

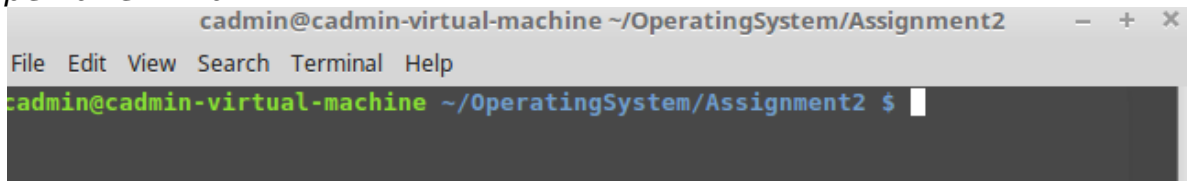
**Mohammed Jahangir Alom**  
**Student Number – R00144214**  
**Module – Operating Systems Fundamental**  
**Class – Software Development**  
**Lecturer – Mr. Karl O`Connell**  
**Assessment 2 Part 1**  
**Date - 12/03/2018**

---

## Question 1

*Show your understanding of directories (terminal view) and folders (desktop GUI view). Note: As with all QUESTIONS, do not forget to record your progress by copying your progress to your report. Save regularly. Please ask your lab lecturer if you have any questions or require assistance.*

- Open a terminal



```
cadmin@cadmin-virtual-machine ~/OperatingSystem/Assignment2 $ ls
A2Part1.pdf
cadmin@cadmin-virtual-machine ~/OperatingSystem/Assignment2 $ ls -l
total 624
-rw-rw-r-- 1 cadmin cadmin 637067 Mar  5 20:27 A2Part1.pdf
cadmin@cadmin-virtual-machine ~/OperatingSystem/Assignment2 $ pwd
/home/cadmin/OperatingSystem/Assignment2
cadmin@cadmin-virtual-machine ~/OperatingSystem/Assignment2 $
```

- Use the `ls`, `ls -l`, and `pwd` command every so often where appropriate.

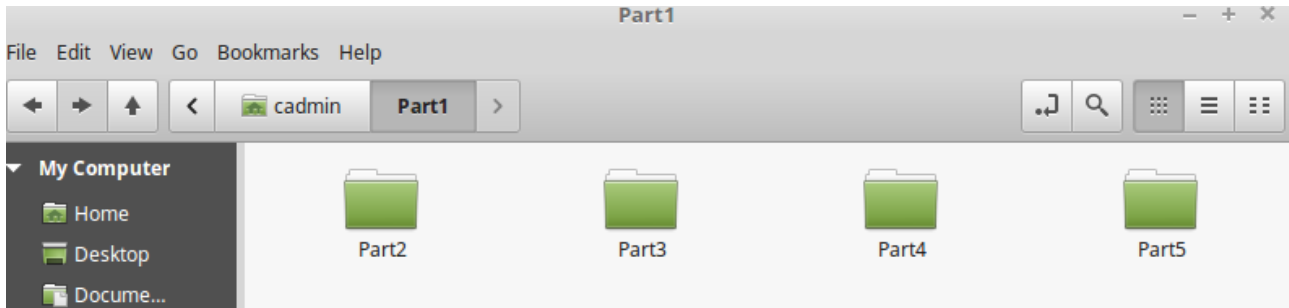
- Make sure you are in the home user directory, by typing `cd ~`, and then check with `pwd`.

- Make the directory called `Part1` by typing `mkdir Part1`. Next, change to this directory by typing `cd Part1`.

- Make 4 directories, choose your own 4 names. Display the contents of this directory `Part1`.

```
cadmin@cadmin-virtual-machine ~ $ cd ~
cadmin@cadmin-virtual-machine ~ $ pwd
/home/cadmin
cadmin@cadmin-virtual-machine ~ $ mkdir Part1
cadmin@cadmin-virtual-machine ~ $ cd Part1
cadmin@cadmin-virtual-machine ~/Part1 $ mkdir Part2 Part3 Part4 Part5
cadmin@cadmin-virtual-machine ~/Part1 $ ls
Part2 Part3 Part4 Part5
cadmin@cadmin-virtual-machine ~/Part1 $
```

-Open the desktop graphical File Manager. Change to the Part 1 folder to show its contents.



## Question 2

The root directory contains many directories. You must describe a selection of directories, and show examples where possible.

