you will *most likely be* in the directory named by your user-name. But how do I verify this? Read next section...

#### **4 . Print Working Directory (*pwd* command)**

Type *pwd* at the prompt

```
CPSec01@Z - 111:~$ pwd
```

```
/home/CPSec01
```



```
CPSec01@CC: ~  
CPSec01@CC:~$ pwd  
/home/CPSec01  
CPSec01@CC:~$
```

**Figure 7: Execution of pwd command**

Try to match this path with the directory structure shown above. You get the path name (where you are in relation to the whole directory structure) of your *home directory*.

***Home Directory???***

When you open a terminal window, your current working directory is your home directory. Your home directory has the same name as your user-name.

You would like to find out what is inside your home directory, Fine. Proceed to next section.

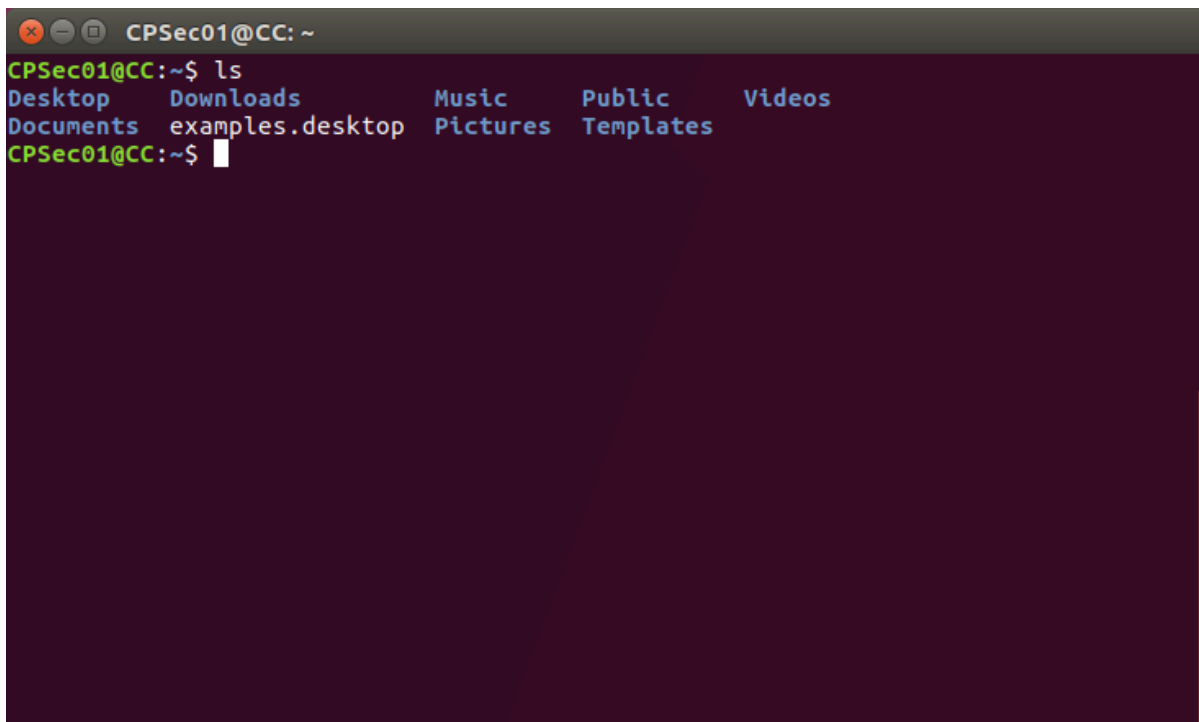
**5. List the contents of the current directory (*ls command*)**

Type ***ls*** at the prompt

```
CPSec01@Z - 111:~$ ls
```

You will get the display similar to figure 8. You will immediately see the prompt back.





