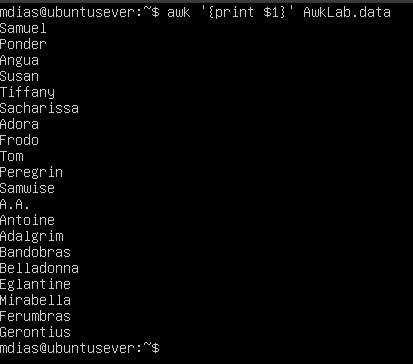
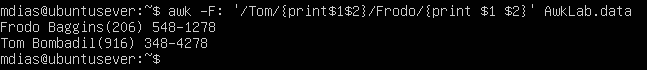
**Awk Lab**

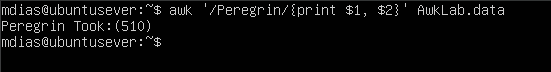
1. Print all the First Names.

* Command: **awk ‘{print $1}’ AwkLab.data**
* Screenshot:
* Explanation: Using awk we are trying to print all the first names in the data set. We use the curly braces {} to define an action we need done. Print is used to display data. ‘$1’ represents a field of input. First names are in the first field of the data set so we use the number 1. Lastly you refer to the file with AwkLab.data

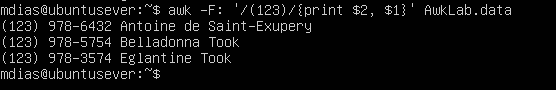
1. Print phone numbers for Tom and Frodo after their names

* Command: **awk -F: ‘/Tom/{print$1$2}/Frodo/{print $1 $2}’ AwkLab.data**
* Screenshot:
* Explanation: -F: is a setting the delimiter to : rather than blank space. /Tom/ is a pattern that is calling for the lines that contain Tom. When the line is found it uses the action of {print$1$2} which prints field 1 ($1)which is names and field 2 ($2) which is the phone numbers. The same is done for Frodo just using his name.

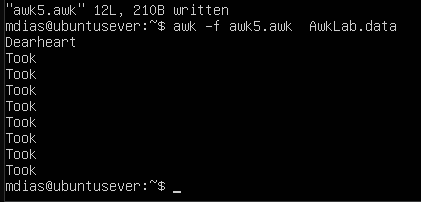
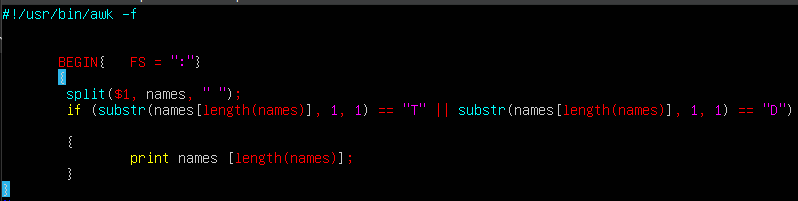
1. Print Peregrin’s full name and phone number area code only.

* Command: **awk ‘/Peregrin/{print $1, $2}’ AwkLab.data**
* Screenshot:
* Explanation: This works very similarly to the previous questions. We use awk and set the pattern with /Peregrin/ and an action of [print $1, $2}. The action is calling to print the 1st ($1) and 2nd ($2) fields that contain Peregrin.

1. Print all phone numbers (full number) in the 123 area code along with the names

* Command: **awk -F: ‘/(123)/{print $2 $1}’ AwkLab.data**
* Screenshot:
* Explanation: This also works very similarly to the last few questions. -F: once again is telling the system to recognize : as the delimiter. /(123)/ is setting the pattern followed by the action. If the pattern is found it will perform the action of {print $2, $1}. The action prints first field 2 ($2) which is phone number followed by names which is field 1 ($1).

5. Print all Last names beginning with either a T or D (careful of middle names!)

* Command: **\*See Script screenshot & explanation breakdown of each statement\***
* Screenshot:
* Explanation: **#!/usr/bin/awk -f** is telling the script to execute using AWK

**BEGIN { FS= “:” }** this sets the field separator in the script to a colon

**Split($1, names, “ “);** this is splitting field 1 which is the names, we are naming this names so we can reference it later, and using blank space as a delimiter

**If (substr(names[length(names)], 1, 1) == “T”** this is extracting the last name from the array names so we can just print last names and avoid middle names. We are also stating at the end (==”T”) to state we want the first letter be equal to the letter T.

