Joseph Williams

AWK Lab

Instructions:

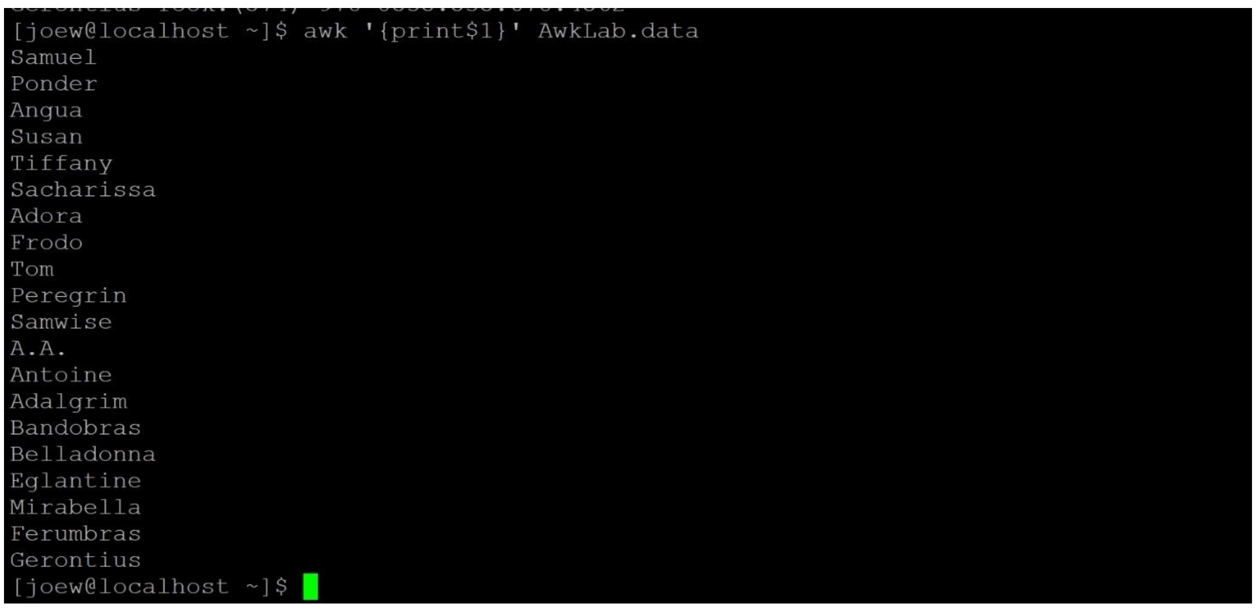
Awk is a scripting data manipulation programming language used to generate reports, similar to SED. In addition, it is a powerful tool for text analysis and piping data bits into a text file. In this lab, you will learn some basic concepts of awk programming on a Linux server. We will be using an awk data file called “AwkLab.data” to create our report.

Lab Report (Commands and Descriptions)

Below you can take a look at lab resources, screenshots, and walkthroughs on manipulating data using awk commands. Several phrases will be used to generate the data, along with the expression syntax required for each assignment in this lab. For example, the phrase "\t" is primarily used to space in the brackets, while "FS" is used to divide fields on the input line. It can also be used to reassign another character to change the field procedure. The character “|” means pipe, which can be describe as what the Linux program identified. As we move along, you find the meaning of those phrases in each exercise and what it does.

1. Print all the First Names.

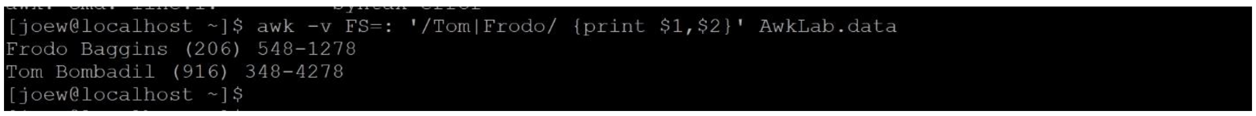
The command: *awk ‘{print$1}’ AwkLab.data* will generate the data of the results as shown in below. The “awk” indicate that the command is awk request, and ‘{print$1}’ will print all the first names, followed by the AwkLab data textfile.



```
[joew@localhost ~]$ awk '{print$1}' AwkLab.data
Samuel
Ponder
Angua
Susan
Tiffany
Sacharissa
Adora
Frodo
Tom
Peregrin
Samwise
A.A.
Antoine
Adalgrim
Bandobras
Belladonna
Eglantine
Mirabella
Ferumbras
Gerontius
[joew@localhost ~]$
```

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

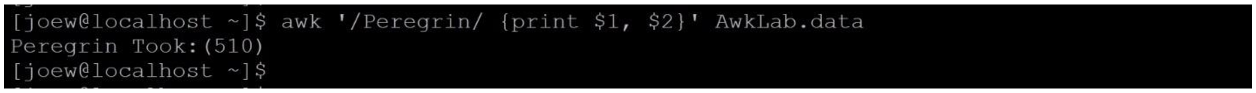
The command: *awk -v FS=: ‘/Tom | Frodo/ {print $1,$2}’ AwkLab.data* will print the phone numbers for Tom and Frodo. The “awk” indicate that it is an awk request, followed by “-v”, “FS” means Field Separator and then the file name “AwkLab.data



```
[joew@localhost ~]$ awk -v FS=: '/Tom|Frodo/ {print $1,$2}' AwkLab.data
Frodo Baggins (206) 548-1278
Tom Bombadil (916) 348-4278
[joew@localhost ~]$
```