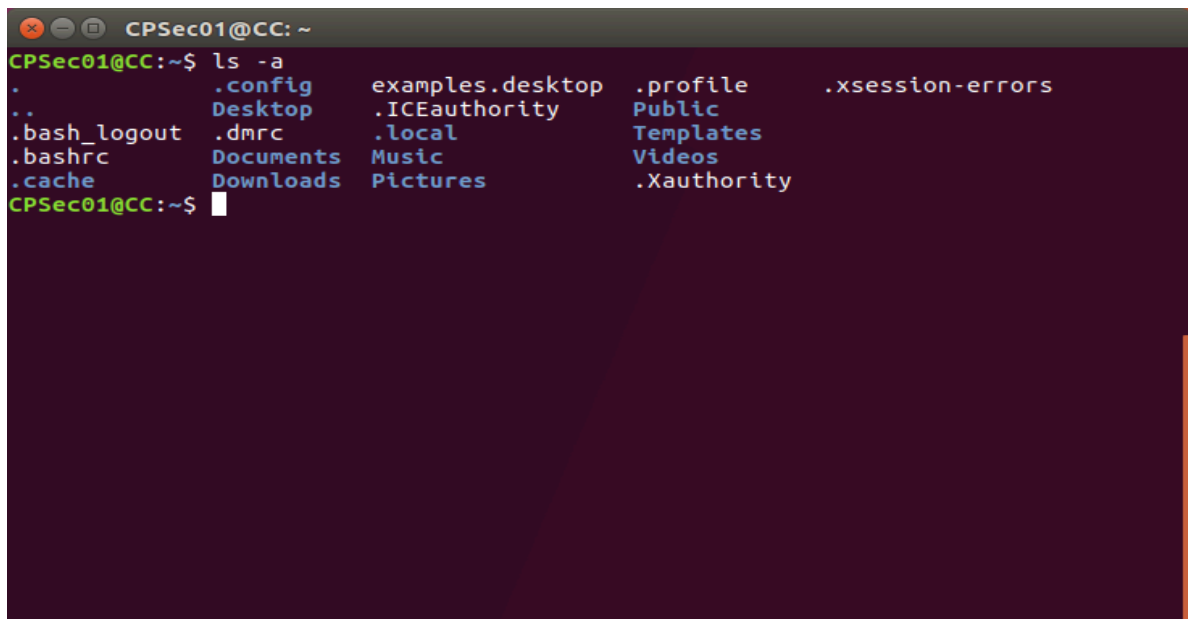
```
CPSec01@CC: ~  
CPSec01@CC:~$ ls  
Desktop    Downloads  Music      Public     Videos  
Documents  examples.desktop  Pictures   Templates  
CPSec01@CC:~$
```

**Figure 8: Output of “ls” command**

Your home directory is empty!!! Only the system defined directories are available. It's empty because you didn't create anything. But how can we make sure, that my home directory has nothing hidden? We can find it out using.

**CPSec01@Z - 111:~\$ *ls -a***

The output of the ***ls -a*** is shown in Figure 9.



```
CPSec01@CC: ~  
CPSec01@CC:~$ ls -a  
.  
..  
.bash_logout  
.bashrc  
.cache  
.config  
Desktop  
.dmrc  
Documents  
Downloads  
examples.desktop  
.ICEauthority  
.local  
Music  
Pictures  
.profile  
Public  
Templates  
Videos  
.Xauthority  
.xsession-errors  
CPSec01@CC:~$
```

**Figure 9: Output of “ls -a” command**

Observe the result of **ls -a** command. All the hidden files listed as result starts with a DOT (.). These files are hidden files. (Do not fiddle with these files now!!!). Also observe the first two results. A Single dot (.) and a double consecutive dots (..). In UNIX, (.) means the current directory and (..) means the parent of the current directory.

Try typing the command,

```
cpsec2@biju:~$ ls .
```

```
cpsec2@biju:~$ ls ..
```

*What is your observation from the result of the above two commands?*

```
CPSec01@CC: ~  
CPSec01@CC:~$ ls .  
Desktop    Downloads    Music        Public       Videos  
Documents  examples.desktop  Pictures     Templates  
CPSec01@CC:~$ ls ..  
CPSec01    lost+found   user  
CPSec01@CC:~$
```

**Figure 10: Output of ls . and ls .. command**

The **ls .** gives the same result as **ls** (displays all the files and directories in the current directory)

The **ls ..** displays the directories and files in the parent directory as shown in Figure 10.

How to explore all the options of **ls**? Fortunately Linux need not depend on Google for this!! Linux maintains manual pages which acts as the interface to the on-line reference manuals. Read more about it in next section.

