M06.01 - Instructions - Text Processing Tools

Assignment Objectives:

On completing this lab, you should be able to:

- Demonstrate filtering text using the grep and regular expressions
- Demonstrate using other text processing tools uniq, sort, wc, head, tail, cut and cat.

Part 1

Instructions:

The Linux command line is incredibly powerful and efficient at processing text in files. This lab will introduce you to several command line tools used to process and manipulate text in files. You will also be required to complete tasks previously covered in prior sections and in class. Perform the task being requested on your vSphere virtual machine. Provide the requested answer or screenshot as indicated in the task.

Task

Task 1

Read the handout on accessing the Virtual Lab in Canvas to access your virtual machine. Log in to the **sysadmin** account with the password **netlab123** and open the terminal program.

Task 2

Create a new directory called TextFiles and change your working directory to this new directory.

Task 3

Create three text files named **avengers.txt**, **candy.txt** and **numbers.txt** with the following text. There are intentional duplicates in these files. Use vim to create the files.

avengers.txt file

```
iron man
hulk
vision
black widow
hulk
hawkeye
captain america
```

Scarlet witch
spider-man
black panther
vision
vision

candy.txt file

M & Ms
milky way
snickers
twix
kit-kat
Reese’s peanut buttercups
twix
kit-kat
sour patch kids
butterfinger

numbers.txt
1
51
52
2
32
31
3
16
4
24
25
5
9
10
11
12
13
20
21

Task 4 - Sort Command

To sort the text for the avengers.txt files alphabetically, use the following command:

```
sort  avengers.txt
```

Task 5

Demonstrate using the sort command with the candy.txt file.

Capture a screenshot showing you successfully completed this task.

Add your screenshot for this task **on the answer sheet** as **Screenshot 1**.

Task 6

You can sort both the avengers.txt and candy.txt at the same time with the following command.

```
sort  avengers.txt  candy.txt
```

Task 7

The sort command has several useful options. For example it can sort in reverse order using the -r options.

```
sort  -r  avengers.txt
```

Sort can also randomize a text file with capital “r” option -R.

```
sort -R  candy.txt
```

Execute the randomize option 2 or 3 times. Each time the list will be in a different order.

Sort can remove duplicates with the -u (unique) option.

```
sort  -u  avengers.txt
```

Notice all the duplicate entries are removed.

Sort does not change the text files in any way. It only displays the text in the terminal. However using the -o (output) option creates a new sorted file. For example, we can sort the avengers.txt file and remove the duplicates and create a new file called avengers_nodups.txt with the following command.

```
sort  -u  avengers.txt  -o  avengers_nodups.txt
```

Use the “ls” command to verify that a new avengers_nodups.txt file was created and use the “cat” command to display the contents of the new file.

Task 8

Attempt to sort the numbers.txt file with the following command:

```
sort numbers.txt
```

Review the output of the command. Are the numbers in the file sorted correctly? Describe how the numbers are sorted.

Submit your answer for this task **on the answer sheet** as **Answer 1**.

Task 9

Use the -n (numbers) to sort the numbers.txt file correctly.

```
sort  -n  numbers.txt
```

Task 10

Use the sort command to reverse sort the candy.txt file and create a new file called candy_rev.txt that contains the reverse sorted candy list.

Capture a screenshot showing you successfully completed this task.

Add your screenshot for this task **on the answer sheet** as **Screenshot 2**.

Task 11

The sort command can also sort files with multiple columns.

Use vim to create the following file called employees.txt. This file lists the employee's name, age and salary. Make sure to only have 1 space between each item.

employees.txt
```
Charlie 25 49000
Bob 28 52000
```

Dana 31 47000
Alice 34 45000

We can now sort the employees by age. Age is the second column, we can do this with the following command:

```
sort  -k2  employees.txt
```

We can sort the employees by pay which is the third column.

```
sort  -k3  employees.txt
```

Task 12 - Uniq command

The uniq command removes duplicate text within a file, however the duplicates text needs to be next to each other. Execute the command:

```
uniq  candy.txt
```

Notice that the duplicates remain in the file.

Execute the command on the reverse sort candy_rev.txt file.

```
uniq  candy_rev.txt
```

Notice that the duplicates have been removed.