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

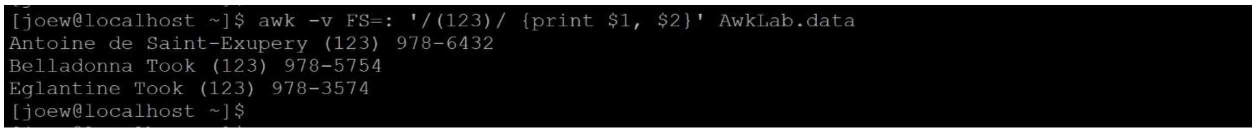
The awk statement: *awk ‘/Peregrin/ {print $1, $2}’ AwkLab.data* will print the full name of Peregrin and the area code. The number “$1” represents the first name, and the $2 stands for the last name and then $3 represents the area code.



```
[joew@localhost ~]$ awk '/Peregrin/ {print $1, $2}' AwkLab.data
Peregrin Took:(510)
[joew@localhost ~]$
```

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

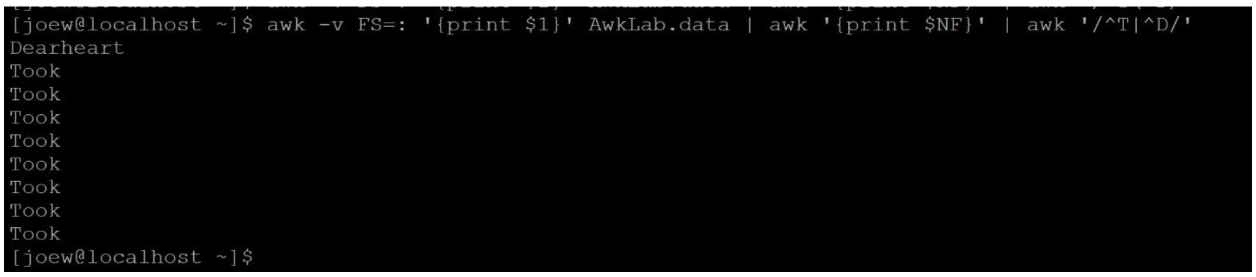
The awk command: *awk -v FS=: ‘/(123)/ {print $1, $2}’ AwkLab.data* print the result of all the names and phone numbers of area code in 123 fellow by the file name “AwkLab.data”. The “FS” is being use as Field Separator and the pattern matching the lines contain the number of 123 representing the area code. The $1 represents the first name and $2 the last name.



```
[joew@localhost ~]$ awk -v FS=: '/(123)/ {print $1, $2}' AwkLab.data
Antoine de Saint-Exupery (123) 978-6432
Belladonna Took (123) 978-5754
Eglantine Took (123) 978-3574
[joew@localhost ~]$
```

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

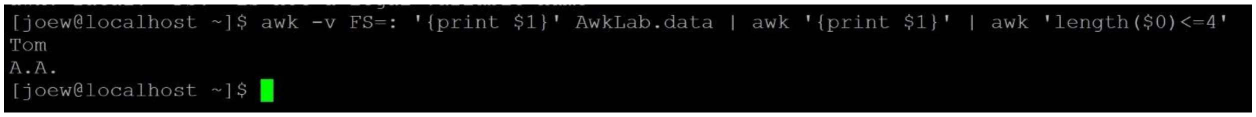
The awk expression: *awk -v FS=: ‘{print $1}’ AwkLab.data | awk ‘{print $NF}’ awk ‘/^T|^D/’ AwkLab.data* will search and print all last names beginning with either a T or D.



```
[joew@localhost ~]$ awk -v FS=: '{print $1}' AwkLab.data | awk '{print $NF}' | awk '/^T|^D/'
Dearheart
Took
Took
Took
Took
Took
Took
Took
Took
[joew@localhost ~]$
```

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

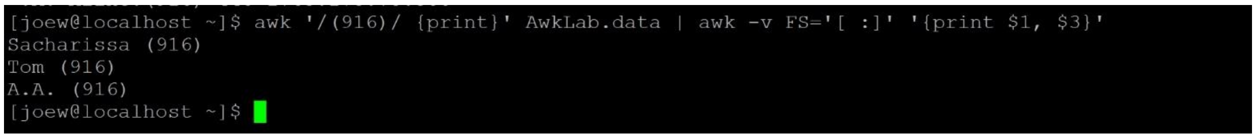
The awk command: _*awk -v FS=: ‘{print $1}’ AwkLab.data awk ‘{print $1}’ | awk ‘Length($0)<=4’*_ will print the first names that has four or less characters. The phrase “{print $1}’ will print the first names.



```
[joew@localhost ~]$ awk -v FS=: '{print $1}' AwkLab.data | awk '{print $1}' | awk 'length($0)<=4'
Tom
A.A.
[joew@localhost ~]$ 
```

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

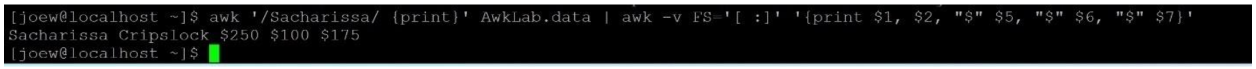
The awk expression: _*awk ‘/(916)/ {print}’ AwkLab.data | awk -v FS=’ [:]’ ‘{print $1, $3}’ AwkLab.data*_ will output the results of the first names and area codes of those in the 916 area code. The sign “|” is a pipe delimiter, which Is following by the filename AwkLab.data.



```
[joew@localhost ~]$ awk '/(916)/ {print}' AwkLab.data | awk -v FS='[ :]' '{print $1, $3}'
Sacharissa (916)
Tom (916)
A.A. (916)
[joew@localhost ~]$ 
```