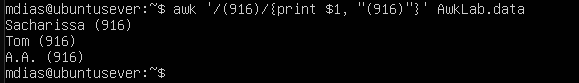
**If (substr(names[length(names)], 1, 1) == “T”** this works exactly the same beside the fact that we are looking for the first letter of the last name to be D (==”D”).

**Print names [length(names)];** we are saying to print the names array that we have stated in a previous line and recalled and other lines. We are then calling for the value that is stored in the array. Which is the last name that we are calling to be printed.

1. Print all first names containing four or less characters.

* Command: **awk -F ‘[: ]’ ‘{if (length($1) <=4) print $1}’ AwkLab.data**
* Screenshot:
* Explanation: -F: sets our delimiter to : rather than blank space. We then move into an if statement that is checking the length of the first field ($4) is less than or equal to 4(<= 4). If this is true it will print the first field {print $1}.

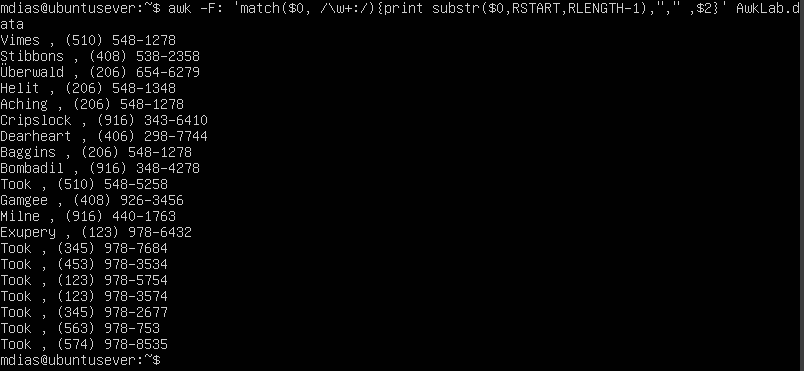
1. 7Print the first names and area codes of all those in the 916 area code.

* Command: **awk ‘/(916)/{print $1, “(916)”}’ AwkLab.data**
* Screenshot:
* Explanation: We are setting the pattern with // to (916) which is the area code. If the pattern is found the action is saying the print the first field ($1) of those lines and to print the area code that is (916).

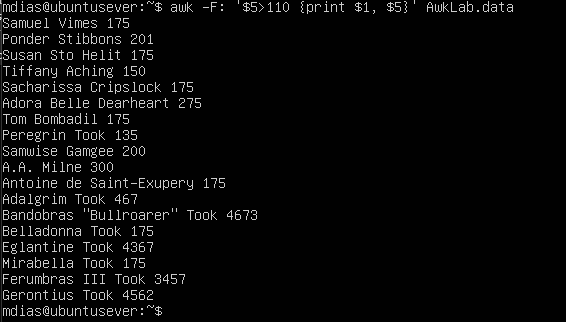
1. Print Sacharissa’s campaign contributions following her name. Each value should be printed with a leading dollar sign; e.g., $250 $100 $175.

* Command: **awk -F: ‘/Sacharissa/{print $1, “$”$3, “$”$4, “$”$5}’ AwkLab.data**
* Screenshot:
* Explanation: -F: is once again setting the delimiter to : for the command. We are then setting the pattern to be found which is Sacharissa using // around it. If the pattern is found the action performed is a print starting with field 1 which is the name. Next we are printing a dollar sign proceeded by field 3, 4, and 5 ($3 $4 $5) Which represent first, second, and third months contributions (in that order).

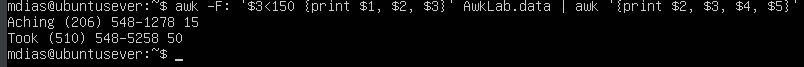
1. Print last names followed by a comma and the phone number. Be careful of the last names’s format.