## 6. Manual Pager (*man command*)

**man** is system's manual pager. The name of the program is given as argument to **man**. For e.g. **man ls** will display the on-line manual page of **ls** command.

```
CPSec01@Z - 111:~$ man ls
```

The **man ls** displays the manual pages of **ls** as shown in Figure 11.

```
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 is specified.

  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 C-style escapes for nongraphic characters

  --block-size=SIZE
      scale sizes by SIZE before printing them; e.g., '--block-size=M' prints sizes in units of 1,048,576 bytes; see SIZE format below

  -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, newest first

  -C      list entries by columns

  --color[=WHEN]
      colorize the output; WHEN can be 'always' (default if omitted), 'auto', or 'never'; more info below

  -d, --directory
  Manual page ls(1) line 1 (press h for help or q to quit)
```

**Figure 11: Output of man ls command**

Is man command the only one to know information about a program in Linux? Not really!!  
We can also get the information document using info command as well.

**CPSec01@Z - 111:~\$ info ls**

The **info ls** displays the information pages of ls as shown in Figure 12.

```
Next: dir invocation, Up: Directory listing

10.1 'ls': List directory contents
=====

The 'ls' program lists information about files (of any type, including
directories). Options and file arguments can be intermixed arbitrarily,
as usual.

For non-option command-line arguments that are directories, by
default 'ls' lists the contents of directories, not recursively, and
omitting files with names beginning with '.'. For other non-option
arguments, by default 'ls' lists just the file name. If no non-option
argument is specified, 'ls' operates on the current directory, acting as
if it had been invoked with a single argument of '.'.

By default, the output is sorted alphabetically, according to the
locale settings in effect.(1) If standard output is a terminal, the
output is in columns (sorted vertically) and control characters are
output as question marks; otherwise, the output is listed one per line
and control characters are output as-is.

Because 'ls' is such a fundamental program, it has accumulated many
options over the years. They are described in the subsections below;
within each section, options are listed alphabetically (ignoring case).
The division of options into the subsections is not absolute, since some
options affect more than one aspect of 'ls's operation.

Exit status:

0 success
1 minor problems (e.g., failure to access a file or directory not
specified as a command line argument. This happens when listing a
directory in which entries are actively being removed or renamed.)
2 serious trouble (e.g., memory exhausted, invalid option, failure
to access a file or directory specified as a command line argument
or a directory loop)

Also see *note Common options::.

* Menu:
-----Info: (coreutils)ls invocation, 57 lines --Top-----
Welcome to Info version 6.3. Type H for help, h for tutorial.
```

**Figure 12: Output of info ls command**

## 7. Creating a Directory (*mkdir command*)

We will create a subdirectory in 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

```
CPSec01@Z - 111:~$ mkdir exercises
```

