Awk Lab

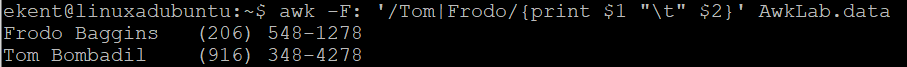
Ethan Kent

Awk is a useful tool which is a programming language. Though the things I will be using it for won’t show off that much of what it can do. It is good for things such as looking for patterns or specifics in files. Though it does read every line so a big file could be very resource heavy if used.

Prompt 1: In our file I want to find only the first names of everyone listed. To do so I will use, awk ‘{print $1}’ filename. The way this works is that awk can find sections of a line using “$” it uses spaces to distinguish each different part of the file (we can also change what it decides to split it by) so since I asked it for “$1” I asked for the first part of each line and to print them out.

Text

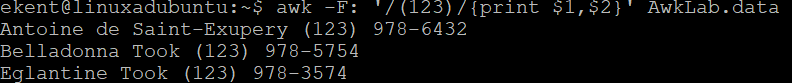
Description automatically generated

Prompt 2: We need to find Tom and Frodo along with their phone numbers. Though with how the file is set up looking for spaces as the dividing character we need to change this up. The command used is, awk -F: '/Tom|Frodo/{print $1 "\t" $2}' filename. The “-F:” is what is going to tell awk to look for “:” instead of a space for our searches. The “/” is to tell awk to look for a pattern, the “|” means or. So our command says “Look for Tom or Frodo”. The “\t” is a tab which is used to space our two sections out that we want printed. 

Prompt 3: Let’s print just Peregrins name and area code. awk '/Peregrin/{print $1,$2}' filename. We search for their name, print it along with the second section which holds last name and the area code after it.



Prompt 4: Let’s grab everyone from the area code (123).

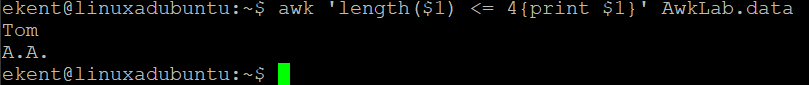


We used “-F” again to give us better results with our search. Looking specifically for that “(123)” and then printing the first section will give us first and last name. Since we used a colon to determine our splits in the sections “$1” “$2” we get the area code and phone number without that space interrupting.

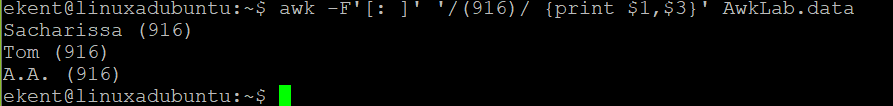
Prompt 5: we’re going to look for only last names with T and D. To do so we need to break things up, sequence by sequence. So our first half “awk -F: ‘{print $1}’ filename, this is going to give us just the whole list of names and nothing else. I only want the end of this so let’s use “NF” NF is used to grab the end of the section we’re looking for which is all the last names. Now we look for the names with only letters T and D. to do so I’m going to use a regular expression with “-e” get our syntax to tell them where we want to look in this case the first section “$” we want /[T]|[D]/ this says “Look for a capital T or capital D” Text

Description automatically generated

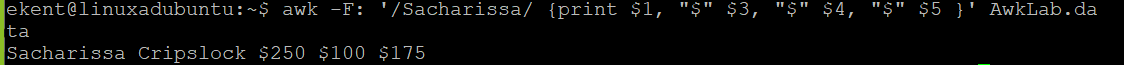
Prompt 6: Looking for things based on how long or short they are. Were going to look for all names that are four or fewer characters. awk ‘length($1) <= 4{print $1}’ filename.