* Command: **awk -F: ‘match($0, /\w+:/){print substr($0,RSTART,RLENGTH-1),”,” ,$2}’ AwkLab.data**
* Screenshot: 
* Explanation: -F: is once again setting the delimiter to :. Next is an awk program. Match is used to search the entire line of $0 for a specific pattern based on the regular expression. The regular expression is /\w+:/ which is saying to match one or more characters (letters, numbers) but more specifically a : since I stated that in the expression. So, it’s looking to find a word followed by a colon which is how the last names are formatted. When a match is found it returns to the position that the match started. The next line of print substr($0, RSTART, RLENGTH-1) is essentially taking the matched word that is followed by the colon and extracts it. Next, we add a comma “,” and lastly print the phone numbers out of field 2 with $2.

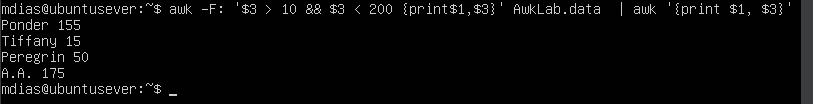
1. Print the first and last names of those who contributed more than $110 in the last month. Make sure to include their last month contribution amount after the name

* Command: **awk -F: ‘$5>110 {print $1, $5}’ AwkLab.data**
* Screenshot:
* Explanation: -F: once again sets the delimiter to :. The pattern of $5 > 110 is finding if the 5th field is greater than 110. If that condition is true, it will execute the command by printing the first and fifth field. First field being the name and fifth being the last months contribution.

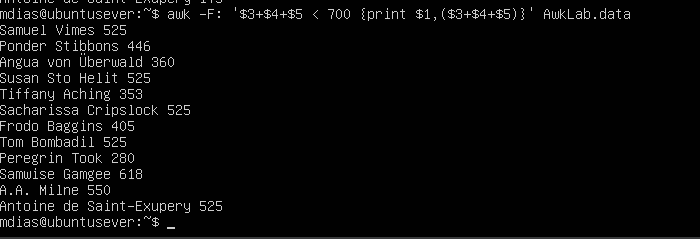
1. Print the last names, phone numbers, and first month contribution of those who contributed less than $150 in the first month.

* Command: **awk -F: ‘ $3<150 {print $1, $2, $3}’ AwkLab.data | awk ‘{print $2, $3, $4, $5}’**
* Screenshot:
* Explanation: -F: once again sets the delimiter to :. The pattern of $3<150 checks if the value of the third field is less than 150. $3 represents the first month’s contribution. If this condition is true it performs the action of {print $1 $2 $3} which would print field 1, 2, and 3. We then pipe the command with | and print field 2 which is last names, field 3 which is the area code, field 4 which is the phone number, and field 5 which is the first month contribution that is less than 150.

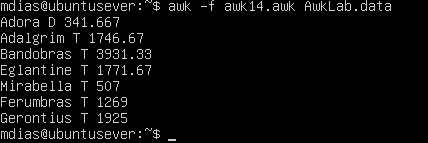
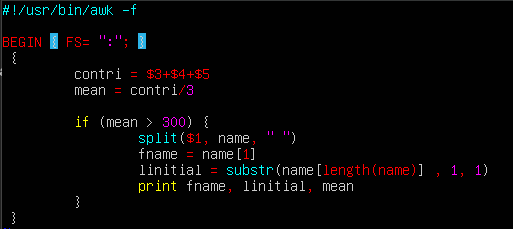
1. Print the first names and contribution of those who contributed between $10 and $200 in the first month.

* Command: **awk -F: ‘$3 > 10 && $3 < 200 {print$1, $3}’ AwkLab.data | awk ‘{print $1 $3}’**
* Screenshot:
* Explanation: -F: once again is setting the delimiter to :. $3>10 && $3<200 is the pattern we are looking for. It is stating that we are looking for field 3 which is the first months contributions. To be greater than 10 and less than 200. If the condition is met we want to print field one which is name and field three which is first months contributions. We then pipe this with | and printing only the first name with $1 along with the contributions that fit the criteria from $3.

1. Print the first name, last names and total contributions of those who contributed less than $700 over the three-month period.

