Linux Grep Commands

**Overview**

Grep or global regular expression print are set of commands in Linux used to search for and filter through files by searching and returning Lines that contain a pattern of characters. In this document I showcased how I used grep to filter through a file called GrepLab.

**Question 1 Returning Lines that Have the word “Lane”**

The Command I used to get every line to return lines that contain “Lane” for my file GrepLab was **grep “Lane” GrepLab** as shown below:

A screenshot of a computer

Description automatically generated

**Explanation:** I start with grep cased exactly as is then I put the pattern of characters I am looking for in quotes which in this case is **“Lane”** and then the name of the file or Location of the file since I am in the directory of my file I just did the file name which is **GrepLab** This will print any line that contains the pattern of characters in the quotes.

Alternatively, I could have used the **-w** flag. A flag is a set of special characters before the quotes that set an argument on how we want to filter through our file. The **-w** flag makes what I search in the quotes a word instead of pattern characters. The command **grep -w “Lane” GrepLab** returns the following.

A screenshot of a computer screen

Description automatically generated

If you look at both pictures you will notice that that the line that contains “CarltonLane” is missing in the second picture this is because a default grep command without flags searches through a pattern of characters that contain **“Lane”**  but the **-w** flag specially treats what’s in the quotes as a strict whole word. “CarltonLane” has the pattern of “Lane” which is why it shows up in the first picture, but “CarltonLane” is not the same as “Lane” which is what the -w flag argues to specifically look for as a whole word.

**Question 2 Using Grep to find users that have Names starting in H**

To use Grep to find users in my GrepLab file that have a first name starting in H I used the command **grep “^H” GrepLab** as shown below.

A black background with white text

Description automatically generated

**Explanation:** Before I explain my code, I need to do a brief introduction to regular expressions. Regular Expressions or RegEx are special characters used to create and specify certain patterns for an output mainly used to filter through files RegEx is its own can of worms and covering all of RegEx is not in the scope of the document, but several Regular Expressions will be used in my grep answers. With that out of the way I start with **grep** then in the quotes I use what is called an anchor a type of regular expression that specify to look for pattern either at the start of a line or word I used the **^**  anchor to specify that I want my pattern to be from the start of the line since if you look at how my GrepLab file is organized that’s where the names are. The **^** anchor is formatted to be before the pattern of characters you want returned for each line and since I am simply looking for people whose name starts with H I put just H as capital the file name after the quotes.

**Question 3 Using Grep to find Lines that end in 000**

The command I used to print lines that end in “000” was **grep “000$” GrepLab**  as shown below

A screenshot of a computer

Description automatically generated

**Explanation:** I used the $ anchor (see the explanation on question 2 on what anchors and regular expressions are) to specify that the pattern I’d like to see returned from the end of the line since the end numbers are well… at the end of each line this anchor is formatted to be placed after the pattern in the quotes afterwards I put the filename I want to search through.

**Question 4 Using Grep to print lines that don’t contain 408**

The command I used to show lines not containing 408 was **grep -m 10 -v “408” GrepLab** as shown below:

A screen shot of a computer

Description automatically generated

**Explanation:** The **-m 10** part after the **grep** is optional the **-m** flag followed by a number simply limits how many lines do you want display by the number this was put here for so I can show both command and output as the output would fill the whole terminal. The **-v** flag will filter out any lines that contain the pattern of characters in the quotes and simply display a blank line instead. After the quotes put the name or location of the file you are searching which in this case is **GrepLab**

**Question 5 Using Grep to Display Users born in 1935**

The command I used to display users born in 1935 is **grep -E “:[0-9]{1,2}/[0-9]{1,2}/35:” GrepLab** as shown below:

A screenshot of a computer

Description automatically generated

**Explanation:** I used the **-E** flag after **grep** to able to use more characters for regular expressions know as extended regular expressions as by default the only available characters for regular expressions are “?, +, {, |, (, and )” I used the -E flag to be able to use the [, ], and – characters for my expressions.

Since grep is filtered by patterns I needed to identify and unique aspects of how a file is organized or made so I can use those as a unique pattern to find what I am looking for since this related to dates If I see the file I notice that dates like everything else has colons at the start end to border each part of the line and it is formatted in MM/DD/YY so I will add a / in between my expressions no space.