‘length’ is looking for its name sake. The ‘($1)’ attached to it is telling what line to look at and the ‘<=’ is less then or equal to, so find something four or less, then print out just the first section with {print $1}

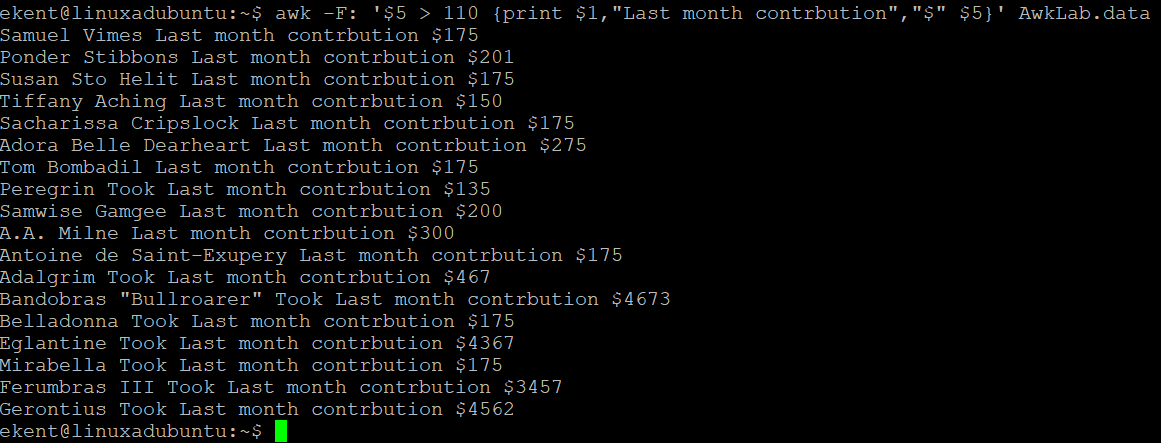
Prompt 7: We want to find everyone who has the area code “(916)” along with their first name. To do so we need to break up our sections more precisely. So we take the “-F” and give it the square brackets as a way to tell that we want to split our sections by both spaces and a colon, then we can just print out the sections we want.

Prompt 8: We need get Sacharissa and her campaign contributions. To look for her contributions easier I made a colon our split. We can search for her name and then print out the contributions, we also want to add the dollar sign as well so we know it’s money. To do so we need to use double quotes before the section we want. The double quotes tell us that we want to place something and we want to do it right at this spot.



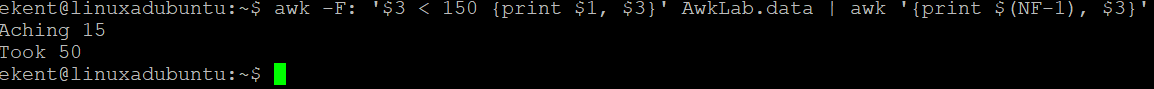
Prompt 9: We want to split our last names and phone numbers by a comma. Let’s make it convoluted! First things first lets print the general idea of what we want with the first part “awk -F: ‘{print $1 “,”, $2}’ filename. Here we said “Print the names and the phone numbers with that comma in-between” though we have the first names still which we don’t want. So, let’s take that “NF” we learned about and make it look through each section with some math. Since NF is end of line we want it to go back in the line. To go backwards we need to write it out with the “()” since these are where we do our math. We need them to tell awk that were subtracting NF’s last position and going backwards. We can do this with multiple positions and print them out. Once we print them in the order we want we get the info we need! So “‘{print $(NF-2), $(NF-1), $NF}’ gives us the format of the information we want.Text

Description automatically generated

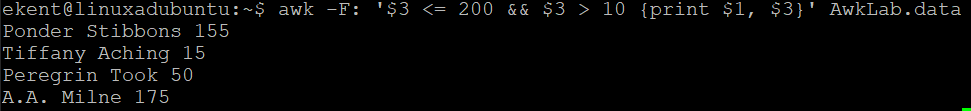
Prompt 10: Let’s get everyone who contributed more then 110 dollars last month. Here’s the results.

Since I wanted only those who contributed over 110 I used simply “$5 > 110” which is saying “If section 5 is bigger then 110 print” Then I wanted the names so I print the first section “$1” to make it distinguishable I have it print out what the number is and the dollar sign after section 5 along with a message telling what the amount of money is for.