* Command: **awk -F: ‘$3+$4+$5 < 700 {print $1, ($3+$4+$5)}’ AwkLab.data**
* Screenshot:
* Explanation: -F: is once again setting the delimiter to :. The pattern we are setting up is adding fields 3, 4, and 5 which the first, second, and third months contributions ($3+$4+$5). The we are only looking for results less that 700 (<700). If this condition is true then we perform the action of printing the first last name ($1) and the total of the 3 fields that are less than 700 ($3+$4+$5)

1. Print the first names and first letter of the last name, and average contribution of those who had an average contribution of more then $300

* Command: **\*See Script screenshot & explanation breakdown of each statement\***
* Screenshot:
* Explanation: **#!/usr/bin/awk -f** is telling the script to execute using AWK

**BEGIN { FS= “:” }** this sets the field separator in the script to a colon

**Contri = $3+$4+$5** this is setting a variable contri to the sum of field 3, 4, and 5. contri is short for contributions.

**Mean = contri/3** mean is taking the result on contri and finding the meanby dividing it by 3

**If (mean > 300) {**

**Split ($1, name, “ “)**

**Fname = name[1]**

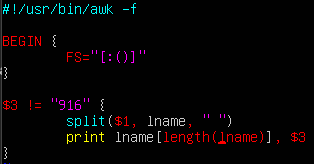
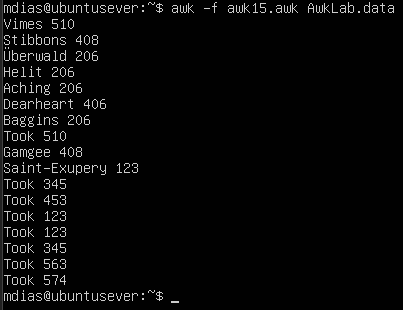
**Linitial = substr(name[length(name)] , 1, 1)**

**Print fname, linitial, mean**

First we made the pattern of the mean being greater than 300. Next we are splitting field one and calling it name. We then set the variable fname to the first name. Then we are setting up a variable to the last name first initial to linititial. We then are extracting the first initial of the last name using substr to extract and name[length(name) to extract to the last part of the array we split and having it produce the results of the first letter. Ending with printing all the variables we made in order of the question.

1. Print the last name and area code of those not in the 916 area code.

* Command: **\*See Script screenshot & explanation breakdown of each statement\***
* Screenshot:



* Explanation: **#!/usr/bin/awk -f** is telling the script to execute using AWK

**BEGAN { FS= “:” }** this sets the field separator in the script to a colon

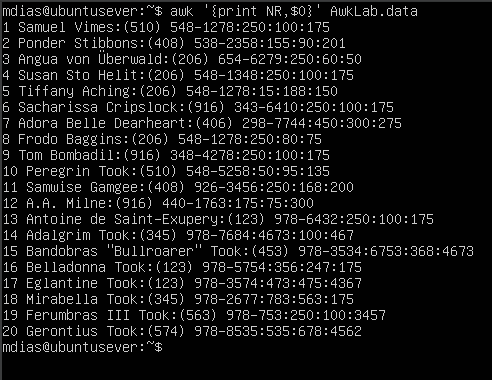
**$3 != “916** is saying that we do not want the values of the third field that contain 916

**Split($1, lname, “ “)**

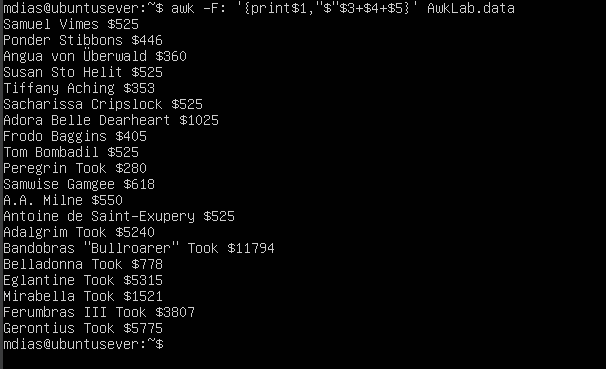
**Print lname [length(lname}, $3**

First we split field one which is the name and I called this variable lname because we want the last name. We then state to print the using length to print just the last name along with field 3. Still keeping in mind that results containing 916 wont print.