- Open a terminal
- Change to the root directory of the file system. Type `cd /` and then type `pwd`. Type `ls`.

```
cadmin@cadmin-virtual-machine ~ $ pwd
/home/cadmin
cadmin@cadmin-virtual-machine ~ $ ls -l
total 180
drwxr-xr-x 2 cadmin cadmin 4096 Mar  5 20:42 Desktop
drwxr-xr-x 2 cadmin cadmin 4096 Sep 16 21:08 Documents
drwxr-xr-x 2 cadmin cadmin 4096 Mar  5 20:27 Downloads
-rw-rw-r-- 1 cadmin cadmin 131870 Mar  5 21:00 MohammedAlomA2Part1.odt
drwxr-xr-x 2 cadmin cadmin 4096 Sep 16 21:08 Music
drwxrwxr-x 3 cadmin cadmin 4096 Mar  5 20:38 OperatingSystem
drwxrwxr-x 6 cadmin cadmin 4096 Mar  5 20:55 Part1
drwxr-xr-x 4 cadmin cadmin 4096 Mar  5 21:01 Pictures
drwxr-xr-x 2 cadmin cadmin 4096 Sep 16 21:08 Public
drwxrwxr-x 9 cadmin cadmin 4096 Sep 16 21:26 pycharm-community-2017.2.3
drwxr-xr-x 2 cadmin cadmin 4096 Sep 16 21:08 Templates
drwxr-xr-x 2 cadmin cadmin 4096 Sep 16 21:08 Videos
drwxrwxr-x 8 cadmin cadmin 4096 Sep 16 22:07 WebStorm-172.4155.35
cadmin@cadmin-virtual-machine ~ $
```

-By changing back and forth between the directories, listing files...., describe each of the following directories (a short clear description [around 6 lines for each directory] with example (s) where possible is required): `/bin`

## Example of `/bin` directory screenshot

```

cadmin@cadmin-virtual-machine /bin $ ls -l
total 16280
-rwxr-xr-x 1 root root 6456 Jul 4 2017 archdetect
-rwxr-xr-x 1 root root 1037440 May 26 2017 bash
-rwxr-xr-x 1 root root 520992 Jun 16 2017 btrfs
-rwxr-xr-x 1 root root 249464 Jun 16 2017 btrfs-calc-size
lrwxrwxrwx 1 root root 5 Jun 16 2017 btrfsck -> btrfs
-rwxr-xr-x 1 root root 278376 Jun 16 2017 btrfs-convert
-rwxr-xr-x 1 root root 249464 Jun 16 2017 btrfs-debug-tree
-rwxr-xr-x 1 root root 245368 Jun 16 2017 btrfs-find-root
-rwxr-xr-x 1 root root 270136 Jun 16 2017 btrfs-image
-rwxr-xr-x 1 root root 249464 Jun 16 2017 btrfs-map-logical
-rwxr-xr-x 1 root root 245368 Jun 16 2017 btrfs-select-super
-rwxr-xr-x 1 root root 253816 Jun 16 2017 btrfs-show-super
-rwxr-xr-x 1 root root 249464 Jun 16 2017 btrfstune
-rwxr-xr-x 1 root root 245368 Jun 16 2017 btrfs-zero-log
-rwxr-xr-x 1 root root 31288 May 20 2015 bunzip2
-rwxr-xr-x 1 root root 1964536 Aug 19 2015 busybox
-rwxr-xr-x 1 root root 31288 May 20 2015 bzipcat
lrwxrwxrwx 1 root root 6 Sep 16 21:02 bzipcmp -> bzdiff
-rwxr-xr-x 1 root root 2140 May 20 2015 bzdiff
lrwxrwxrwx 1 root root 6 Sep 16 21:02 bzegrep -> bzgrep
-rwxr-xr-x 1 root root 4877 May 20 2015 bzeze

```

## Short description of bin directory

The usage of bin directory is which it contains many commands, scripts and more number of executable files and it is mainly for executing the above in order to accomplish a task.

/bin — Essential command binaries that need to be available in single user mode; for all users, e.g., cat, ls, cp.

bin directory contains executable files that we have in our system..its like program files if we compare to MS windows .every user individually will have a bin directory which has his own executables...and there is also/sbin directory which will have system level programs.