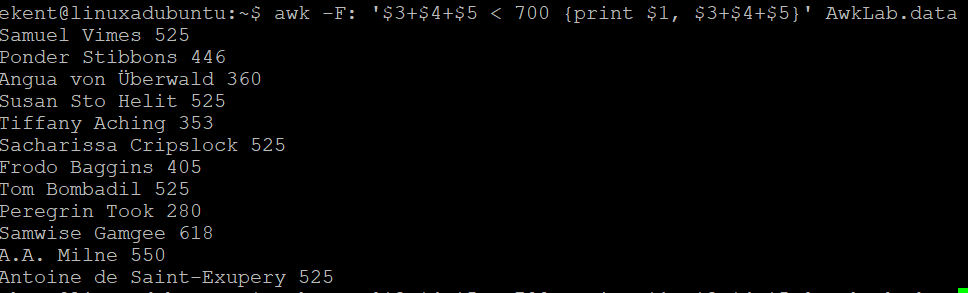
Prompt 11: Let’s grab the first month contributions of anyone who gave less than $150 dollars along with their last names. Do our math out first “‘$3 < 150” so section 3 is where the first month contribution is so now awk knows where to apply this math. Then let’s just print out full names and the contributions. So now we have it formatted mostly how we want, now we just need to grab last names. So lets pipe a command “|” we have this command say “now that we have how we want formatted, go back 1 space from the first month contribution to the last name” and it gives us just the last name and the contribution I asked for.



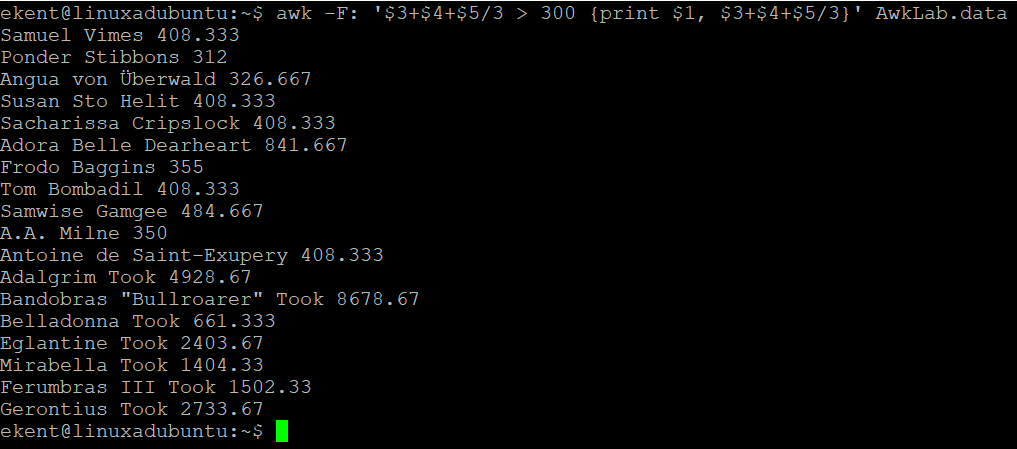
Prompt 12: Find a range of peoples contributions for the first month, only those who contributed 10 to 200 dollars though. To do so we just do out the equations and use “&&” to say “and” then we can just print out what we want for lines and we get the range of contributions we want.



