Nathan Christensen-Padilla

Adrianna Holden-Gouveia

Linux Administration

10/12/2021

Awk

1. Code: awk '{print $1}' AwkLab.data >> AwkLab\_output

Text

Description automatically generated

First, I use awk then ‘{print $1}’ which prints out the first variable in the line which is the first name.

1. Code: awk -F: '/Frodo|Tom/ {print $2}' AwkLab.data >> AwkLab\_output



First, I use awk then -F: which makes the delimiter a colon, and then I put ‘/Frodo|Tom/ which is used to search for lines with Frodo or Tom in them using ‘|’ to separate the search parameters. Then I put {print $2}’ which prints out the second variable.

1. Code: awk -F\) '/Peregrin/ {print $1 }' AwkLab.data >> AwkLab\_output



First, I use awk then I write -F\) which makes the delimiter ‘)’ and the backslash is there to break so it counts the right parenthesis as a actual character. Next, I put '/Peregrin/ {print $1 }' which first searches for lines with Peregrin, then prints the first variable in the line, which is everything up to the right parenthesis.

1. Code: awk -F: '/408/ {print $2 }' AwkLab.data >> AwkLab\_output



I start off with awk then ‘-F:’ which turns the delimiter into a colon, next I write ‘/408/ {print $2 }’ which searches for lines that have 408 in them and then prints out the second variable in that line which is the phone number.

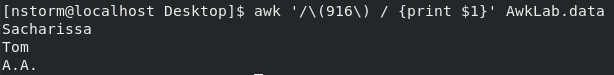
1. Code: awk -F' |:' '/ Ba| Bo/{print $2}' AwkLab.data ; awk -F' |:' '/ D/{print $3}' AwkLab.data >> AwkLab\_outputA screenshot of a computer

   Description automatically generated with medium confidence

The code starts off with awk then it uses -F’ |:’ which makes the delimiter space and colon, so it doesn’t grab part of the phone number with the last name. Then I put '/ Ba| Bo/{print $2}' which searches for words that have space Ba and space Bo in them and then prints the second variable of that line which is the last name. Now for the last part it’s the same as the previous command except it searches for space D and prints the third variable to grab the last, last name that starts with B or D.

1. Code: awk 'length($1) <= 4 {print $1}' AwkLab.data >> AwkLab\_output 

The code starts with the awk command then ‘length($1) <= 4’ which checks the length of first variable to make sure it’s less then or equal to 4 characters, and then prints out the first variable ‘{print $1 }’.

1. Code: awk '/\(916\) / {print $1}' AwkLab.data >> AwkLab\_output

First I use awk then ‘/\(916\) /’ which searches for lines containing the area code 916 by searching for the number 916 surrounded by parenthesis. Then it prints out the first variable in the line which is the first name.

1. Code: awk -F: '/Sacharissa/ {print "$"$3 ",$"$4 ",$"$5}' AwkLab.data >> AwkLab\_output

I start with awk then ‘-F:’ to change the delimiter to colon, and then use ‘/Sacharissa/’ to find the line with Sacharissa in it and then it prints out the third, fourth and fifth variables with “$” to print dollar signs at the beginning of each number and commas behind the dollar sign so they aren’t all clumped up together.

1. Code: awk -F: '{print $1","$2}' AwkLab.data | awk '{print $2 " "$3 $4}' >> AwkLab\_output Text

   Description automatically generated

First I use the awk command then ‘-F:’ which makes the delimiter colon, then I put ‘{print $1”,”$2}’ which prints the first and second variable separated by a comma. This gets the first and last names and the phone numbers. Then this output gets piped into the next awk command which is '{print $2 " "$3 $4}' and first it prints the second variable add a space with “ “ then prints the third and fourth variables.

1. Code: The code is in the Awk\_Script File attached but I have provided some screenshots of the output and the code.

Text

Description automatically generated

Text

Description automatically generated

Text, letter

Description automatically generated

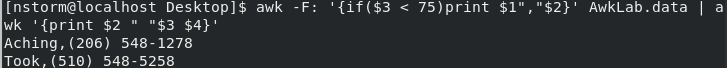
2nd Part:

1. Code: awk -F: '{if($5 > 110)print $1}' AwkLab.data >> AwkLab\_output

Text

Description automatically generated

First I use the awk command then I put ‘-F:’ to change the delimiter to a colon then I put the statement ‘{if($5 > 110)print $1}’ which first checks if the fifth variable (the last month contribution) is more then 110, and if it is then it prints out the first variable which is the first and last name.

1. Code: awk -F: '{if($3 < 75)print $1","$2}' AwkLab.data | awk '{print $2 " "$3 $4}' >> AwkLab\_output

First I start with awk then I use ‘-F:’ to change the delimiter into a colon, then I put the statement ‘{if($3 < 75)print $1”,”$2}’ which first checks if the third variable (First month contribution) is less than 75 and if it’s true then it prints out the first and second variable separated by a comma. Now the first command gets piped to this next part which is used to get rid of the first name, ‘{print $2 “ ”$3 $4}’ first it prints out variable 2 which is the last name, then “ “ adds a space and finally prints variable 3 & 4 which is the phone number.