## Example of /etc directory screenshot

```

cadmin@cadmin-virtual-machine /etc $ ls
acpi                                hostname
adduser.conf                       hosts
adjtime                           hosts.allow
alternatives                       hosts.deny
anacrontab                         hp
apg.conf                          ifplugd
apm                                iftab
apparmor                           ImageMagick-6
apparmor.d                        init
appport                           init.d
apt                                initramfs-tools
at-spi2                            inputrc
avahi                              insserv
bash.bashrc                       insserv.conf
bash_completion                   insserv.conf.d
bash_completion.d                 inxi.conf
bindresvport.blacklist            iproute2
binfmt.d                          issue
bluetooth                         issue.net
brlapi.key                        kbd
brltty                           kernel
brltty.conf                      kernel-img.conf
ca-certificates                  kerneloops.conf
pnm2ppa.conf
polkit-1
ppp
printcap
profile
profile.d
protocols
pulse
purple
python
python2.7
python3
python3.5
rc0.d
rc1.d
rc2.d
rc3.d
rc4.d
rc5.d
rc6.d
rc.local
rcS.d
request-key.conf

```

## Short description of etc/passwd directory

**/etc/passwd** - file stores essential information, which is required during login i.e. user account information. /etc/passwd is a text file, which contains a list of the system's accounts, giving for each account some useful information like user ID, group ID, home directory, shell, etc. It should have general read permission as many utilities like ls use it to map user IDs to user names, but write access only for the superuser/root account.

## Short description of etc/shadow directory

**/etc/shadow** - /etc/shadow is an encrypted file that holds user passwords. Beside containing encrypted passwords, an entry in this file also contains ageing and expiration information of password. An entry in this file can be classified into following information:

- Login name
- The corresponding Encrypted password
- Number of days since 1st Jan 1970, that password was last changed
- Number of days before password may be changed
- Number of days after which password has to be changed
- Number of days before password expiry warning starts popping up
- Number of days after password expires that account is disabled
- Number of days since 1st Jan 1970, that account is disabled
- Reserved field for further use.

## An example of to check if this root protected or not

```
mohammed@mohammed-virtual-machine:/$ cat /etc/shadow
cat: /etc/shadow: Permission denied
mohammed@mohammed-virtual-machine:/$ █
```

Permission has been denied. Only root user can access to the shadow file.

## Short description of etc/group directory

**/etc/group** - Similar to /etc/passwd, but describes groups instead of users. /etc/group is a text file which defines the groups to which users belong under Linux and UNIX operating system. Under Unix / Linux multiple users can be categorized into groups. Unix file system permissions are organized into three classes, user, group, and others. The use of groups allows additional abilities to be delegated in an organized fashion, such as access to disks, printers, and other peripherals. This method, amongst others, also enables the Superuser to delegate some administrative tasks to normal users.

## Screen shot examples of etc/group

```
mohammed@mohammed-virtual-machine:~$ more /etc/group
root:x:0:
daemon:x:1:
bin:x:2:
sys:x:3:
adm:x:4:syslog,mohammed
tty:x:5:
```

```
mohammed@mohammed-virtual-machine:~$ groups
mohammed adm cdrom sudo dip plugdev lpadmin sambashare
mohammed@mohammed-virtual-machine:~$ groups mohammed
mohammed : mohammed adm cdrom sudo dip plugdev lpadmin sambashare
mohammed@mohammed-virtual-machine:~$ id -g
1000
mohammed@mohammed-virtual-machine:~$ id -g mohammed
1000
mohammed@mohammed-virtual-machine:~$ id -G
1000 4 24 27 30 46 113 128
mohammed@mohammed-virtual-machine:~$ id -G mohammed
1000 4 24 27 30 46 113 128
mohammed@mohammed-virtual-machine:~$ id -Gn mohammed
mohammed adm cdrom sudo dip plugdev lpadmin sambashare
mohammed@mohammed-virtual-machine:~$
```

## Short description of usr directory

**/usr** usually contains by far the largest share of data on a system. Hence, this is one of the most important directories in the system as it contains all the user binaries, their documentation, libraries, header files, etc.... X and its supporting libraries can be found here. User programs like telnet, ftp, etc.... are also placed here. In the original Unix implementations, /usr was where the home directories of the users were placed (that is to say, /usr/someone was then the directory now known as /home/someone).

```
mohammed@mohammed-virtual-machine:/usr$ cd /usr
mohammed@mohammed-virtual-machine:/usr$ ls -l
total 128
drwxr-xr-x  2 root root 61440 Mar  4 20:05 bin
drwxr-xr-x  2 root root  4096 Oct 23 23:59 etc
drwxr-xr-x  2 root root  4096 Feb 27 20:55 games
drwxr-xr-x 38 root root 16384 Feb 28 10:57 include
drwxr-xr-x 156 root root  4096 Feb 28 11:22 lib
drwxr-xr-x 10 root root  4096 Aug  1 2017 local
drwxr-xr-x  3 root root  4096 Aug  1 2017 locale
drwxr-xr-x  2 root root 12288 Feb 28 10:57 sbin
drwxr-xr-x 323 root root 12288 Mar  4 20:05 share
drwxr-xr-x  6 root root  4096 Feb 27 20:53 src
mohammed@mohammed-virtual-machine:/usr$
```