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

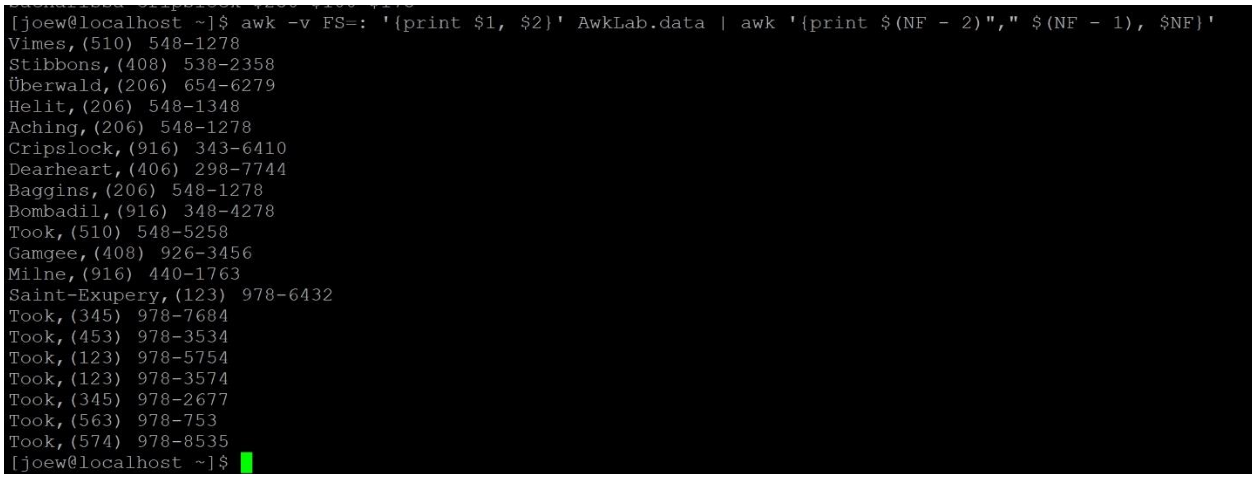
The blow screenshot will print Shacharissa’s campaign contributions using the command: _*awk ‘/Sacharissa/ {print}’ AwkLab.data | awk -v FS=’ [:]’ ‘{print $1, $2, “$” $5, “4” $6, “$” $7}’.*_ The phrase (“) is used several times in this command which indicate as field separator. The $1, $2, $5, $7 will print her contribution.



```
[joew@localhost ~]$ awk '/Sacharissa/ {print}' AwkLab.data | awk -v FS='[ :]' '{print $1, $2, "$" $5, "$" $6, "$" $7}'
Sacharissa Cripslock $250 $100 $175
[joew@localhost ~]$ 
```

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

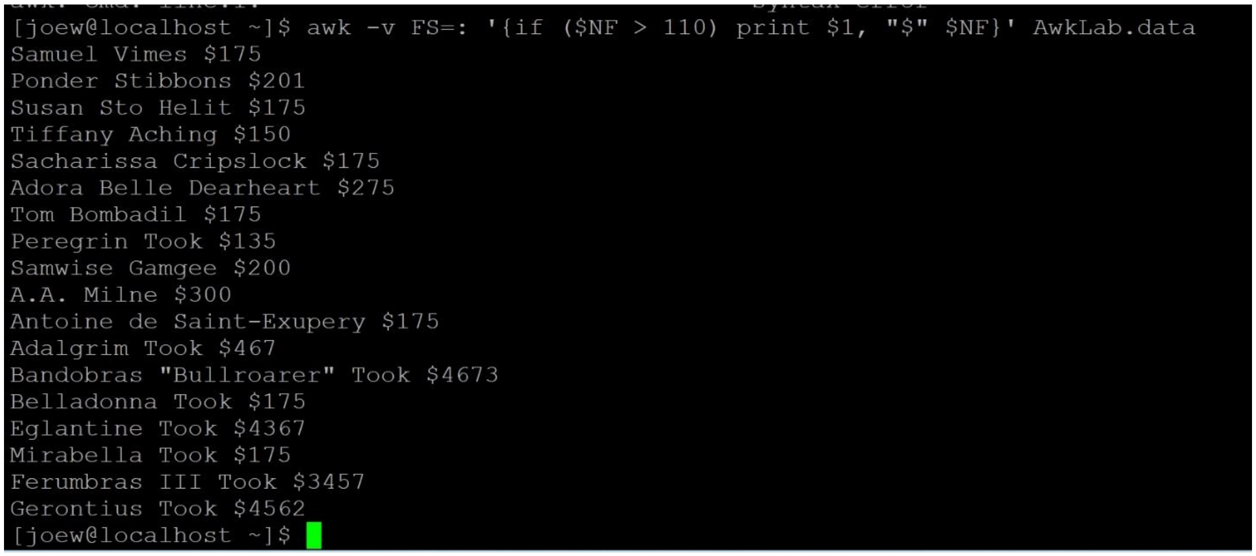
The awk command: _*awk -v FS=: ‘{print $1, $2, AwkLab.data | awk ‘{print – (NF – 2) “,” {NF – 1), $NF}’*_ will print the last names with comma and phone numbers. In this case, we are only printing the last name not the first name, which pipes the output to awk to complete the requirement.



```
[joew@localhost ~]$ awk -v FS=: '{print $1, $2}' AwkLab.data | awk '{print $(NF - 2)"," $(NF - 1), $NF}'
Vimes,(510) 548-1278
Stibbons,(408) 538-2358
Überwald,(206) 654-6279
Helit,(206) 548-1348
Aching,(206) 548-1278
Cripslock,(916) 343-6410
Dearheart,(406) 298-7744
Baggins,(206) 548-1278
Bombadil,(916) 348-4278
Took,(510) 548-5258
Gamgee,(408) 926-3456
Milne,(916) 440-1763
Saint-Exupery,(123) 978-6432
Took,(345) 978-7684
Took,(453) 978-3534
Took,(123) 978-5754
Took,(123) 978-3574
Took,(345) 978-2677
Took,(563) 978-753
Took,(574) 978-8535
[joew@localhost ~]$
```

10. 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.