1. Code: awk -F: '{if(75 < $3 && $3 < 150)print $1}' AwkLab.data >> AwkLab\_output

I start off with awk then -F: to change the delimiter to a colon then I put the command ‘{if(75 < $3 && $3 < 150)print $1}’ which first checks if the third variable (first month donation) is higher then 75 and (&&) lower then 150, and if it meets those requirements it print’s out the results.

1. Code: awk -F: '{if($3 + $4 + $5 < 700)print $1}' AwkLab.data >> AwkLab\_outputText

   Description automatically generated

The code starts with awk -F: which makes the delimiter a colon, then I put ‘{if($3 + $4 + $5 < 700)print $1)}’ which first checks if the three donations are less then 700, and if so then it prints out the first variable which is the First and Last name.

1. Code: awk -F: '{if($3 + $4 + $5 / 3 > 100 )print $1}' AwkLab.data | awk -F" " '{print $1 " "substr($2,0,1)}' >> AwkLab\_output Text

   Description automatically generated

First I start with awk -F: which changes the delimiter to a colon, then I put '{if($3 + $4 + $5 / 3 > 100 )print $1}' which checks if the third, fourth, and fifth variable divided by three is greater then 100 and if so then it prints the first variable. The reason I did this was because the third, fourth, and fifth variables are the donations and adding them all together and dividing by how many numbers I added together is the average. Then I piped the result to the next command which first changes the delimiter to “ “ space and then I put '{print $1 " "substr($2,0,1)}' which first prints the first variable (first name) and then substr splits up the second variable and prints out only the first letter of the string. Finally, we get the first name and the first letter of the last name.

1. Code: awk -F: '!/(916)/{print $1}' AwkLab.data | awk '{print $2}' >> AwkLab\_output Text

   Description automatically generated

First I use awk -F: which changes the delimiter to a colon, then I put '!/(916)/{print $1}' which searches for line that don’t contain the area code 916, and then prints out the first variable. Then this output gets piped into the next command which prints out the second variable, which is the last names.

1. Code: awk '{print NR" "$0}' AwkLab.data >> AwkLab\_outputA black screen with white text

   Description automatically generated with low confidence

First I use awk then '{print NR" "$0}' which prints out the number of the line, and a space to look visually better, then it prints out the whole line.

1. Code: awk -F: '{print $1", Total Contribution: $" $3 + $4 + $5}' AwkLab.data >> AwkLab\_outputText

   Description automatically generated

I start off with awk -F: which changes the delimiter to a colon, then I put '{print $1", Total Contribution: $" $3 + $4 + $5}' which first prints the first variable (name), then it says “Total Contribution” for better visual appearance, and finally it adds the third fourth and fifth variables together which are the contributions.

1. Code: awk -F: '/Tiffany Aching/{print $3 + 10}' AwkLab.data >> AwkLab\_output

First I start with awk -F: which changes the delimiter to a colon, then I type '/Tiffany Aching/{print $3 + 10}’ which first searches for the line with Tiffany Aching in it, then it adds +10 to variable $3 (first contribution) and then finally prints out that variable.

1. Code: awk -F: '/Samwise Gamgee/{ sub($1,"Sean Astin"); print $0}' AwkLab.data >> AwkLab\_output 

First I used awk -F: which makes the delimiter a colon, then I put '/Samwise Gamgee/{ sub($1,"Sean Astin"); print $0}' which first finds the line with Samwise Gamgee and then it substitutes the first variable (the first and last name) with Sean Astin, and then finally it prints the whole line.

Sources:

Links I found:

[How to use awk to select substring in linux/unix - LinuxCommands.site](https://www.linuxcommands.site/linux-text-processing-commands/linux-awk-command/awk-substr/)

This website helped me better use commands, because it gave simple definitions and instructions on how to use them like substr, if and length.

[AWK - String Functions (tutorialspoint.com)](https://www.tutorialspoint.com/awk/awk_string_functions.htm)

This website helped me define more awk commands like “sub” and it also had a lot of useful information about scripting in it.

Links you provided:

<https://youtu.be/4HpWO_RAMq4>

[Chapter 6. The awk Utility | UNIX Shells by Example (4th Edition) (flylib.com)](https://flylib.com/books/en/4.356.1.52/1/)

[6.13. Variables | UNIX Shells by Example (4th Edition) (flylib.com)](https://flylib.com/books/en/4.356.1.67/1/)

[6.17. Conditional Statements | UNIX Shells by Example (4th Edition) (flylib.com)](https://flylib.com/books/en/4.356.1.72/1/)