Exsercise 2: Exploring Linux Command-Line tools (Cont.)

I. Preparing lab evironment

1. Login to the CentOS system and Install vim package with the following command

   $ sudo yum install vim

   Supply your password

   On “Is this ok [y/d/N]:” input y and press enter

II. Vim editor

1. Ensure that you are in your home directory. Create a file in your home directory named vitest.
2. When you open a **vim** file, you are automatically placed in command mode. Press the **i** key (insert) to switch to input (text) mode. You can also press the **a** key (append). Use of **i** or **a** simply determines if typing starts before or after the cursor. There is no indication to tell you that you are in input mode.

   Switch from input mode to command mode by pressing the ESC key. Press ESC a second time. Notice that if you press ESC twice, you will get a "beep" from the terminal (some ASCII terminals do not beep). The beep indicates that you are in command mode already. Now press **i** again to put you back in input mode. Continue to the next step.

3. Input the following text EXACTLY as it is presented line by line. Then key in the alphabet, one character per line. Following will show a-d but continue on through z. Adding the alphabet is an easy way to fill a couple of screens of information needed for later use.

```
This is a training session about the usage of the vim editor.
We need some more lines to learn the most common commands
of the editor. We are now in the entry mode and we will
switch right after this to the command mode.
a
b
c
d
.
.
.
.
z
```

4. Return to command mode. Write and quit the file. Notice that as soon as you press the : (colon), it appears below the last line of your input area. Once the buffer is empty and the file is closed, you will see a message giving the number of lines and characters in the file.

Cursor Movement Keys

5. Open **vimtest** using **vim**.
6. Using both the arrow keys and the **h, j, k, l** keys, practice moving the cursor down one line, up one line, right a couple characters, and back a couple characters.
7. You may not want to cursor one character or one line at a time throughout an entire file. Practice using cursor movement keys to work around by page or by line. Using the cursor movement keys from instruction 6, position your cursor at the first line of the file. While in command mode, do the following:

    i. Move cursor to last line in the file
    ii. Move cursor to first line in the file
    iii. Move cursor to line 4 of the file
    iv. Move cursor to end of line
    v. Move cursor to beginning of line

8. Move your cursor to the top of the file. Search for the word **entry**. Your cursor should be on the **e**. Switch to input mode and add the word **text**. Don't forget the space after the word.
9. Move the cursor to the space after the word **mode** on the same line. Insert a comma. Remember, you are still in input mode.
10. Enter command mode. Position the cursor anywhere on the line beginning with "some more lines." Insert a blank line to form two paragraphs.
11. Opening up a blank line as in the previous step, automatically puts you in input mode; therefore, return to command mode. Now save the changes you have made so far, but DON'T exit the editor.
12. While still in command mode, remove the alphabetic characters **c,e,g** but leave the blank lines in their place; in other words, don't delete the entire line, just the character. Then go back and remove the blank lines. This will give you practice using two of the delete functions.
13. You just decided you really don't want to save the changes to the alphabetic characters. Quit the editing session without saving the changes made since the last save.
14. Edit **vimtest** one more time. First, copy the first paragraph one line at a time to the end of the file. When that is complete, copy the second paragraph all at once to the end of the file.
15. You just decided that the lines you just added to the end of file don't look right. Delete them all with one command.
16. Now, before you do anything else with this file, you decide you need to imbed the current date and time as the first line of the file. Do this without leaving the vim editor.
17. Using vim create the following files
    - ***string.txt*** with the following content:

Alpha
Tango
Bravo
Echo
Foxtrot

- ***number.txt*** with the following content:

135
178
238
982
911

Process text file utilities

18. Using cat command to view the content of string.txt file.
19. Display the content of string.txt and number.txt files to the screen
20. Using od to display the content of string.txt in octal
21. Split the vimtest file to files with maximum of 4 line each
22. Sort the content of string.txt in alphabetical order and sort the numbers in number.txt from the smallest to the biggest.
23. Display the string.txt content to screen with line number, not including blank lines.
24. Use less and more to view the content of /var/log/messages, move next and back accross pages. You need to use sudo to gain the superuser previleges.
25. View the content of /var/log/messages again but using the head and tail.
26. Display the content of /var/log/messages in real-time with tail. Use Ctrl+C to escape.
27. Count the number of lines in /etc/passwd file, is that the number of users created on server?
28. Display only the username from the /etc/passwd
29. Create the hash using md5 and sha512 algorithm for the number.txt file. Create a new file with the same content of number.txt, name it number1.txt. Hashing the number1.txt and compare with the hash value of number.txt.

Exsercise Instructions

I. Preparing lab environment

1. Log in to the system

Log int to the CentOS system with the user name and password provided: student/lpic1@123

2. Install the vim package

[student@centos7~]$ **sudo yum install vim**

[sudo] password for student:

….

Is this ok [y/d/N]: y ***← input y and press enter***

II. Vim editor

1. Ensure that you are in your home directory. Create a file in your home directory named **vimtest**.

   - **$cd**
   - **$pwd**
   - **$ vim vimtest**

2. When you open a **vim** file, you are automatically placed in command mode. Press the **i** key (insert) to switch to input (text) mode. You can also press the **a** key (append). Use of **i** or **a** simply determines if typing starts before or after the cursor. There is no indication to tell you that you are in input mode.

   Switch from input mode to command mode by pressing the ESC key. Press ESC a second time. Notice that if you press ESC twice, you will get a "beep" from the terminal (some ASCII terminals do not beep). The beep indicates that you are in command mode already. Now press i again to put you back in input mode. Continue to the next step.

