The grep Family

In the following we will show different instances on how to use the **Grep** command. We will be using a file with data named GrepLab

For instance, to see all the lines that have the word **Lane** you would type the following assuming you copied your file to your home directory on your server

***grep Lane /home/efelix/GrepLab*** and you will get the following results

A screenshot of a computer screen

Description automatically generated

Since I didn’t use any parameters with command it printed out all the lines with case sensitivity regardless of there is a space or not between words. This is useful as you could see in the example of **CarltonLane**, maybe there should be a space between words and now I could edit the file and correct it.

In the next example, I want to find all the lines where the first person’s lane starts with H

***grep ^H /home/efelix/GrepLab*** and you will get the following results

A black background with white text

Description automatically generated

By using **^** before the search parameter it will look for the line in this the line that has an **H** to start. This is useful as we know that in this file, we know that the first word is a name. We don’t need a street name.

In this next example, we are asked to find lines that end with at least 3 zeros

***grep 000$ /home/efelix/GrepLab*** and you will get the following results

A black screen with white text

Description automatically generated

By using the **$** it will look at the end of the line. In this case, we are looking for zip codes which makes this command convenient.

In this next example we want to avoid displaying the lines that have 408

***grep -v 408 /home/efelix/GrepLab*** and you will get the following results

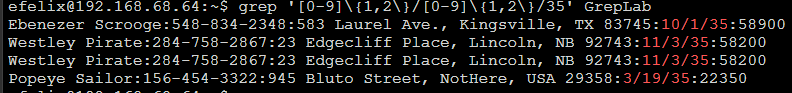
A screen shot of a computer

Description automatically generated

**-v** reverses inverts the match option and thus avoids all lines with 408. This saves time by avoiding data that you don’t need to see.

In the next example we will print all lines where birthdays are in the year 1935 (be careful of the date format! it’s MM/DD/YY)

***grep '[0-9]\{1,2\}/[0-9]\{1,2\}/35' GrepLab***



Since the dates have 1 or 2 digits to represent the day of the month, we added **{1,2\}** to produce one or the other. Otherwise, March (**3**) would not have been printed.

In this example we want to print all lines where the phone number is in an area code that starts with an 8

***grep -E '8[0-9]{2}-[0-9]{3}-[0-9]{4}' GrepLab***

A black screen with white text

Description automatically generated

The reason this command works is because we are using **^** to tell what digit we want it to begin with. That leaves us with 2 remaining digits before the dash. Then you put **{3}** and {**4}** between the remaining digits to match the phone number format. We start the grep command with **-E** to enable extended regular expressions.

In this example we want to print all lines containing an uppercase letter, followed by 4 lowercase letters, a space and one uppercase letter.

***grep -E '[A-Z][a-z]{4} [A-Z]' GrepLab***

A screen shot of a computer

Description automatically generated

As you notice this command worked because we didn’t add **{}** after the first capital letter thus telling the command that only one capital letter is required between **A-Z**. For the lower case it has to be 4 **{4}** and thus only 5 word names show up. We left a space between the search parameters so it could match what we are looking for.

In this example we will print lines where the address begins with a two or three digit number (so this would be 12 main st or 123 main street but not 1234 main street).

***grep -E '[0-9]{2,3} ' GrepLab***

A screen shot of a computer

Description automatically generated

This command should work as it is to show no more than only 3 digits by using {2,3) and then a space to continue the characters but could not get the exact results that I’m looking although I have tried the command in various forms.

In this example we will print lines where the person lives in Mass or Illinois

***Grep -E ‘IL|MA’ GrepLab***

A black screen with white text

Description automatically generated

The pipe command allows you to search multiple items.

In this example we will print lines containing the addresses that aren’t on a street (You might see St as shorthand for street)

***grep -Ev ‘Street|St’ GrepLab***

A screenshot of a computer screen

Description automatically generated

By using **v** in the parameters, we are excluding the items within the quotation marks.

Used the following websites as guides to find solutions for the problems that were assigned.

[How to Use the grep Command on Linux (howtogeek.com)](https://www.howtogeek.com/496056/how-to-use-the-grep-command-on-linux/)

[Regular expressions in grep ( regex ) with examples - nixCraft (cyberciti.biz)](https://www.cyberciti.biz/faq/grep-regular-expressions/)

[How to use grep command In Linux / UNIX with examples - nixCraft (cyberciti.biz)](https://www.cyberciti.biz/faq/howto-use-grep-command-in-linux-unix/)