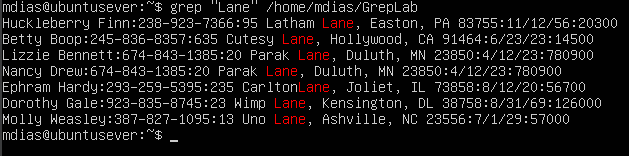
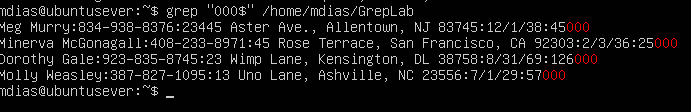
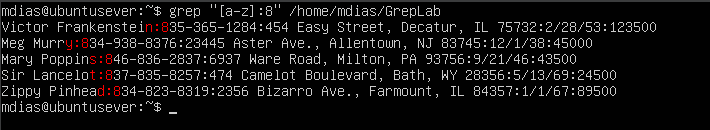
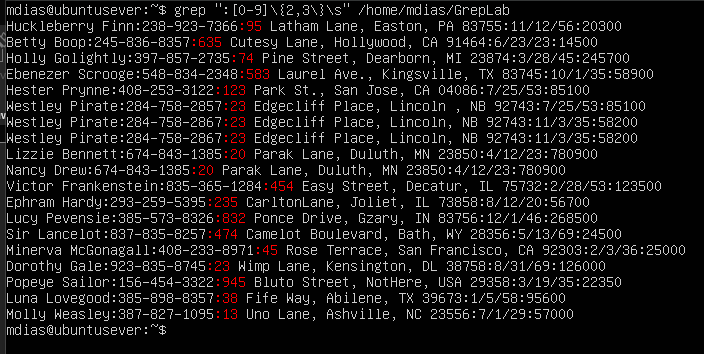
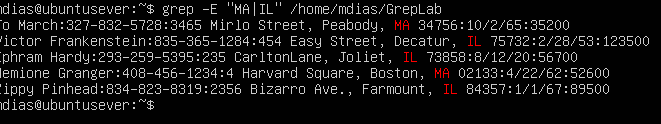
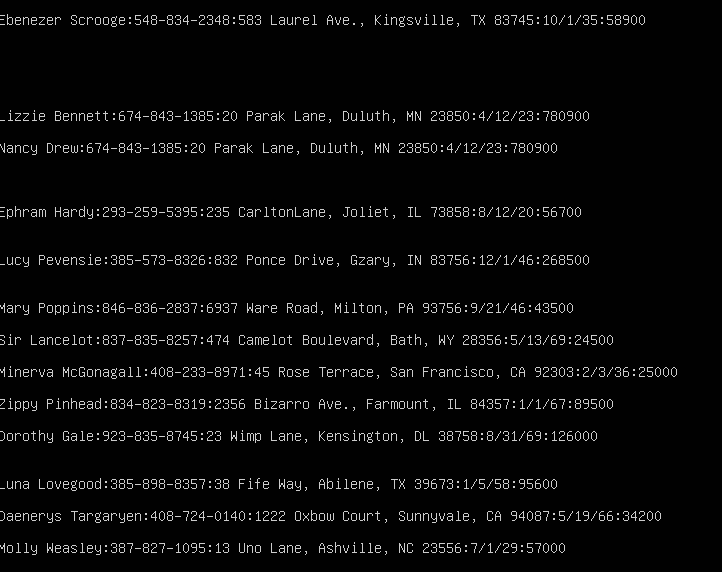
1. Grep (Example done in Ubuntu)
   1. Before we start
      1. Make sure you launch your Ubuntu Server
      2. Open your command prompt in Windows
      3. Transfer the GrepLab file over to the server
   2. Lines containing Lane
      1. Command used: *grep “Lane” /home/mdias/GrepLab*
      2. 
         * Since we are just looking for a specific word, we can simply use “grep” along with the word in quotations. Grep is used to search for text and can be used on its own to find a single word like “Lane”.
   3. Lines with the first name starting with H
      1. Command used: *grep “^H” /home/mdias/GrepLab*
      2. 
         * Since we are looking for words starting with a specific letter, we are going to us “^”. This is an anchor; it is signifying to the machine that I only want lines printed that the very beginning of the line starts with “H”. I was able to use this since we are looking for first names that start with “H” and this document luckily had their first names starting every line!