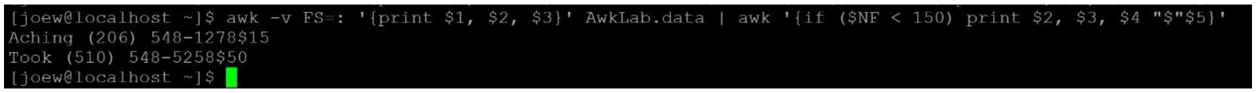
The awk expression: *awk -v FS=: '{if ($NF > 110) print $1, "$" $NF}' AwkLab.data* will print the first and last names of those who contributed more than $110 in the last month. The awk command will search for field that are greater than 110, which will actually print the results along with the names. Additionally, you can also see that "if statement" was used to check the last entry in the line and match to the 110.



```
[joew@localhost ~]$ awk -v FS=: '{if ($NF > 110) print $1, "$" $NF}' AwkLab.data
Samuel Vimes $175
Ponder Stibbons $201
Susan Sto Helit $175
Tiffany Aching $150
Sacharissa Cripslock $175
Adora Belle Dearheart $275
Tom Bombadil $175
Peregrin Took $135
Samwise Gamgee $200
A.A. Milne $300
Antoine de Saint-Exupery $175
Adalgrim Took $467
Bandobras "Bullroarer" Took $4673
Belladonna Took $175
Eglantine Took $4367
Mirabella Took $175
Ferumbras III Took $3457
Gerontius Took $4562
[joew@localhost ~]$
```

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

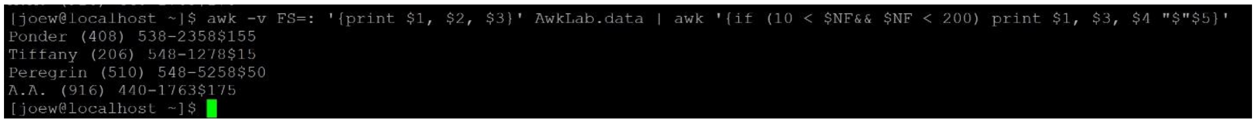
With this awk command: *awk -v FS=: '{print $1, $2 $3}' AwkLab.data | awk '{if ($NF < 150) print $2, $3, $4 "$"$5}'* , we can then print the last names, phone numbers and first month contributions of those who contributed less than $150 in the first month. The awk statement will search for lines that have a $3 field less than150 and print. We can see that the data compelled and separated by a colon, where we print out first names, phone number and contributions.



```
[joew@localhost ~]$ awk -v FS=: '{print $1, $2, $3}' AwkLab.data | awk '{if ($NF < 150) print $2, $3, $4 "$"$5}'
Aching (206) 548-1278$15
Took (510) 548-5258$50
[joew@localhost ~]$
```

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

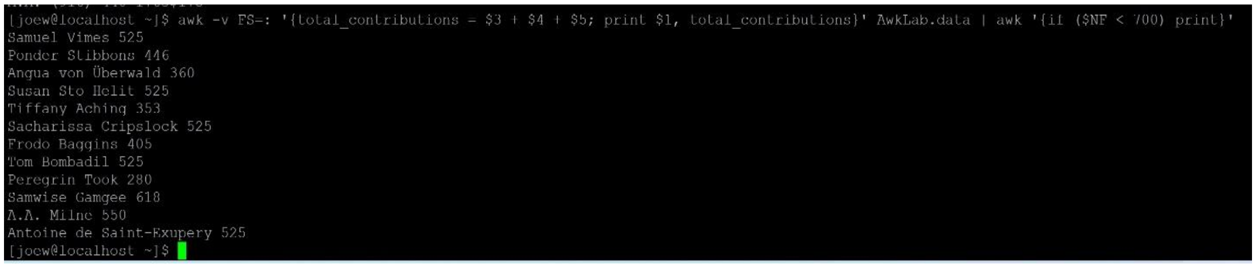
The below awk statement: *<u>awk -v FS=: ‘{print $1, $2, $3}’ AwkLab.data | awk ‘{fi (10 < $NF&& $NF < 200) print $1, $3, $4 “$”5}’</u>* prints the results of first names and contributions of those who donated $10 and $200 in the first month. We also use the if statement in this case to check the entry in the lines and print. The phrases && operator confirm conditionals NF less than 200 and greater than 10.



```
[joew@localhost ~]$ awk -v FS=: '{print $1, $2, $3}' AwkLab.data | awk '{if (10 < $NF&& $NF < 200) print $1, $3, $4 "$"$5}'
Ponder (408) 538-2358$155
Tiffany (206) 548-1278$15
Peregrin (510) 548-5258$50
A.A. (916) 440-1763$175
[joew@localhost ~]$
```

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

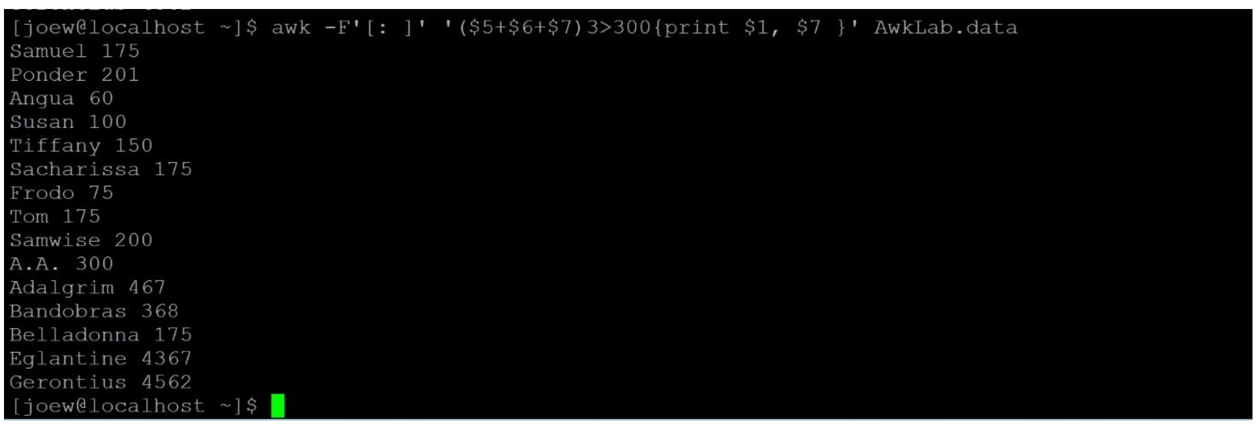
The awk expression: *<u>awk -v FS=: ‘{total_contributions - $3 | $4 | $5; print $1, total_contributions}’ AwkLab.data | awk ‘{if ($NF < 700) print}’</u>* will print first and last names including total contributions of those donated less than 700 over three months. The awk statement print the total line of contributions and then checked to see if the total was less than 700 while it print the specific lines.



```
[joew@localhost ~]$ awk -v FS=: '{total_contributions = $3 + $4 + $5; print $1, total_contributions}' AwkLab.data | awk '{if ($NF < 700) print}'
Samuel Vimes 525
Ponder Stibbons 446
Angua von Überwald 360
Susan Sto Helit 525
Tiffany Aching 353
Sacharissa Cripslock 525
Frodo Baggins 405
Tom Bombadil 525
Peregrin Took 280
Samwise Gamgee 618
A.A. Milne 550
Antoine de Saint-Exupery 525
[joew@localhost ~]$
```