   - **i**
   - **ESC**
   - **ESC again (did you hear a beep) •i**

3. Input the following text EXACTLY as it is presented line by line. Then key in the alphabet, one character per line. Following will show a-d but continue on through z. Adding the alphabet is an easy way to fill a couple of screens of information needed for later use.

```
This is a training session about the usage of the vim
editor. We need some more lines to learn the most common
commands of the editor. We are now in the entry mode and
we will switch right after this to the command mode.
a
b
c
d
.
.
.
.
z
```

4. Return to command mode. Write and quit the file. Notice that as soon as you press the : (colon), it appears below the last line of your input area. Once the buffer is empty and the file is closed, you will see a message giving the number of lines and characters in the file.

    - **ESC** (puts you in command mode)

    - **:wq (<shift-ZZ>** is another way to write and quit)

Cursor Movement Keys

5. Open **vimtest** using **vim**. Notice the bottom line of the file indicates the name of the file and number of characters.

    - **$ vim vimtest**

6. Using both the arrow keys and the **h, j, k, l** keys, practice moving the cursor down one line, up one line, right a couple characters, and back a couple characters.
    - **j** (down a line)
    - **k** (upaline)
    - **l** (right a character)
    - **h** (left a character)
    - **Repeat using the appropriate arrow keys**

7. You may not want to cursor one character or one line at a time throughout an entire file. Practice using cursor movement keys to work around by page or by line. Using the cursor movement keys from instruction 6, position your cursor at the first line of the file. While in command mode, do the following:

    i. Move cursor to last line in the file
    ii. Move cursor to first line in the file
    iii. Move cursor to line 4 of the file
    iv. Move cursor to end of line
    v. viii.Move cursor to beginning of line

- **<shift-G>**
- **1<shift-G> or :1 Enter**
- **4<shift-G> or :4 Enter**
- **$**
- **0 (this is a zero)**

8. Move your cursor to the top of the file. Search for the word **entry**. Your cursor should be on the **e**. Switch to input mode and add the word **text**. Don't forget the space after the word.

    - **1<shift-G> or :1 • /entry**
    - **i**
    - **text**

9. Move the cursor to the space after the word **mode** on the same line. Insert a comma. Remember, you are still in input mode.

    - **ESC**
    - **Position the cursor to the space after mode**
    - **i ,** (comma)

10. Enter command mode. Position the cursor anywhere on the line beginning with "some more lines" Insert a blank line to form two paragraphs.

    - **ESC**
    - **Position cursor on line starting "some more lines"**
    - **o** (lowercase o opens the line after the cursor)

11. Opening up a blank line as in the previous step, automatically puts you in input mode; therefore, return to command mode. Now save the changes you have made so far, but DON'T exit the editor.

    - **ESC**
    - **:w**

12. While still in command mode, remove the alphabetic characters **c,e,g** but leave the blank lines in their place; in other words, don't delete the entire line, just the character. Then go back and remove the blank lines. This will give you practice using two of the delete functions.

    - **Position cursor on c; Press x**
    - **Position cursor on e; Press x**
    - **Position cursor on g; Press x**
    - **Position cursor on each of the blank lines; Press dd**

13. You just decided you really don't want to save the changes to the alphabetic characters. Quit the editing session without saving the changes made since the last save.

- **:q!**

14. Edit **vimtest** one more time. First, copy the first paragraph one line at a time to the end of the file. When that is complete, copy the second paragraph all at once to the end of the file.

    - **$ vim vimtest**
    - **Position cursor on line one; Press yy**
    - **<shift-G>; Press p**
    - **2<shift-G>; Press yy**
    - **<shift-G>; Press p**
    - **3<shift-G>; Press yy**
    - **<shift-G>; Press p**
    - **4<shift-G>; Press 3yy**
    - **<shift-G>; Press p**

15. You just decided that the lines you just added to the end of file don't look right. Delete them all with one command.

    - **Position the cursor on the first copied line at the bottom of the file to be deleted**
    - **Count the number of lines to delete**
    - **5dd** (your number may be different if you moved the blank line as well)

16. Now, before you do anything else with this file, you decide you need to imbed the current date and time as the first line of the file. Do this without leaving the **vim** editor.

    - **:r!date**

17. Create string.txt and number.txt
    $ vim string.txt
    - Type i to change to input mode
    - Type in the content
    - Esc
    - :wq!

    $ vim number.txt
    - Type i to change to input mode
    - Type in the content
    - Esc
    - :wq!

18. Viewing the string.txt file with cat
    $ cat string.txt

19. Display the string.txt and number.txt to the screen
    $ cat string.txt number.txt

20. Using od to display the string.txt content in octal
    $ od string.txt

21. Split the vimtest file to files with maximum 4 line each
    $ split -l 4 vimtest

22. Sorting the string.txt and number.txt content
    $ sort string.txt
    $ sort -n number.txt

23. Display the string.txt content to screen with the line number, not including blank lines
    $ nl string.txt

24. Use less and more to view the content of /var/log/messages
    $ sudo more /var/log/messages
    $ sudo less /var/log/messages

25. View the /var/log/messages with head and tail
    $ sudo head /var/log/messages
    $ sudo tail /var/log/messages

26. View the /var/log/messages in real-time with tail
    $ sudo tail -f /var/log/messages

27. Count the line of /etc/passwd
    $ wc -l /etc/passwd

28. Display only the user name from /etc/passwd
    $ cut -d ":" -f 1 /etc/passwd

29. Hashing the number.txt using the md5 and sha512 algorithm. Create a new file with the same content of number.txt, name it number1.txt. Hashing the number1.txt and compare with the hash value of number.txt
    $ md5sum number.txt
    $ sha512sum number.txt

    $ vim number1.txt
    - :r number.txt
    - :wq!

    $ md5sum number1.txt
    <hash value 1>
    $ md5sum number.txt
    <hash value 2>

    Compare the above hash values