Prompt 13: Let’s find who contributed $700 dollars or less. To do so we just need to add up the sections we know are our contributions and make sure that if it’s less 700 we print the full name of said person. “$3+$4+$5 <700” saying “Add these sections and if less then 700” {print $1, $3+$4+$5} “print out the name and result of all the added contributions”



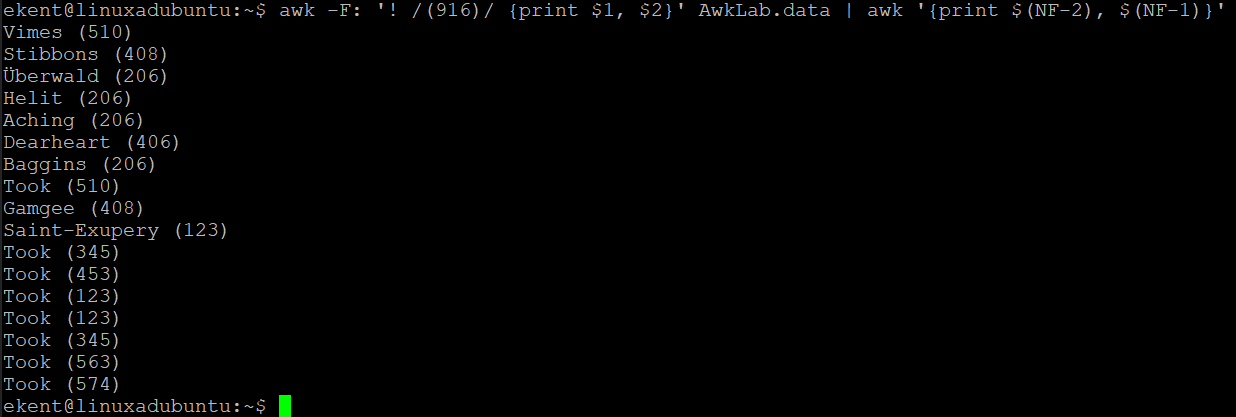
Prompt 14: For this prompt we need to grab the first names and last names first letter. To do so I was trying to use the split function but was unable to get it to work to get what was asked. I was trying to have it search for spaces and then after that a capital letter but couldn’t get them both to work or for the capitals to work. I tried “awk -F: ‘{split($1,a,” “[A-Z])}” Though I kept getting errors. I was able to get the second half which is finding those who contributed over 300 dollars on average though. Just have to add each of the contributions and divide them by three, make sure they say greater then as well then print out the names and the equation.



