KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

COLLEGE OF COMPUTER SCIENCES AND ENGINEERING

INFORMATION AND COMPUTER SCIENCE DEPARTMENT

ICS 431: OPERATING SYSTEMS (LAB) (3-3-4)

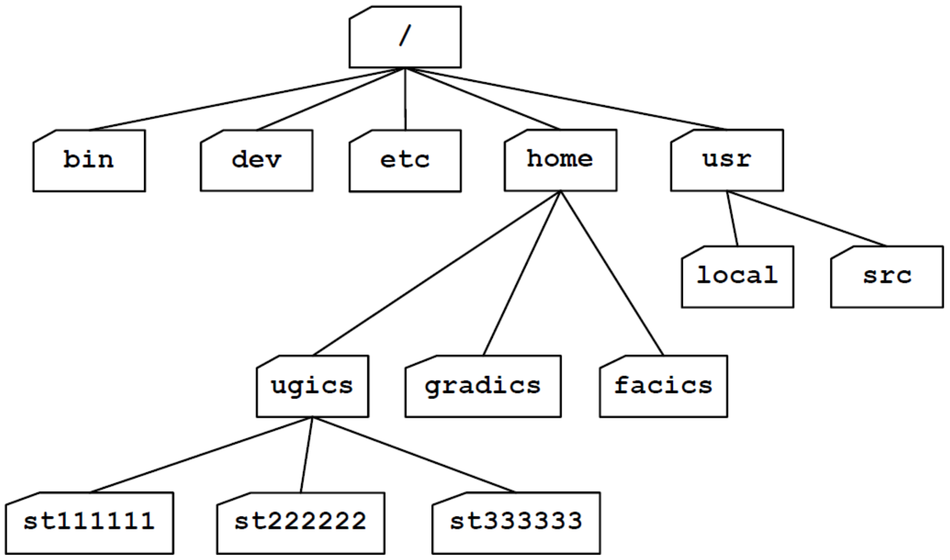
LAB 01

Introduction to Linux

Objectives:
This Lab is for new users of the Linux system. The students assumed to have some prior experience with computers, but not necessarily with Linux. The followings are the primary objectives of this lab session:

- Linux File System
- Create and maintain files and directories of files.
- Create and modify a text file

Linux File System
A file system is a logical organization of storage space designed to contain files in directories. The Linux file system is quite similar to that of MS-DOS or Windows. It is organized hierarchically (inverted tree) into directories for efficient organization. However, in Windows, there are many logical trees represented by drive letters such as C:, D:, H:, etc… . In LINUX, all file systems (Hard Disks, CD-ROMs, Floppy Disks, ZIP drives, network mounts, etc …) are "mounted" onto one logical tree. The top of the hierarchy is traditionally called **root** which is represented by a ***/*** (**slash**).



Part of file-system tree

In Linux, everything is treated as a file. A directory is a file. It is a file that contains a list of files and information belonging to those files. This would include things like who "owns" (created) the file, how long it is, and who can use it. Since a directory is simply a list of files, it can contain any file in it, including other directories.

Absolute and Relative Names

You can specify a file or directory by its **path name**. There are two ways of expressing the path name: **Full (absolute)** path name or **relative** path name. The full path name starts with the root. /, and follows the branches of the file system, each separated by /, until you reach the desired file.
However, a relative path name specifies the path relative to your current working directory. Relative path names are more convenient because they are shorter, but must be used with care. They never begin with / (slash). Now, we have to introduce two special directory entries:
. the current directory
.. the parent directory

Accessing Linux Shell

You may access the shell by starting a terminal.

Starting a new terminal session

To open a terminal window, click on Applications -> Accessories -> Terminal. A terminal window will then appear with a prompt, waiting for you to start entering commands:

Linux Commands:

1. Listing files and directories

ls (list)

When you start a terminal session, your current working directory will be your HOME directory. Your home directory has the same name as your username.
To find out what is in your home directory, type:

```
> ls
```

There may be no files visible in your home directory, in which case, the Linux prompt will be returned. Alternatively, there may already be some files inserted by the System Administrator when your account was created.

ls does not, in fact, cause all the files in your home directory to be listed, but only those ones whose name does not begin with a dot (**.**). Files beginning with a dot (**.**) are known as hidden files and usually contain important program configuration information. They are hidden because you should not change them unless you are very familiar with LINUX!!!

To list all files in your home directory including those whose names begin with a dot, type:

```
> ls -a
```

The –a indicates an option to show ***all*** files in your home directory.

ls is an example of a command which can take options: **-a** is an example of an option. The options change the behavior of the command. There are online manual pages that tell you which options a particular command can take, and how each option modifies the behavior of the command. (See later in this Lab)

Type the following command:

```
> ls /
```

The / character is the directory name for the **root** directory. What you should see are the subdirectories and any files that are located in the root directory.

2. Making Directories

mkdir (make directory)

We will now make a subdirectory in your home directory to hold the files you will be creating and using in this course.
To make a subdirectory called **ics431** in your current working directory type

```
> mkdir ics431
```

Now to see the directory you have just created, type

```
> ls
```

Please note that creating a subdirectory will not change your current position in the file system tree. Thus, after creating ics431, your current directory will still be your HOME directory.