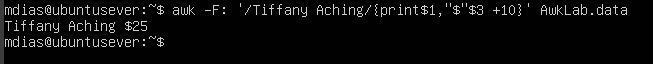
1. Print each record preceded by the number of the record.

* Command: **awk ‘{print NR, $0}’ AwkLab.data**
* Screenshot:
* Explanation: This is a very quick command. We want to print all results and number them using a built in awk variable of NR. $0 represents the entire line. Resulting in printing all lines and numbering them in order.

1. Print the name and total contribution of each person.

* Command: **awk -F: ‘{print$1,”$”$3+$4+$5}’ AwkLab.data**
* Screenshot:
* Explanation: Once again -F: is setting the delimiter to :. We are then printing $1 which is the names, we then print a dollar sign that will sit in front of the total, we then print the total by adding $3+$4+$5.

1. Add $10 to Tiffany Aching’s first contribution and print her full name and first contribution.

* Command: **awk -F: ‘/Tiffany Aching/{print$1,”$”$3 +10}’ AwkLab.data**
* Screenshot:
* Explanation: Once again -F: is setting the delimiter to :. We then set the pattern to Tiffany Aching using //’s. Following by the action of when the result is found to print $1 which is the name, along with printing the first contribution $3 with adding 10 dollars on by throwing +10 on the end.

1. Change Samwise Gamgee’s name to Sean Astin

* Command: **awk ‘{gsub(“Samwise Gamgee” , ”Sean Astin”);print $1, $2}’ AwkLab.data**
* Screenshot:
* Explanation: Starting with gsub which will search the entire file for Samwise Gamgee and it will replace it with the new value of Sean Astin. I added a print statement so you can see the new results.

1. Write an awk script to do the following (MUST be an awk script not just a bash script or commands on the commandline)

**#!/usr/bin/awk -f** is telling the script to execute using AWK

**BEGAN { FS= “:” }** this sets the field separator in the script to a colon

1. Prints first name of the all the Tooks followed by their total campaign contributions

**/Took/ {** \*\*stating the pattern we are looking for which is the last name Took

**Total = $3 + $4 + $5** \*\*stating a variable (total) and adding the contribution fields

**Split($1, name, “ “);** \*\*splitting the name so we can use the first name and naming the variable (name)

**Print name[1], “$” total, “(Took’s Total Contribution)” }** \*\* printing the first value of name which is the first name. Along with a $ and the total which we stated a few lines up. I also added a statement of “Took’s total contributions” to make the results more clear.

1. Print the full names and contributions of anyone who contributed between $10 and $200 in the last contribution

**$5 > 10 && $5 < 200 {print $1, “$”$5 }** \*\*we are stating the pattern we are looking for. We are taking field $5 which is the last contribution and ensure its greater than 10 (>10) and less than 200 (<200). We then print $1 which is the full name, a dollar sign and then field $5 with the results of the pattern.

1. Prints the full names and average contribution of those who contributed less than $300 on average

**$3+$4+$5/3<300 {print $1, “$”$3+$4+$5/3, “(Less Than 300)”}** \*\*stating our pattern, starting with finding the average so we are adding fields 3, 4, and 5. Then divided that by 3 (/3) and state that we only want results less than 300 (<300). Then we want to state our action by printing $1 which is the full name. Then a dollar sign with the average ($3+$4+$5/3). I also added a statement “Less Than 300” to make this easy to identify. A screenshot of a computer

Description automatically generatedA screenshot of a computer code

Description automatically generated

**Resources**

[**https://www.geeksforgeeks.org/awk-command-unixlinux-examples/**](https://www.geeksforgeeks.org/awk-command-unixlinux-examples/)

[**https://phoenixnap.com/kb/awk-command-in-linux**](https://phoenixnap.com/kb/awk-command-in-linux)

[**https://www.grymoire.com/Unix/Awk.html**](https://www.grymoire.com/Unix/Awk.html)