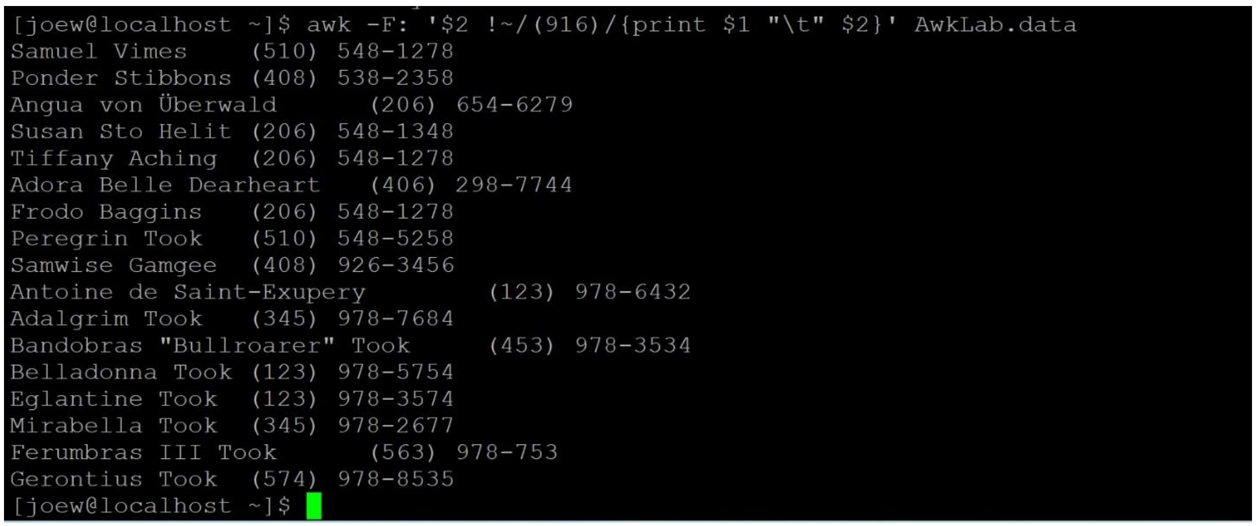
14. Print the first names and first letter of the last name, and average contributions of those who had an average contribution of more then $300.

With the below awk statement: *<u>awk -F' [:]' '($5+$6+$7) 3>300{print $1, $7 }' AwkLab.data</u>*, we print the first names and first letter of the last name, average contribution of those more than 300. The awk expression also establishes the variable average with the values of the months donations and finally average all by 3 to find the average.


```
[joew@localhost ~]$ awk -F'[: ]' '($5+$6+$7)3>300{print $1, $7 }' AwkLab.data
Samuel 175
Ponder 201
Angua 60
Susan 100
Tiffany 150
Sacharissa 175
Frodo 75
Tom 175
Samwise 200
A.A. 300
Adalgrim 467
Bandobras 368
Belladonna 175
Eglantine 4367
Gerontius 4562
[joew@localhost ~]$
```

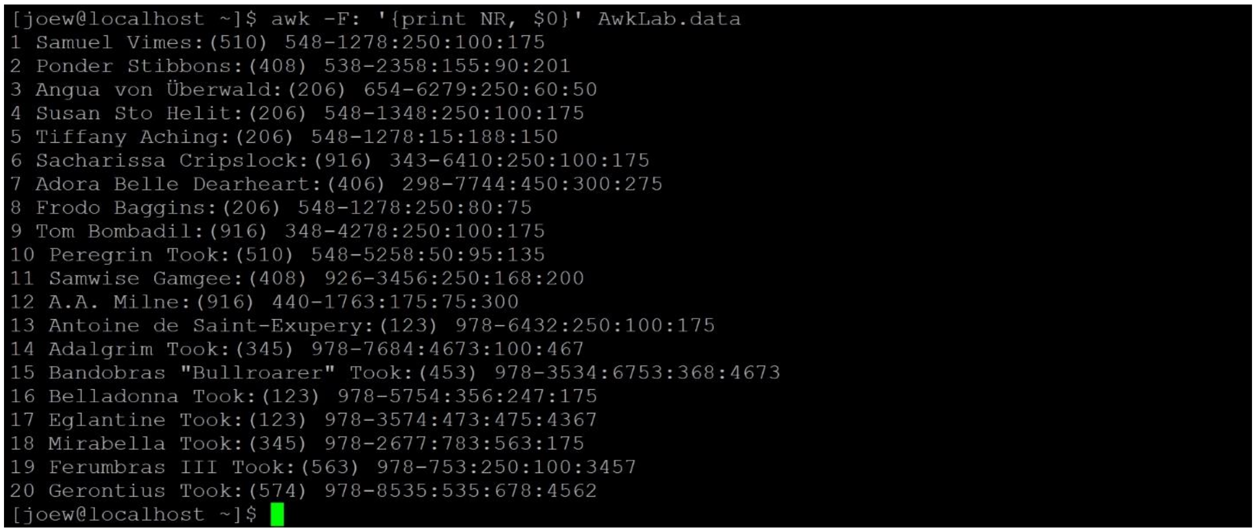
15. Print the last name and area code of those not in the 916 area code.