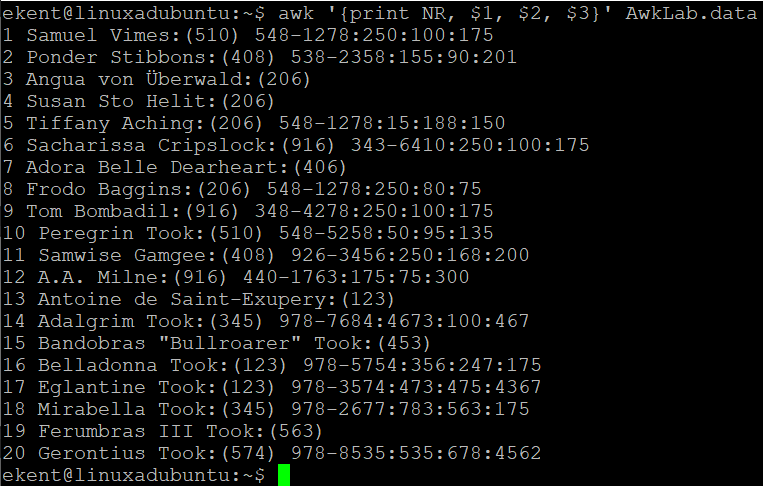
Text

Description automatically generated

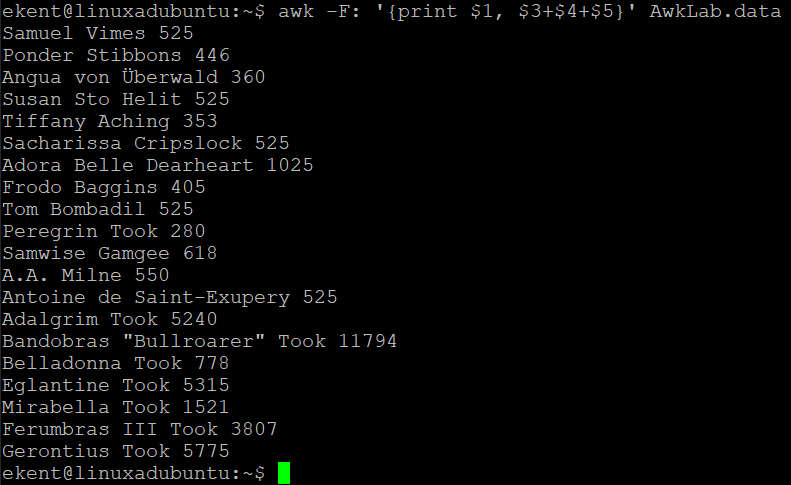
Prompt 15: We want to find everyone who isn’t in the (916) area code. We can use an exclusion with “!” the start of the line says “DON’T grab (916)” then we want the names and numbers, looking back at other prompts that only want specific things we can use “$(NF-x)” to get the positions of the areas we want.



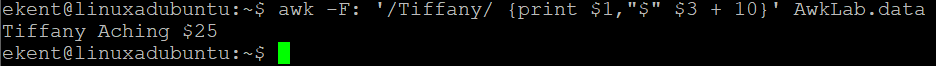
Prompt 16: Print the number record and then each line. To do this I used “NR” which is what prints out the number for each record on file, then after that I just had it print each field. Now each record shows what number it is on file and the information with it.

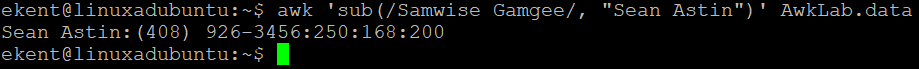


Prompt 17: Print all names and their total contributions. A nice and easy one, print our names by section “$1”, then print all the contributions added together with “$3+$4+$5”



Prompt 18: We need to fix a mistake on file. Tiffany donated 10 more dollars then we have written down. We use “/Tiffany/” to grab her line so that we change just hers. Then grab her name “$1” and add to her first month contribution “$3 + 10”.



Prompt 19: Looks like we mixed up someone’s name, let’s do a substitution, it is a bit similar to sed. Instead of an “s” we use “sub” the “()”s to tell what is being changed. Using the “/”s we say this is what is going to be substituted (if found), then the double quotes is what we want to replace it with. Then it shows us the line after it’s been changed.

Prompt 20: Let’s make a script looking for somethings we need. This is our script. We asked awk that at the start we want our delimiter to be a colon and to do this before anything else, indicated by the “BEGIN”. The next part is us going and grabbing all of the Tooks by their first name and then the total amount they contributed. To grab their first names and only their first name I’m using the split function. First, I declare where I want to split from, line one where we have the name. I assign a variable to put the result of the split and after that I tell the split function where I want to split things. In this case a space. Then when I print by assigning “a[1]” it will print out what I want. The other part is just all three contributions added together, I added a printout to make it easier to read the results by using the double quotes. The second half is finding anyone who contributed between two hundred and ten dollars as their last month’s contribution, to do so I just did greater then and less the for the values, the “&&” saying “and” so we have both values calculated together. Then print the names and last contribution to get what we need, I tried to make a printout that would make it easier to read but it wouldn’t print out multiple instances and they had nothing next to them, so I chose to leave this one alone. The last part is getting all contributions averages and seeing whose total contribution was less than 300. To start I did the math for it, adding them all and dividing them, less then 300 and printing out the totals with a blurb to help read it. The script printed looks like this.

Graphical user interface, text

Description automatically generated

Text

Description automatically generated

Sources used for lab

<https://www.theunixschool.com/2012/05/awk-match-pattern-in-file-in-linux.html>

<https://riptutorial.com/awk/example/17378/length--string-->

<https://www.unix.com/shell-programming-and-scripting/56223-awk-greater-than.html>

<https://www.tecmint.com/learn-awk-variables-numeric-expressions-and-assignment-operators/>

<https://www.poftut.com/awk-text-split-delimit-examples/>

NECC Tutoring Center: Noah