Lab 1: UNIX, Editors, WEB, Mail, Newsgroup

1 Objectives

- Familiarize yourself with basic UNIX commands and learn how to use them.
- Discover the structure of the UNIX commands and where to look for help and instructions.
- Find out how to use redirection and piping.
- Practice work with several editors.
- Learn to use Network applications under UNIX.

2 Introduction

- Organization: basic information, books, assignments, marking, etc.
- **UNIX and its history:**
  See

  http://www.unix-systems.org/what_isi_unix.html

  http://vertigo.hsrl.rutgers.edu/ug/unixi_history.html

  http://www.ugu.com/

  for nice pages about history of UNIX and further links to UNIX documentation, specification, books etc.
- **Setting up your account:**

  You must be given a username and a password before receiving access to the system. Visit MC 3011 for that purpose.

  Login into your account takes two steps: entering of user's ID and password. For example

  login: j2lastnm
  password:

  logs in the person with username *j2lastnm*. UNIX is case sensitive. If your username is *jimbo*, do not enter *JIMBO* or *Jimbo*. Your password is never displayed on the screen preventing other users from seeing it! Change your

password when you access the system for the very first time. To change your password type:

passwd

and follow the dialog:

Old password: enter your current password
New password: enter your new password
Retype new password: re-enter your new password

The password should be changed on a regular base. Do not write it down or tell it to anyone. You should NOT use any personal information such as name, birthday or any word from dictionary as your password. Rather combine it mixing randomly upper and lower cases with non-alphabetic characters.

3 Basic commands (15 minutes)

man is the most important command one should know. It prints on the screen the help page for the given command. Its format is

man nameofthecommand

So, for example, if help is needed for ls command, type:

man ls (enter)

The help page for man command itself is brought up by:

man man (enter)

Typing

man (enter)

without other parameters displays the required structure of the command. It is not useful for UNIX novices but a helpful memory refreshment for older users.

Beginners often do not know the name of command needed to accomplish their task. Suppose it is to send a file to the printer. To find the name of the relevant command type

man -k print

The man command with option k prints out one-line summaries from the reference manual that contain the keyword print. This provides at least some particular options to choose from, though it still may take time to find the name that suits the purpose best. Just out of curiosity, it is lpr that does the job.

Most texts consist of more than one pages, but only one page fits the screen. It will be helpful to see one page at the time. Suppose the desired, but long text is in the file named mytext. The command, which allows us to see one screen at the time is more. The format is

more mytext

The output of *man -k print* is not a file. It is just, say, the screen output. UNIX allows us to use the output from one command as an input to another command without saving it into a (temporary) file. To use the result from *man -k print* as input for the more command type

man -k print | more

Notice that *man -k print* precedes | and *more*, i.e. it is not written instead of the file name after *more*! Symbol | is called the **pipe**. It is the pipe that literally converts the output of *man -k print*, or generally of any command preceding |, into a file accepted by *more*, or a command following the pipe. Result from the previous command serves as an input of the second command. More examples with pipe will be later today and in lab 2, theory of piping is explained in cs354. Pipe allows chaining of commands into one arbitrary long command. For example,

man -k print | sort -u | more

extracts the text on printing, sorts the lines alphabetically according the first character on the line and displays the result page by page on the screen. Once again, notice that the commands are executed one after the other as they stand on the line. As an exercise, read the man page for sort and see, what option u does.

3.1 Other UNIX commands

3.1.1 Listing files, directories - ls, lc

The list *ls* command in UNIX is an equivalent to the *dir* command in Dos. ls shows all subdirectories and files in the current directory. By default it does not distinguish between files and directories. *ls -l* displays the list of all files in the current directory including their size, permissions etc. Directories differ from files by character 'd' in the first position on the line. '-' denotes a regular file.

Another handy command is *lc* because it lists directories and files separately. Type

lc

and check carefully the result. Notice the files you did not see with *ls*. They start with '.' and have a special meaning, such as setup files. The setup file

.cshrc used when you log in is the most important one. You will hear about it later today and in the next lab.

3.1.2 Changing the working directory, creating and removing a directory

Those already registered in cs241 have a directory called cs241 in the home directory. Terms like home directory, current directory, working directory etc. will be explained in lab 2. Use lc command to check, if the cs241 directory is in your home directory. All files related to cs241 course are normally kept in the cs241 directory. Change of one directory to another is achieved by the UNIX command

cd dirname

meaning change to directory dirname. Next, let us switch to the cs241 directory typing *cd cs241*. UNIX is not very friendly. By default it will not show if the working directory was changed. The pathname of the current directory is displayed by the UNIX command 'print working directory':

pwd

Type *pwd* and verify that you are currently in the directory /u/youruserid/cs241. It is recommended to create new directories for all your cs241 assignments and other directories related to cs241 course in the cs241 directory.

The command 'make directory', *mkdir*, creates a new directory.

Type:

mkdir temp

and switch to the newly created directory temp by typing: *cd temp* and display the path of temp using pwd. To switch back to cs241 directory type

cd ..

This is similar to DOS environment. Notice that space is needed between cd and the two dots. Command *cd ..* returns one level up to a higher directory, *cd* goes straight to the home directory (up or down).

Exercise 1:
Go to the temp directory created before, make a directory called lab1 and switch into it. Verify that your working directory is indeed lab1:

Solution:

cd
cd cs241/temp
mkdir lab1
cd lab1

pwd

A directory can be removed by the command

rmdir

For example, rmdir lab1 removes the directory lab1. The command must be applied from the directory containing lab1 as subdirectory.