## Short description of home directory

Linux is a multi-user environment so each user is also assigned a specific directory that is accessible only to them and the system administrator. These are the user home directories, which can be found under '/home/\$USER' (~/.). It is your playground: everything is at your command, you can write files, delete them, install programs, etc.... Your home directory contains your personal configuration files, the so-called dot files (their name is preceded by a dot).

```
mohammed@mohammed-virtual-machine:/usr$ cd /home
mohammed@mohammed-virtual-machine:/home$ ls
mohammed  user1  user2  user3  user4
mohammed@mohammed-virtual-machine:/home$ ls -l
total 20
drwxrwxr-x 38 mohammed mohammed 4096 Mar 11 12:23 mohammed
drwxr-xr-x 16 user1      user1    4096 Mar  7 00:25 user1
drwxr-xr-x  2 user2      user2    4096 Feb 27 21:23 user2
drwxr-xr-x  2 user3      user3    4096 Feb 27 21:23 user3
drwxr-xr-x  2 user4      user4    4096 Mar  4 23:44 user4
mohammed@mohammed-virtual-machine:/home$
```

## Question 3

Show your understanding of terminal editors and graphical editors.

- Open the terminal.
- Go to the Part1 directory.
- Create 3 sample short files of your own choice, however, you must use 1) the nano editor, 2) the vi editor and 3) the graphical editor program gedit.
- Ask your lecturer if you need help.
- Give around an 8 line discussion for each of these editors, discussing how to create, add lines, save, ...
- Do not forget to use the pwd, ls, ls -l commands where appropriate

## Creating the nano file

```
mohammed@mohammed-virtual-machine: ~/OperatingSystem/Assignment2/Part1 - □
File Edit View Search Terminal Help
GNU nano 2.5.3 File: nanoFile.txt Modified
This is nano based text file.
Create and Open Files
There are different ways to start nano depending on what type of file you wish you edit.
Open Text Files
To edit a basic text file end with .txt, you can use nano /path/to/file.txt to open or create a text file in a specific location
```

## Nano editor - how to create add lines and save file

There are different ways to start nano depending on what type of file you wish to edit. Nano can be used to create blank files that are then opened to be edited. To create a new file, issue a command similar to the following:  
mohammed@mohammed-virtual-machine:~/OperatingSystem/Assignment2/Part1\$ nano nanoFile.txt

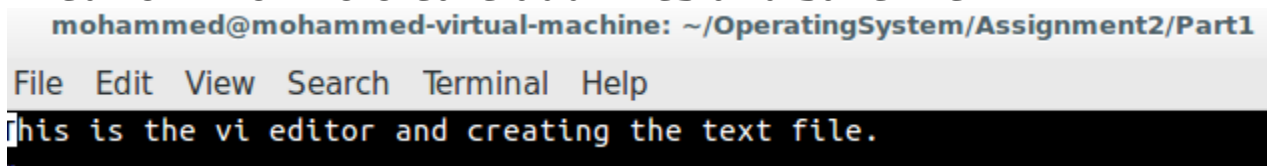
Ctrl + X will quit the editor and you will be asked if you want to save your changes. If you do, press Y for Yes.

Ctrl + O should also work, since that means to save the file, but you won't be asked "*Save modifier buffer ?* " because you already told nano to save.

## Cut and Paste Lines of Text in the nano editor

To cut a line of text, use ^K. To paste it back, simply move the cursor where you want the text to be placed and use ^U. If you would like to cut multiple lines, use a series of ^K commands until all lines you wish to cut have been removed. When you go to paste them back with ^U, the lines will all be pasted at once.

## vi editor - how to create add lines and save file



```
mohammed@mohammed-virtual-machine: ~/OperatingSystem/Assignment2/Part1
File Edit View Search Terminal Help
this is the vi editor and creating the text file.
```

**vi is short for** visual editor. To create a new file called 'example1': vi example1

vi operates in two basic modes

- command mode where everything that is typed is taken to be a command to the editor to do something. All commands are single letter commands. This is the mode in which you always start.

- insert mode where everything that is typed is included as text in the file. Before you insert any text you must enter one of the insertion modes.

ZZ - Save the file and exit the editor.

I - enter insert mode to begin entering text at the cursor position.