The awk command: *<u>awk -F: '$2 !~/(916)/{print $1 "\t" $2}' AwkLab.data</u>* prints the last name and area code of those that are not 916. The awk statement matches the lines that are $2 which indicate as phone number that are not 916 and the print the results.


```
[joew@localhost ~]$ awk -F: '$2 !~/(916)/{print $1 "\t" $2}' AwkLab.data
Samuel Vimes    (510) 548-1278
Ponder Stibbons (408) 538-2358
Angua von Überwald      (206) 654-6279
Susan Sto Helit (206) 548-1348
Tiffany Aching  (206) 548-1278
Adora Belle Dearheart   (406) 298-7744
Frodo Baggins   (206) 548-1278
Peregrin Took   (510) 548-5258
Samwise Gamgee  (408) 926-3456
Antoine de Saint-Exupery        (123) 978-6432
Adalgrim Took   (345) 978-7684
Bandobras "Bullroarer" Took     (453) 978-3534
Belladonna Took (123) 978-5754
Eglantine Took  (123) 978-3574
Mirabella Took  (345) 978-2677
Ferumbras III Took      (563) 978-753
Gerontius Took  (574) 978-8535
[joew@localhost ~]$
```

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

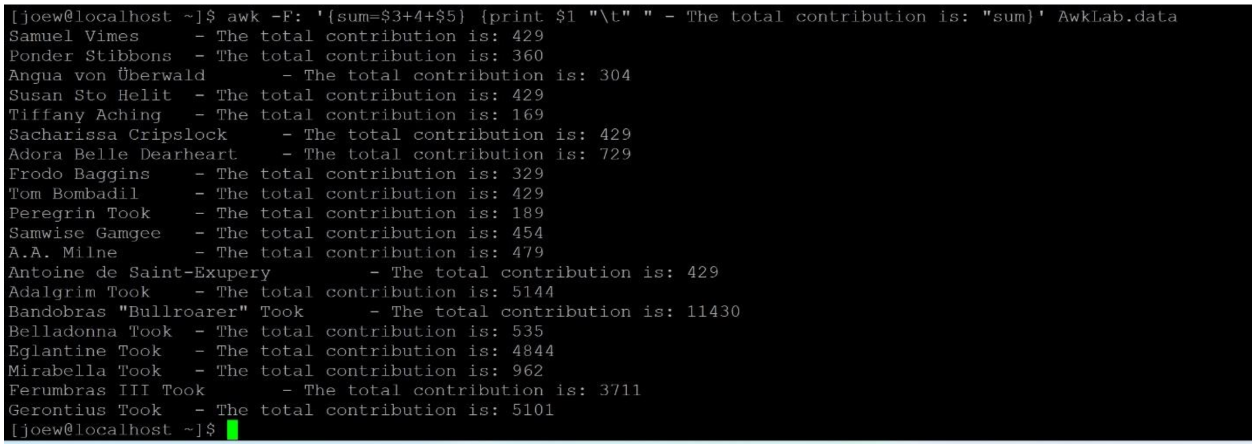
The awk command: *awk -F: ‘{print NR, $0}’ AwkLab.data* will print a unique record along with the number. This awk command will essentially print the entire record of lines with the NR involved. You can see that the entire number is represented by the field $0 and Number of Records NR values and then appends the line at the initial stage.



```
[joew@localhost ~]$ awk -F: '{print NR, $0}' AwkLab.data
1 Samuel Vimes:(510) 548-1278:250:100:175
2 Ponder Stibbons:(408) 538-2358:155:90:201
3 Angua von Überwald:(206) 654-6279:250:60:50
4 Susan Sto Helit:(206) 548-1348:250:100:175
5 Tiffany Aching:(206) 548-1278:15:188:150
6 Sacharissa Cripslock:(916) 343-6410:250:100:175
7 Adora Belle Dearheart:(406) 298-7744:450:300:275
8 Frodo Baggins:(206) 548-1278:250:80:75
9 Tom Bombadil:(916) 348-4278:250:100:175
10 Peregrin Took:(510) 548-5258:50:95:135
11 Samwise Gamgee:(408) 926-3456:250:168:200
12 A.A. Milne:(916) 440-1763:175:75:300
13 Antoine de Saint-Exupery:(123) 978-6432:250:100:175
14 Adalgrim Took:(345) 978-7684:4673:100:467
15 Bandobras "Bullroarer" Took:(453) 978-3534:6753:368:4673
16 Belladonna Took:(123) 978-5754:356:247:175
17 Eglantine Took:(123) 978-3574:473:475:4367
18 Mirabella Took:(345) 978-2677:783:563:175
19 Ferumbras III Took:(563) 978-753:250:100:3457
20 Gerontius Took:(574) 978-8535:535:678:4562
[joew@localhost ~]$
```