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

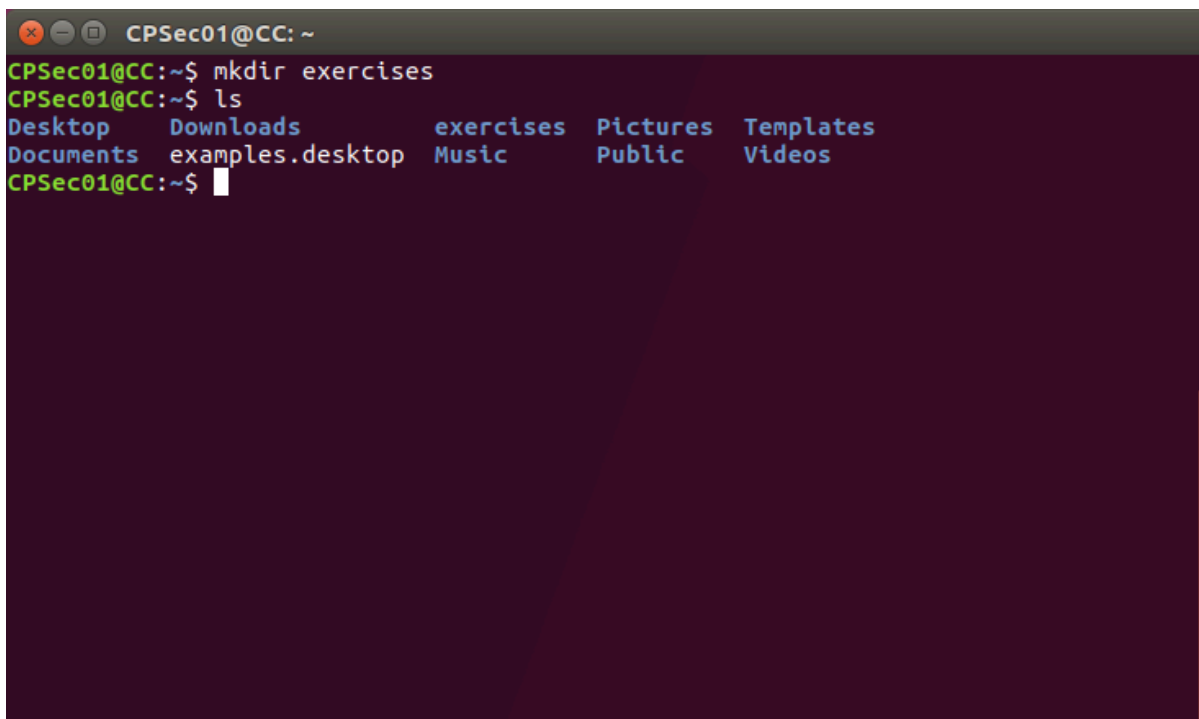
```
CPSec01@Z - 111:~$
```

But, how do we make sure that if the directory named 'exercises' is created or not???

We have **ls** command 😊

**CPSec01@Z - 111:~\$ ls**

You can find the listing of your directory along with the precious **ls** output.



```
CPSec01@CC: ~  
CPSec01@CC:~$ mkdir exercises  
CPSec01@CC:~$ ls  
Desktop    Downloads  exercises  Pictures    Templates  
Documents  examples.desktop  Music      Public      Videos  
CPSec01@CC:~$
```

**Figure 13: Output of mkdir and ls command**

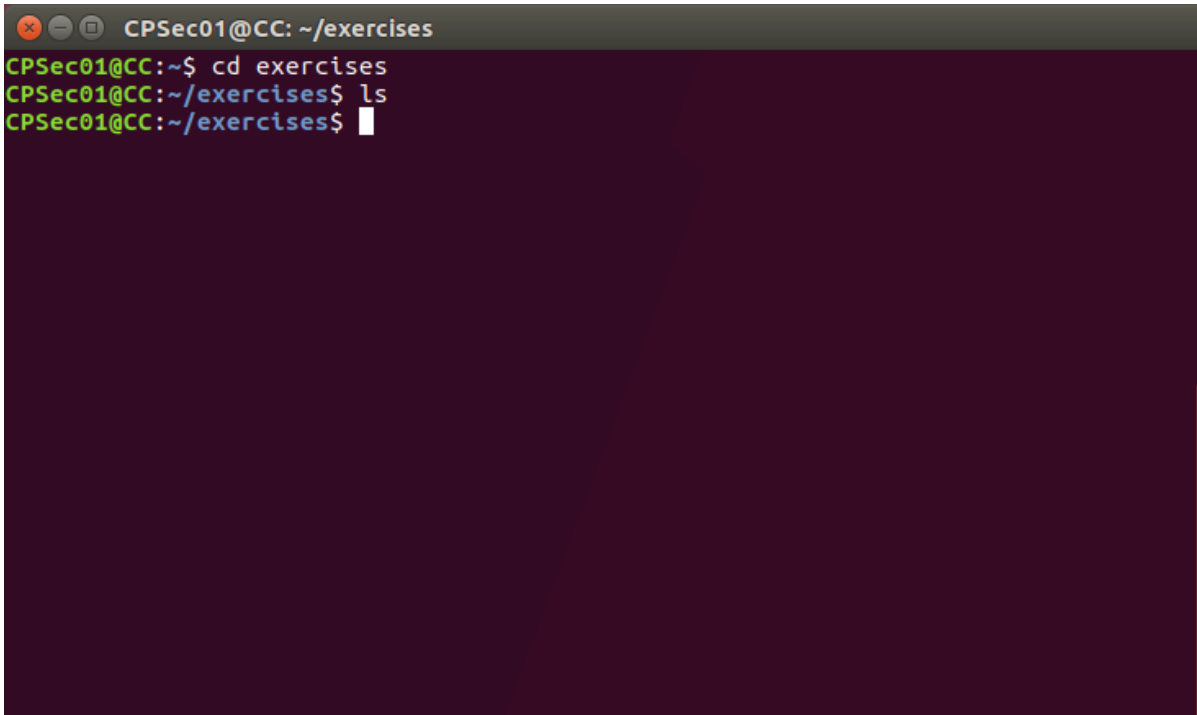
How do we go to the directory created??? Read on section-8.