4 Editors (15 minutes)

This section will discuss several editors that are available in UNIX. We start with two editors, emacs and xemacs working under X-Windows. Be sure the current directory is lab1 in subdirectory cs241. Type

emacs temp1

Emacs is the name of the editor and temp1 is the name of the file to be edited. The editor opens in a new window and the window with *emacs* command becomes disabled. You will learn how to fix this problem shortly, but for now, enter few lines of text in emacs. Once the file temp1 is finished, it has to be saved. Like with Notepad under Windows95/98 click 'File' in the menu, then 'Save buffer', again 'File' and 'exit'.

To use emacs editor and still have all other windows available for work, type

emacs temp1 &

& "runs your emacs editor in the background". Now you may use both windows: the editor and the window where you originally typed the command. & can be applied to other X-Windows applications such as Netscape. In the recommended book, 'Unix in a nutschell', chapter 7 covers emacs editor with detailed description of all its internal commands.

Another X-Windows editor is xemacs which differs from emacs by buttons similar to Windows95 applications. Type:

xemacs temp1 &

One should be also familiar with editors not based on X-Windows. They can be handy for work from home. When the network at school is busy they may be faster. Pico and vi are two such editors.

Type

pico temp2

Pico opens no additional window. Do not use & because it runs pico in background and the application becomes unavailable! Pico has a menu at the bottom, which makes the use very straightforward. Enter few characters and save and close the file. As you can see on the bottom, CTRL X will do the work.

vi is a powerful, but not user-friendly editor. It is fully described in ch. 8 of Unix in a Nutshell.

To create and edit a file called temp3 using vi editor, type

vi temp3

Nearly each key on the keyboard triggers an action in the editor, hence, do not push any keys without knowing their meaning. Once the vi editor is open, we are in the so called 'command mode'. The editor resists any typing until the 'insert mode' is entered, for example, by hitting the 'i' key. Hit 'i', switch into the insert mode and enter a new text in the file. Once you are done, use ESC key to exit the insert mode and return into the command mode. To save changes and quit the editor type

:wq (enter)

Many UNIX applications use vi by default. Hence, in Labs we will use vi editor mainly.

5 Copying, moving and removing files in UNIX

(10 minutes)

The *cp* command is used to copy a file. To create a copy of the file temp1 with name temp6 type

cp temp1 temp6

If the file temp6 already exists it will be overwritten without a warning. Therefore, one has to be careful. The command with option '-i',

cp -i temp1 temp6

prompts for confirmation before overwriting an existing file and is thus safer to use.

To remove the file temp6 use

rm temp6

The *mv* command renames files or moves files from one directory into another directory. For example, to rename file temp1 to temp4 execute:

mv temp1 temp4

As with *cp*, if temp4 already exists it will be overwritten.

A directory containing files cannot be deleted. It must be emptied first using*rm, mv* etc. The empty directory is deleted using *rmdir*.

Example:

cd ..
rm temp/temp4
rm temp/temp2
rm temp/temp3
rmdir temp

As an exercise you should familiarize yourself with other commands, such as *cat, more, less, head, tail* and *diff* that are useful for examining contents of files.

6 Network applications (5 minutes)

6.1 e-mail

This section presents two applications for reading and sending of e-mails. One program is known as *pine*. Its environment is similar to the pico editor and is self-explanatory. The options are at the bottom of the window. Type

pico

To send an e-mail press first *c*, then type the message and hit *CTRL x* to send it.

Another e-mail program is called *mail*.
Type

mail

The program will list the messages in your mailbox. To read message #3, for example, type

3 (enter)

Press

h (enter)

for help and

q (enter)

to quit the program. The *mail* software needs some practice but its knowledge pays off as you will see later.

6.2 WEB, newsgroup

Most people are familiar with the Netscape navigator also available in UNIX. Type

netscape &

to let it run in the background as any other X-windows application.

In case the X-windows environment is not available or it is very slow, knowledge of applications such as *lynx* allowing browse the web or read a newsgroup without X-windows is helpful. To read a newsgroup one may use pine or trn. For example

trn cs241

opens the cs241 newsgroup. Help while running trn is available by pressing 'H' To send a message to a newsgroup use 'f' or 'F', to read the next message, press 'n', to quit trn press 'q'.

7 The set up of environment variable for lynx (5 minutes)

An environment variable has to be set before one can use lynx for reading newsgroups. As a part of assignment 0 you have to set another environment variable in order to use applications needed for cs241 course. More detailed discussion of environment variables is in lab 2.

The command for setting environment variable is called *setenv* and must be placed in the file *.cshrc*. This is the first file executed when you log in. It is located in your home directory and you may get to it by typing:

cd

Then open .cshrc by typing

vi .cshrc

and what you see is a shell program. Shell will be discussed in next two labs.

Find the block with setenv where other environment variables are defined and move the cursor to that block. Pressing 'o' will add a new line and switch into editing mode. Now type the comment line:

 # Used by lynx to read news.

And the command itself:

setenv NNTPSERVER news.math.uwaterloo.ca

After the editing is done, press ESC to switch into command mode and type:

:wq

to save the file and quit editor.
In case you made errors, quit without saving the changes using

:q!

and repeat the above procedure correctly.
To execute *.cshrc* you must log out and log in again.