Task 13

The uniq command has several options that makes it useful. For example the -c (count) option will count the number of duplicates.

```
uniq  -c  candy_rev.txt
```

uniq can also only display the duplicate lines using the capital -D (duplicates) option.

```
uniq  -D  candy_rev.txt
```

It can combine only the duplicates with the lowercase -d option.

```
uniq  -d  candy_rev.txt
```

Task 14

Execute the following command to sort the avengers.txt and create a new file called avengers_sort.txt.

```
sort  avengers.txt  -o  avengers_sort.txt
```

Use the uniq command to count the number of duplicates in the new avengers_sort.txt file.

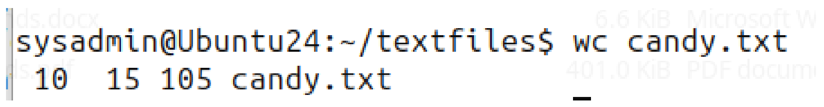
Capture a screenshot showing you successfully completed this task.

Add your screenshot for this task **on the answer sheet** as **Screenshot 3**.

Task 15 - wc command

wc is the word count command. It counts the characters, lines or words of a file. Execute the following command:

```
wc  candy.txt
```



```
sysadmin@Ubuntu24:~/textfiles$ wc candy.txt
 10  15 105 candy.txt
```

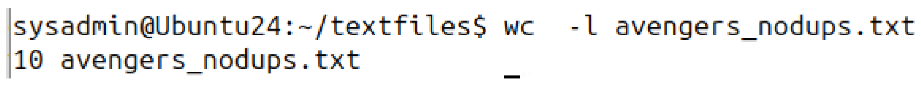
The first number represents the number of lines in the file. In the example above there are 10 lines in the candy.txt file. The next number represent the number of words in the file, in this case 15 and the last number, 105 is the number of characters.

The wc command has 3 useful argument options which isolates each item it counts.

-l   displays only the lines count
-w displays on the word count
-c   displays only the character count

The option if find the most useful is the -l option to count the number of lines. For example, we can count the number of avengers in the avengers_nodups.txt with the following command:

```
wc  -l  avengers_nodups.txt
```



```
sysadmin@Ubuntu24:~/textfiles$ wc  -l avengers_nodups.txt
10 avengers_nodups.txt
```

This tells us that there are 10 avengers without ever opening the file. One superhero on each line. This may not be too impressive on a text file with superheroes, but it is extremely helpful when you need to quickly know how many times someone accessed a system for a log file that is extremely large or need to know how many records are in a flat data record.

Task 16 - cat command

The cat command is short for concatenate, which means to connect or link together. When used with a single file it displays the contents of the file on the screen, such as with the command:

```
cat  avengers.txt
```

However, when used with multiples it connects or links these files together, such as this command:

```
cat  numbers.txt  avengers.txt
```

Describe what happens when you run the above command. Which text is at the top and which text is at the bottom?

Submit your answer for this task **on the answer sheet** as **Answer 2**.

Task 17

View the avengers.txt file using the cat command with the -n option.

```
cat  -n  avengers.txt
```

Describe what the -n option does?

Submit your answer for this task **on the answer sheet** as **Answer 3**.

Task 18

Use vim to create a file called pawnee.txt with the following content.

list.txt file

```
1%ron%director%Parks Dept*
2%leslie%deputy director%Parks Dept*
3%andy%shoe shiner%staff*
4%li’l sebastian%mascot%Mini Horse*
5*april*intern*Parks Dept%
6*tom*associate*Parks Dept%
7*tammy*pure evil*Library Dept%
```

8*shauna*reporter*The Pawnee Journal%

Task 19 - cut command

The cut command is used to cut columns from files separated by a delimiter. A delimiter is a character that is used to separate fields. Notice in the pawnee.txt file created in the previous task, that each field (column) is separated by either a % or a *. **NOTE:** Ignore the % and * at the end of each line. Those are there to control the output for demonstration purposes.

Let look at the first line:

```
1%ron%director%Parks Dept
```

These lines use % to separate each field. The fields are as follows:

First field = 1
Second field = ron
Third field + director
Forth field = Parks Dept

For the first 4 lines using the delimiter %, we would like to display the second field which contains the names of the employees.