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

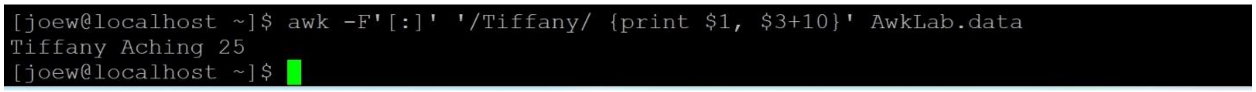
The below awk statement: *awk -F: ‘{sum=$3+4+$5} {print $1 “\t” “ – The total contribution is: “sum}’ AwkLab.data* prints the name and total contribution of each person. The awk statement saved the three month of contributions in the sum and print the sum output in the bracket.



```
[joew@localhost ~]$ awk -F: '{sum=$3+4+$5} {print $1 "\t" " - The total contribution is: "sum}' AwkLab.data
Samuel Vimes    - The total contribution is: 429
Ponder Stibbons - The total contribution is: 360
Angua von Überwald      - The total contribution is: 304
Susan Sto Helit - The total contribution is: 429
Tiffany Aching  - The total contribution is: 169
Sacharissa Cripslock    - The total contribution is: 429
Adora Belle Dearheart   - The total contribution is: 729
Frodo Baggins   - The total contribution is: 329
Tom Bombadil    - The total contribution is: 429
Peregrin Took   - The total contribution is: 189
Samwise Gamgee  - The total contribution is: 454
A.A. Milne      - The total contribution is: 479
Antoine de Saint-Exupery        - The total contribution is: 429
Adalgrim Took   - The total contribution is: 5144
Bandobras "Bullroarer" Took     - The total contribution is: 11430
Belladonna Took - The total contribution is: 535
Eglantine Took  - The total contribution is: 4844
Mirabella Took  - The total contribution is: 962
Ferumbras III Took      - The total contribution is: 3711
Gerontius Took  - The total contribution is: 5101
[joew@localhost ~]$
```

18. Add $10 to Tiffany Aching's first contribution and print her full name and first contribution.

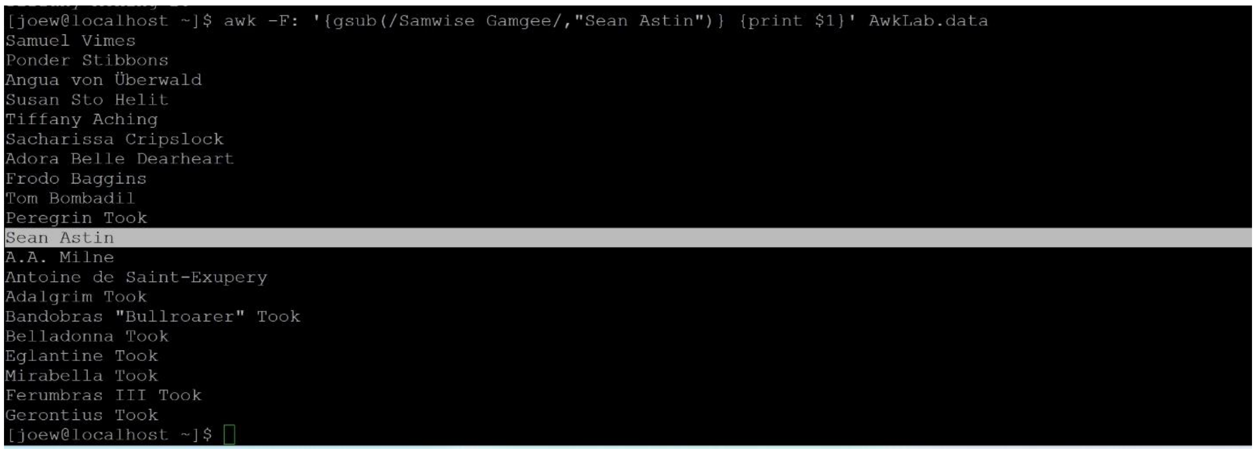
The awk statement below: <u>*awk -F'[:]' '/Tiffany/ {print $1, $3+10}' AwkLab.data*</u> added $10 to Tiffany Aching first contribution and printed the full name and first contribution. The word Tiffany reflected in the command in addition to adding 10+3 of 15 to make it 25.



```
[joew@localhost ~]$ awk -F'[:]' '/Tiffany/ {print $1, $3+10}' AwkLab.data
Tiffany Aching 25
[joew@localhost ~]$ 
```

19. Change Samwise Gamgee's name to Sean Astin.

The awk command: <u>*awk -F: '{gsub(/Samwise Gamgee/,"Sean Astin")} {print $1}' AwkLab.data*</u> changed the Samwise Gamgee to Sean Astin. In the command you will see that "gsub substitute Sean Astin to search for lines matching pattern of Samwise Gamgee. The awk will replaced the new name as showed on the below screenshot.



```
[joew@localhost ~]$ awk -F: '{gsub(/Samwise Gamgee/,"Sean Astin")} {print $1}' AwkLab.data
Samuel Vimes
Ponder Stibbons
Angua von Überwald
Susan Sto Helit
Tiffany Aching
Sacharissa Cripslock
Adora Belle Dearheart
Frodo Baggins
Tom Bombadil
Peregrin Took
Sean Astin
A.A. Milne
Antoine de Saint-Exupery
Adalgrim Took
Bandobras "Bullroarer" Took
Belladonna Took
Eglantine Took
Mirabella Took
Ferumbras III Took
Gerontius Took
[joew@localhost ~]$ 
```

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

    (a) Print first name of the all the Tooks followed by their total campaign contributions.

The created an awk script to print the results of the above requirements.

To create the awk script, you need to type: vi follow by the script file name.

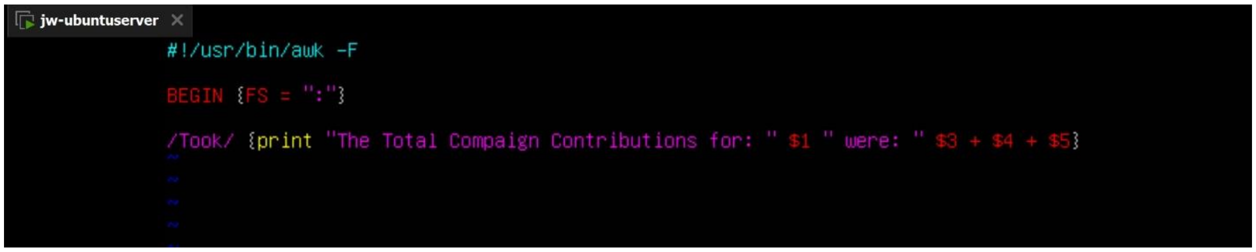
#1/usr/bin/awk -F *(this line of code will run as an awk script in which the system recognized)*

BEGIN {FS = ":"} *(the line indicates that the awk start the file as Field Separator)*

/Took/ {print "The Total Campaign Contribution for: "$1 " were: " $3 + $4 +$5} <u>(this line is the code of the requirement given)</u>

Use wq! to exit and save the file.