   4. Lines ending in three zeros (000)
      1. Command used: *grep “000$” /home/mdias/GrepLab*
      2. 
         * Since we are looking for lines that end in something specific like “000” we can use a different anchor. This anchor is “$” which tells the machine that I want it to only print lines that end with “000”.
   5. Lines that don’t contain 408
      1. Command used: *grep -v “408” /home/mdias/GrepLab*
      2. 
         * Since we are looking to have something not included in our print we can use “-v” after grep. -V is telling the machine that I want you to print all the results that do NOT include “408”. Instead of when we use grep alone and we get all the results that DO include that data.
   6. Lines where birthdays are in the year 1935
      1. Command used: *grep “/35\b” /home/mdias/GrepLab*
      2. A screenshot of a computer

         Description automatically generated
         * This one contains a few parts. First, you must keep in mind the structure of the birthdays. Since the year is written as 35, I used that in the command. To really pinpoint this data, I added the “/35” since all the birthday’s do contain a forward slash. I also decided to add a word boundary with “\b” just to ensure the data is only going to print “35” alone and nothing that has additional numbers or letters.
   7. Lines with phone number starts with 8
      1. Command used: *grep “[a-z]:8” /home/mdias/GrepLab*
      2. 
         * For this one we must get a bit creative to get the results we need. Since we need phone numbers with the area code starting with “8” you’d think we could just print lines starting with 8, WRONG. That doesn’t work as easily as you’d think since we have a lot of additional data. I was able to make it a bit more specific. I used a letter range of “[a-z]” along with adding the “:” into the expression to pinpoint this data. All phone numbers are attached to a colon along with the last letter of someone’s last name. Using “[a-z]” makes it so it can be any letter that is lowercase. This expression was able to use those pieces of information to find the results we needed!
   8. Lines that contain an uppercase letter, followed by 4 lowercase letters, a space and one uppercase letter
      1. Command used: *grep “[[:upper:]][[:lower:]]\{4\} [[:upper:]]” /home/mdias/GrepLab*
      2. A screenshot of a computer screen

         Description automatically generated
         * This one has quite a few parts to it so let’s break it down. Based on what we are looking for we need to first have an uppercase letter. You can use “[[:upper:]]” for this and it needs to be followed by 4 lowercase letters. With no space between we add on “[[:lower:]]” to signify lowercase letters. Along with “\{4}\”, this signifies that we want it be 4 letters in a row. Next, you then **MUST ADD A SPACE!** That way it shows proper results and you can then end it with “[[:upper:]]” for our next word beginning with an uppercase letter.
   9. Lines where the address begins with a 2 or 3 digit number
      1. Command used: *grep “:[0-9]\{2,3\}\s” /home/mdias/GrepLab*
      2. 
         * For this one we got a bit creative to make it work like we wanted. Breaking it down, “:” the colon is present to help shrink down the data to make it more accurate (as you can see a “:” is present before each address). The [0-9] is to show that we can have any number be present in the print. “\{2,3\}” is telling the machine that the pattern must be either 2 or 3 digits long, nothing shorter or longer. Finally I added a “\s” which signifies that there is a blank space after. Ensuring we don’t get results that have additional digits.
   10. Lines where someone lives in Massachusetts or Illinois
       1. Command used: *grep -E “MA|IL” /home/mdias/GrepLab*
       2. 
          * For this one we need to use -E since we are using a “|” so it knows to search for both MA and IL. Rather than search for “MA|IL” literally. “MA|IL” is the two results we want to find.
   11. Lines for addresses not on a street
       1. Command used*: grep -vi “St\|Street” /home/mdias/GrepLab*
       2. 
          * For this one I used “-vi”, “-v” is for printing lines that do not contain the data, “-i” I used to make sure it wasn’t case sensitive (help avoid possible lines we could miss). Next there is “St\|Street”, St and Street is what we are looking for, “|” is signifying we want both results, and I used “\” as an escape character for “|”
2. Sources I used for reference, I also used some notes from a Linux Course I took in the past (Along with videos since its been a little while since I used regular expressions)
   1. <https://www.digitalocean.com/community/tutorials/using-grep-regular-expressions-to-search-for-text-patterns-in-linux#regular-expressions>
   2. <https://regexr.com/>