The cut command requires 4 parts.

cut - the command
-d% - delimiter. each field (column) is separated by the % character.
-f2 - field. display the second (2) field.
pawnee.txt - the file

The command should look like the following:

```
cut  -d’%’  -f2  pawnee.txt
```

If we would like to view the name of the employees in lines 5-8 that are separated by a *, we would change the delimiter to *.

```
cut  -d’*’  -f2  pawnee.txt
```

If we wanted to return the job positions in the 3rd field, we would change the field to option 3.

```
cut  -d’*’  -f3  pawnee.txt
```

The cut command can return multiple fields for example the following command will return fields 1 through 3:

```
cut  -d'*'  -f1-3  pawnee.txt
```

This command will return fields 2 and 4.

```
cut  -d'*'  -f2,4  pawnee.txt
```

Task 20

The pawnee.txt file was a weirdly formatted file created to help demonstrate delimiters and the cut command. Many businesses work with data files that have millions of lines of data. Many of these files are delimited files. The most common delimited files are comma separate (csv) or tab separated (tsv) files. This just means that the delimiters are commas or tabs. An example of a record in a csv file might look like the following:

```
jared,123 Main st,402–123-4567,faculty,mcc
```

Really anything can be a delimiter. For example take a look at the /etc/passwd file using the cat command. This file contains system information about users.

```
cat  /etc/passwd
```

Notice the bottom 2 lines contain information about sysadmin and wadams.

What delimiter does the /etc/passwd file use?
What field are the users sysadmin and wadams in?

Submit your answer for this task **on the answer sheet** as **Answer 4**.

Task 21

Use the cut command to display the first field in the /etc/passwd file (i.e. the field with sysadmin and wadams).

Capture a screenshot showing you successfully completed this task.

Add your screenshot for this task **on the answer sheet** as **Screenshot 3**.

Task 22 - grep command

The grep command is a powerful Linux tool used to search for specific patterns or keywords in text files. It displays the line that matches that keyword or pattern. Grep is great for filtering through log files or other large datasets.

To search for the keyword "sysadmin" to find the sysadmin user in the /etc/passwd file use the following command:

```
grep  sysadmin  /etc/passwd
```

Notice this only returns the line that contains "sysadmin"

There are 3 parts to this grep command.

grep - the command
sysadmin - the keyword we are filtering for
/etc/passwd - the file we are searching

The following example returns any lines that contain the word "false" from the /etc/passwd file.

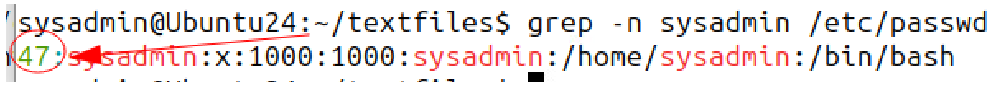
```
grep  false  /etc/passwd
```

Task 23

grep has a large number of useful options and can do so much there have been entire books written about grep. This assignment will only cover a small sampling of grep options.

Grep can display the line number of the lines it returns with the -n option. For example the following command displays the line number that "sysadmin" appears in the /etc/passwd file.

```
grep  -n sysadmin /etc/passwd
```



```
sysadmin@Ubuntu24:~/textfiles$ grep -n sysadmin /etc/passwd
47:sysadmin:x:1000:1000:sysadmin:/home/sysadmin:/bin/bash
```

The -v option does an invert match, meaning it only displays the lines that are NOT the keyword. The following command will only display those that do NOT have the keyword "nologin".

```
grep  -v nologin  /etc/passwd
```

The -c option will count all the instances of the keyword. The following command counts how many time the keyword "false" is found in the /etc/passwd file.

```
grep  -c  false  /etc/passwd
```

The -i option does case insensitive searches, meaning it ignores if the letters in the keyword are uppercase or lowercase. For example the following grep command will not find the keyword "WADAMS" because all the letters are uppercase and the file only contains a lowercase "wadams".

```
grep  WADAMS  /etc/passwd
```

However, when you add the -i option case is ignored.

```
grep -i  WADAMS  /etc/passwd
```

Task 24

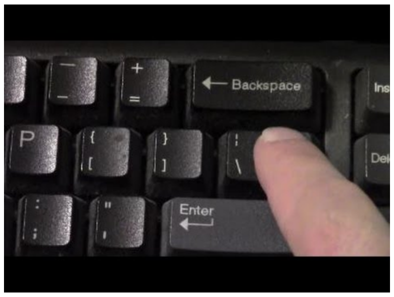
Use grep to search for the keyword "local" from the /etc/hosts file.

Capture a screenshot showing you successfully completed this task.

Add your screenshot for this task **on the answer sheet** as **Screenshot 4**.

Task 25 - regex with grep

One of the superpowers with grep is that it can use regular expressions also known as regex to create complex searches. Regular expressions (regex) are sequences of characters that define search patterns. For example, to search for two or more options from a file, use the -E option for regular expression and use a **| (pipe) symbol** to represent "or" in our search. The | (pipe) is **not** a capital "i" or lowercase "L". It is a vertical line symbol on your keyboard and is usually found above the right enter key.