3. Changing to a different directory

cd (change directory)

If you just logged into your Linux account, you are placed in your home directory. The cd (change directory) command is used to change your current directory (your current position) to another directory.
To change to the directory you have just made, type

```
> cd ics431
```

Type ls to see the contents (which should be empty)

Create a new directory named "lab01" inside the directory "ics431"

4. Pathnames

pwd (print working directory)

The pwd command is very useful to know your current position in the file system. It displays the full path name of your current working directory.

```
> pwd
```

What is your current working directory?

Be sure your current working directory is "lab01" before moving to next section

5. Displaying the contents of a file on the screen

clear (clear screen)

At the prompt, type: `> clear`
This will clear all text and leave you with the > prompt at the top of the window.

cat (concatenate)

The command **cat** can be used to display the contents of a file on the screen.
To view the content of *hello.txt* type:

```
> cat hello.txt
```

The cat command is useful for displaying short files of a few lines. To display longer files use **less** or **more** commands.

6. Copying Files

cp (copy)

The cp command allows you to copy a file from one location to another location. There are different syntaxes to use **cp** command as shown below:
Syntax 1: (copy a file to another file)

```
> cp file1 file2
```

where ***file1*** is the name of an existing file and ***file2*** is the name for the new copy of that file. The original file will remain unchanged and a copy will be placed in ***file2***. If ***file1*** and ***file2*** are not in current directory, then you have to specify its pathname.

Syntax 2: (copy a file to another directory)

```
> cp file directory
```

where ***file*** is the name of an existing file and ***directory*** is the name for the destination directory. The original file will remain unchanged and a copy will be placed in that ***directory***.

Be sure your current working directory is lab01 before moving to examples

Create a new directory named “backup” inside your current directory

To create a backup of **hello.txt** by copying it to a file called **test.txt**, type:

```
> cp hello.txt test.txt
```

Now to see the file you have just created, type:

```
> ls
```

Observe that you have two files in your current directory; **hello.txt** and **test.txt**.
To copy the file named **hello.txt** to the directory names "**ics431**", type:

```
> cp hello.txt ..
```

Now to see the file you have just copied, type:

```
> ls ..
```

To put a copy of **hello.txt** into your backup directory, type

```
> cp hello.txt backup
```

Now to see the file you have just copied, type:

```
> ls backup
```

7. Moving Files

mv (move)

To move a file from one place to another, use the **mv** command. This has the effect of moving rather than copying the file, so you end up with only one file rather than two.
It can also be used to ***rename*** a file, by moving the file to the same directory, but giving it a different name.
We are going to move the file **test.txt** to your backup directory, type

```
      > mv test.txt backup
```

Now what is the content of your current directory? What is the content of your backup directory?

8. Removing Files and Directories

rm (remove)

rmdir (remove directory)

To delete (remove) a file, use the **rm** command. However, to delete a directory use the **rmdir** command. There is one constraint to delete a directory; the directory must be empty.
We know that your backup directory contains two files named **hello.txt** and **test.txt**. We are going to delete the files

```
      > rm backup/hello.txt
      > rm backup/test.txt
```

You can delete the backup directory. To delete the backup directory, type:

```
      > rmdir backup
```

9. Getting Further Help: man

man (manual display)

There is plenty of useful Linux commands and utilities in which detailed listing would be out of this lab's scope. Some of these commands might be very convenient, so it is advisable that you learn them on your own. Fortunately, there is a way of searching online manual of all Linux commands and utilities. The **man** command (shortcut for manual) displays in detail information about a given command and allows searching for related commands to specific keywords. **man** command is very useful in displaying online manuals of Linux commands.

Syntax

man [-] [-k keywords] topic **-**	Displays the manual without stopping.
-k keywords	Searches for keywords in all of the manuals available.
topic	Displays the manual for the topic or command typed in.

10. Some additional commands:

1) **history** Shows the list of all previously entered commands. Try
   > history 4
   (Displays the previously entered *n* commands)
   and
   > history -c
   (Clears the history)
2) **wc** is short for wordcount. Try
   > wc filename
   on any text file.
3) **grep** pattern filename
   finds a pattern matching in a given file. For example, try
   > grep ls newfile
4) **sort** filename
   sorts a given file. For example, try
   > sort newfile

Input and Output redirection

You may send your output (redirect it to a file by using the > operator as follows):

$**> ls > listOffiles**

The text-file listOfFiles now contains the output of ls
You may also get input from a file using the < operator as follows

$**> wc < listOfFiles**

The two may also using a **<u>Pipe</u>**

$**> ls | wc**

The vertical bar is the notation of a “pipe”. Informally it means that the output of the first command becomes input to the second command.

Exercise

1. Practise with some of the linux commands mentioned above.
2. Using the above commands (and/or other commands) find and display the three largest files by size in your folder. (Hint: use ls, grep, cat and/or sort).