## 8. **Changing to a different directory (*cd command*)**

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

CPSec01@Z - 111:~\$ **cd exercises**

CPSec01@Z - 111:~/exercises\$



```
CPSec01@CC: ~/exercises
CPSec01@CC:~$ cd exercises
CPSec01@CC:~/exercises$ ls
CPSec01@CC:~/exercises$
```

Figure 14: Output of cd exercises and ls command

**Observe the two prompts above.**

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

## 9. Knowing Date, Time and Calendar (*date command and cal command*)

The date command displays the current date and time in the given format or sets the system date.

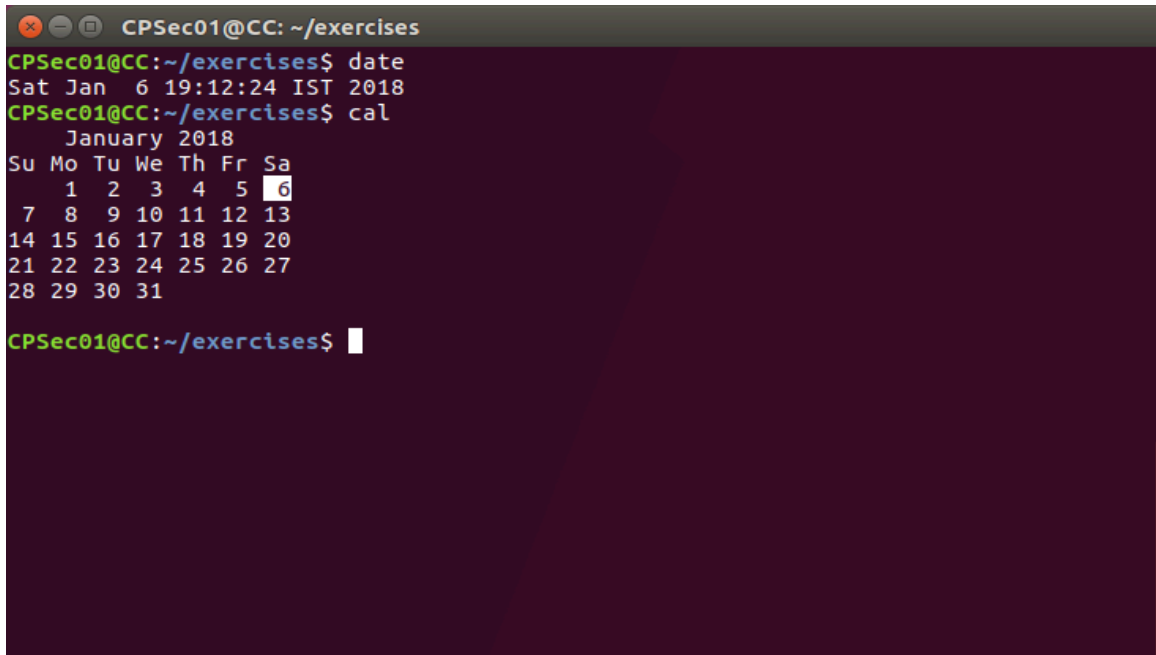
CPSec01@Z - 111:~\$ **date**

It displays the date as shown in Figure 15. You can explore more options of date using man page.

The cal command displays a simple calendar in traditional format.

**CPSec01@Z - 111:~\$ cal**

It displays the date as shown in Figure 15. You can explore more options of cal using man page.



```
CPSec01@CC: ~/exercises
CPSec01@CC:~/exercises$ date
Sat Jan  6 19:12:24 IST 2018
CPSec01@CC:~/exercises$ cal
    January 2018
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 31
CPSec01@CC:~/exercises$
```

**Figure 15: Output of date and cal commands**

## 10. Additional Reading

### **Absolute and Relative Path Addressing:**

All platforms have a method for describing the paths for files and directories. You usually specify a path as a series of names separated by separator characters. Typically, the first name is the top level of the hierarchical specification of the path. The last name is the file or directory the path identifies.