To search for both hulk and hawkeye from the avengers.txt file, use the following command:

```
grep  -E  "hulk|hawkeye"  avengers.txt
```

Explanation of the command.

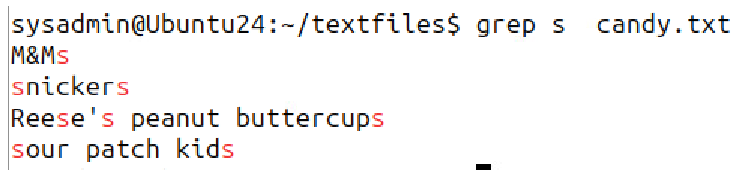
grep - the command
-E - states a regular express will be used.
“hulk|hawkeye” - This the regular expression, the | (pipe) means or, so hulk or hawkeye.
avengers.txt - the file to search

Task 26

To search for a pattern that starts at the beginning of the line, use the regular express symbol ^ found on the 6 key on the keyboard. For example, notice the following two grep commands.

The first searches for any line that contains the letter s from the candy.txt file.

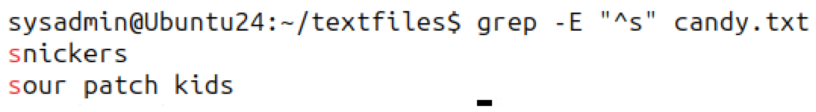
grep s candy.txt



```
sysadmin@Ubuntu24:~/textfiles$ grep s  candy.txt
M&Ms
snickers
Reese's peanut buttercups
sour patch kids
```

However we can use regular expressions to perform a more specific search for lines that start with the letter “s” using the ^ symbol.

grep -E “^s” candy.txt

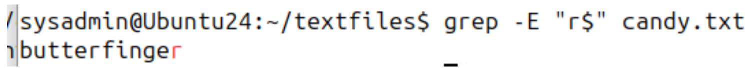


```
sysadmin@Ubuntu24:~/textfiles$ grep -E "^s" candy.txt
snickers
sour patch kids
```

The ^ represents the beginning of the line.

We can perform a similar search for a letter at the end of the line using the $ symbol. The following example matches any line that ends with the letter “r”.

grep -E “r$” candy.txt



```
sysadmin@Ubuntu24:~/textfiles$ grep -E "r$" candy.txt
butterfinger
```

The $ means the end of the line.

Task 27

You are not expected to be proficient in using regular expressions for this class, only to be aware what regex is and to provide some exposure as to what grep and other command line tools such as awk and perl with also use regex can do. There are several online tutorials, videos and books on using regular expressions if you are interested in researching more about regex.

Below is a sample regex special symbols and what they do.

.: (dot) Matches any single character.

*: Matches zero or more of the preceding character.

?: Matches zero or one of the preceding character.

+: Matches one or more of the preceding character.

^: Matches the start of a line.

$: Matches the end of a line.

[ ]: Matches any one character inside the brackets (e.g., [aeiou] matches any

Task 28 - head and tail command

When using the cut or grep command, it may require that you need to see some of the text in the file to know how to use the command. The cat command will display the entire file. If the file is large it can fill the terminal and can continue to scroll for a while until it reaches the end of the file.

The head command allows you to view a small section of the top of a file and the tail command allows you to view a small section of the bottom of a file. By default they show the top or the bottom 10 lines. To view the top 10 lines of the /etc/group file use the following command:

```
head  /etc/group
```

To view the bottom 10 lines of the same file use the following command:

```
tail  /etc/group
```

If you want to view a specific number of lines use the -n option with each command. For example:

```
head  -n 3 /etc/group
```

and

```
tail  -n  3 /etc/group
```

will show you the top and the bottom 3 lines of the /etc/group file.

The tail command is great when you only need to view the last few entries of a log file.

Task 24

Display only the top 5 lines of the avengers.txt file.

Capture a screenshot showing you successfully completed this task.

Add your screenshot for this task **on the answer sheet** as **Screenshot 5**.

Task 25

Most of these tools do one thing and do it well. That concept of design is called the Unix philosophy which Linux is modeled after. In a future assignment you will learn how to combine these basic tools so you can build your own custom tools to do whatever you need done.

Part 2 - Reflection

Task 25

Which command(s) in this assignment did you find to be the most useful. Explain why?

Submit your answer for this task **on the answer sheet** as **Answer 5**.