Now In the quotes since this pattern is mainly related to numbers I use the **[0-9]** expression which is used to search for any number in the brackets range which can be used to display any month and day but there is one problem this is specifically for one digit but the dates while using a MM/DD/YY format which is two digits but also the dates don’t always use two digits in each part of the date to cover for both of these issues I use the **{n}** expression after the **[0-9]** no space to show how many digits of that expression I want in this pattern I can have create a maximum and minimum of digits that can be displayed formatted as **{n,m}** since the the MM and DD parts of the date use 1 or 2 digits I do **{1,2}** to show 1 or 2 digits for the expression in the **[0-9]{1,2}** part of the code I then add a **/** since that is part of the pattern I am looking for in the file do the two expressions I just did then a **/** and since I am looking for specifically anyone born in 1935 I don’t need to ensure to be open to show any other years so I just put **35** after the second dash then a closing quote then the name of the file I want to search

**Question 6 Using Grep to Get users with a Phone Number starting in 8**

The command I used to get users with a phone number starting in 8 was **grep -E “:8[0-9]{2}-“**  **GrepLab** Which outputs the following

A screenshot of a computer

Description automatically generated

**Explanation:** I used the -E flag to be able to use the [ ] characters for my special expression (see explanation on question 5 for more information on extended regular expressions) Then in the quotes I first put a **:** since each part of the line bordered by a colon and since this is about the starting number the number will be after a colon so I can use that as a unique part of the text to specify where I want to be. I then put in an 8 since I am looking for phone numbers starting with **8** and two account for the other two digits to get any possible phone numbers I use the **[0-9]** expression which will show any digit in the range of the square bracket I put a **{2}** expression after so I have this done twice to account for the 3rd digit then I add a **–** since that is what borders the rest of the phone number. Outside the brackets I put the name of file I am searching which is **GrepLab**

**Question 7 Using Grep to find a string with an Uppercase then three lowercase a space then one Uppercase**

The command I used to find a string with an uppercase followed by three lowercase letters a space than an uppercase letter is **grep -E “[A-Z][a-z]{3} [A-Z]” GrepLab** as shown below

A screenshot of a computer screen

Description automatically generated

**Explanation**: I used the -E flag after the grep to be able to use the [ ] characters for my special expression (see explanation on question 5 for more information on extended regular expressions) then in the quotes I put the **[A-Z]** expression that will look for any uppercase letter in that position of the pattern I then a **[a-z]** that does the same but for lowercase I then I put a **{3}** to have the **[a-z]** bracket next to be searched for 3 times for the three lowercase letters I then put a space as that is part of the part then one more **[A-Z]** expression for any uppercase letter followed by the space. After the brackets I put the name of the file I was searching which was **GrepLab**

**Question 8 Using Grep to display lines where address number is 2 or 3 Digits**

The grep command I used to display lines where the address number is either 2 or 3 digits is **grep -E “:[0-9]{2-3} “ GrepLab** as seen below

**A screenshot of a computer screen

Description automatically generated**

**Explanation:** I used the -E flag after the grep to be able to use the [ ] characters for my special expression (see explanation on question 5 for more information on extended regular expressions) then in the quotes I put a colon as the colons are used to border each segment of the line and the address number is the starting point of the section of the line I then put a **[0-9]** expression to search for any of the digits in that range in the brackets I then put **{2,3}** expression next to the **[0-9]** expression this is so it that would look for that expression displayed no less than 2 times and no more than 3 times to assure that the address number is either two or three digits. I then add a space since every address number is followed up by a space before getting to the street name. After the quotes I then add the name of the file I am searching which is **GrepLab**

**Question 9 Using Grep to find People who live in Either Mass or Illinois**

The Command I used to find anyone who lives in Massachusetts or Illinois is **grep -E “MA|IL” GrepLab** Outputting the following.

A screen shot of a computer

Description automatically generated

**Explanation**: I used the -E flag after the grep to be able to use **|** to look for any of the patterns between the **|** expression (see explanation on question 1 on what are flags). The **|** expression is used as an or used to separate the either of the two patterns to look for since this question is asking to look for people in Massachusetts or Illinois, I will use the **|** expression since the file has the states in abbreviations, I will format the quotes part as **“MA|IL”** no space this will have it so the grep will look for a pattern of characters that has either **MA** or **IL** after the quotes put the name or location of the file you are searching which is **GrepLab**

**Question 10 Using Grep to filter out anyone living in an address ending in Street**

The command I used for filtering out anyone who has street in their address is **grep -m 10 -v “Street|St” GrepLab** as shown below.

A screen shot of a computer

Description automatically generated

**Explanation**: The **-m 10** part after the **grep** is optional the **-m** flag followed by a number simply limits how many lines do you want display by the number this was put here for so I can show both command and output as the output would fill the whole terminal. Afterwards the -v flag would filter out any line that contains the pattern of characters in the quotes showing lines that don’t have that pattern and the ones that do are blank. The **|** expression is used as an or used to separate the either of the two patterns to look for since this asking to filter out anyone who has Street in their address I will the format the quotes as **“Street|St”** so that my command filters out any line that either has **Street** or **St** in their address after the quotes I put the name or location of the file I am searching which is **GrepLab**.

**Sources Used**

<https://www.opensourceforu.com/2012/06/beginners-guide-gnu-grep-basics/>

<https://stackoverflow.com/questions/5013151/how-do-i-limit-the-number-of-results-returned-from-grep>

<https://www.geeksforgeeks.org/javascript-regexp-0-9-expression/#>