A path can be one of the following types:



1. Relative path
2. Absolute path

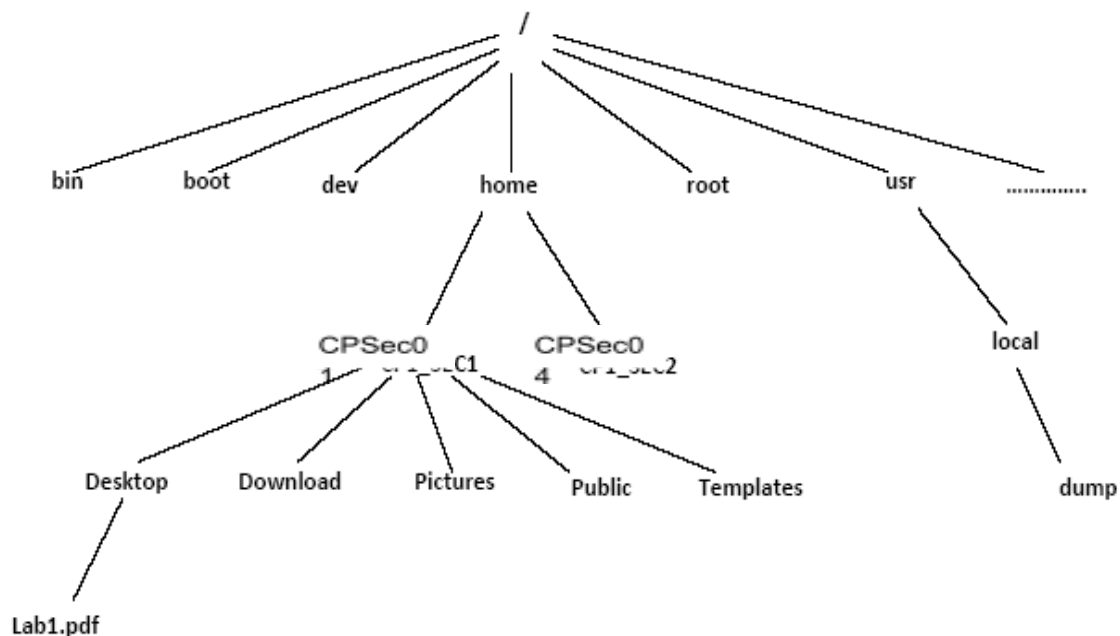
A relative path describes the location of a file or directory relative to an arbitrary location in the file system. An absolute path describes the location of a file or directory starting from the top level of the file system.

A path does not necessarily go from the top of the hierarchy down to the target. You can often use a platform-specific tag in place of a name that indicates the path should go up a level from the current location.

**Certain Linux-specific tags:**

- ~ Home Directory (of the user logged in)
- / Root Directory (note: it is different from the directory named “root” in / directory )
- . Current Directory
- .. Parent Directory

Consider the hierarchy tree given below:



Suppose currently you are in the directory “Downloads”. The PWD command shall give you the following:

/home/CPSec01/Downloads.

Now, if you want to access the contents of the directory “*Public*”, you can use either **absolute** or **relative** addressing to meet the purpose.

Talking about relative addressing, one thing to remember in mind is that you can travel only along the links. Since there is no link between “*Downloads*” and “*Public*”, you can’t access “*Public*” directly. So, you shall first traverse through the node “*CPSec01*” (the parent directory accessed with the help of ..) and then reach “*Public*”.

So the path would be: “*../Public*”.

Now assume you were in the Directory “*Home*”. Then your path would be:

“*CPSec01/Public*”

Thus, depending on your relative location in the hierarchy tree, your path to the destination address changes. Hence the name, **Relative Path Addressing**.

Now, consider a situation in which you are not so familiar with the hierarchy of the tree.

Then you can alternatively follow Absolute Path Addressing, starting from root directory, i.e. “*/*”

The path would be “*/home/CPSec01/Public*”.

The important thing to note is that when you want to reach a destination node from the root directory, there is one and only one path leading to it. Hence the name, Absolute Path Addressing.