<ESC> - enter command mode

- x - delete the current character
- dd - delete the current line
- :q! - Quit the editor without saving changes
- :w - filename Save the file as 'filename'

## gedit editor - how to create add lines and save file

```
mohammed@mohammed-virtual-machine:~/OperatingSystem/Assignment2/Part1$ gedit geditFile.txt &  
[5] 16136  
mohammed@mohammed-virtual-machine:~/OperatingSystem/Assignment2/Part1$ █
```

with gedit editor we can do create, save and adding text from visual menu bar File and Edit.

## Question 4

*Examine the operators of pipe, redirection and append.*

### Pipe |

A pipe is designated in commands by the vertical bar character, which is located on the same key as the backslash on U.S. keyboards. The general syntax for pipes is:

```
command_1 | command_2 [| command_3 . . . ]
```

A very simple example of the benefits of piping is provided by the `dmesg` command, which repeats the startup messages that scroll through the console (i.e., the all-text, full-screen display) while Linux is booting (i.e., starting up). `dmesg` by itself produces far too many lines of output to fit into a single screen; thus, its output scrolls down the screen at high speed and only the final screenful of messages is easily readable. However, by piping the output of `dmesg` to the filter `less`, the startup messages can conveniently be viewed one screenful at a time, i.e.,

```
dmesg | less
```

Likewise, the output of the `ls` command (which is used to list the contents of a directory) is commonly piped to the `less` (or `more`) command to make the output easier to read, i.e.,

```
ls -al | less
```

or

```
ls -al | more
```



## Redirection >

*Redirection* is the switching of a *standard stream* of data so that it comes from a source other than its default source or so that it goes to some destination other than its default destination.

Redirection is similar to pipes except using files rather than another program. The standard output for a program is the screen. Using the > (greater than) symbol the output of a program can be sent to a file. Here is a directory listing of /dev again but this time redirected to a file called listing.txt

There won't be anything displayed on the terminal as everything was sent to the file. You can take a look at the file using the cat command (which can be piped into more) or for convenience you can just use the more command on its own:

```
mohammed@mohammed-virtual-machine:~$ ls -la > listing.txt
mohammed@mohammed-virtual-machine:~$ more listing.txt
total 308
drwxrwxr-x 38 mohammed mohammed 4096 Mar 12 00:18 .
drwxr-xr-x 7 root root 4096 Mar 4 23:41 ..
drwx----- 3 mohammed mohammed 4096 Nov 9 16:21 .adobe
drwxrwxr-x 4 mohammed mohammed 4096 Feb 27 21:19 alom
-rw-rw-r-- 1 mohammed mohammed 86 Feb 27 21:05 .asoundrc
drwxrwxr-x 9 mohammed mohammed 4096 Oct 22 10:17 .atom
```

If listing.txt had already existed, it will be overwritten. But you can append to an existing file using >>

## Append >>

To append text to a file you use >>

```
mohammed@mohammed-virtual-machine:~$ cd OperatingSystem/Assignment2
mohammed@mohammed-virtual-machine:~/OperatingSystem/Assignment2$ cd Part1
mohammed@mohammed-virtual-machine:~/OperatingSystem/Assignment2/Part1$ la
geditFile.txt nanoFile.txt viFile.txt .viFile.txt.swo .viFile.txt.swp
mohammed@mohammed-virtual-machine:~/OperatingSystem/Assignment2/Part1$ echo 'For
desktop usage I prefer Apple OS unix operating system.' >bar.txt
mohammed@mohammed-virtual-machine:~/OperatingSystem/Assignment2/Part1$
```

```
mohammed@mohammed-virtual-machine:~/OperatingSystem/Assignment2/Part1$ cat nanoFile.txt cat bar.txt
This is a test.
I like Linux Operating System.cat: cat: No such file or directory
For desktop usage I prefer Apple OS unix operating system.
mohammed@mohammed-virtual-machine:~/OperatingSystem/Assignment2/Part1$
```

Display both files on screen

**To append a contains of bar.txt to to nanoFile.txt, enter:**

```
cat bar.txt >> foo.txt
```

```
cat foo.txt
```

```
mohammed@mohammed-virtual-machine:~/OperatingSystem/Assignment2/Part1$ cat bar.txt >> nanoFile.txt
mohammed@mohammed-virtual-machine:~/OperatingSystem/Assignment2/Part1$ cat nanoFile.txt
This is a test.
I like Linux Operating System.For desktop usage I prefer Apple OS unix operating system.
For desktop usage I prefer Apple OS unix operating system.
mohammed@mohammed-virtual-machine:~/OperatingSystem/Assignment2/Part1$
```