The name my awk script that I create is "scripta.awk".

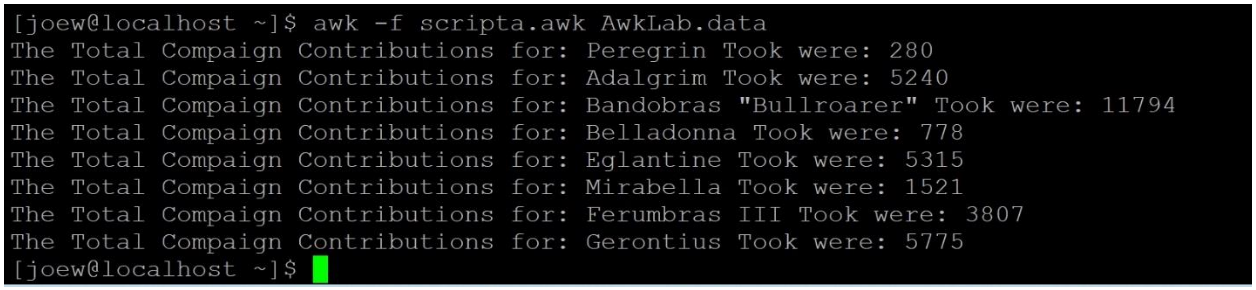


```
#!/usr/bin/awk -F

BEGIN {FS = ":"}

/Took/ {print "The Total Compaign Contributions for: " $1 " were: " $3 + $4 + $5}
~
~
~
~
```

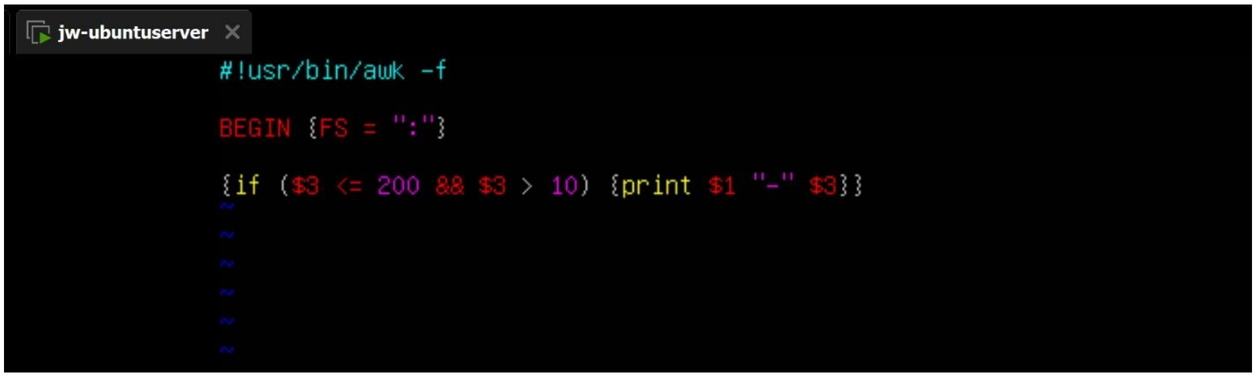
Use the command: *awk -f scripta.awk AwkLab.data* will run the script and print first name of the all the Tooks followed by their total campaign contributions.



```
[joew@localhost ~]$ awk -f scripta.awk AwkLab.data
The Total Compaign Contributions for: Peregrin Took were: 280
The Total Compaign Contributions for: Adalgrim Took were: 5240
The Total Compaign Contributions for: Bandobras "Bullroarer" Took were: 11794
The Total Compaign Contributions for: Belladonna Took were: 778
The Total Compaign Contributions for: Eglantine Took were: 5315
The Total Compaign Contributions for: Mirabella Took were: 1521
The Total Compaign Contributions for: Ferumbras III Took were: 3807
The Total Compaign Contributions for: Gerontius Took were: 5775
[joew@localhost ~]$
```

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

Below is the second awk script create: script.awk. This script will print names and contributions of anyone who donated between 10 and 200 lastly.

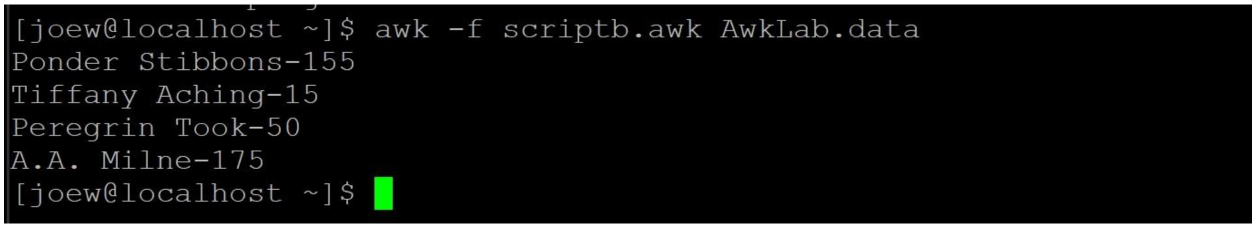


```
#!usr/bin/awk -f

BEGIN {FS = ":"}

{if ($3 <= 200 && $3 > 10) {print $1 "-" $3}}
~
~
~
~
~
~
```

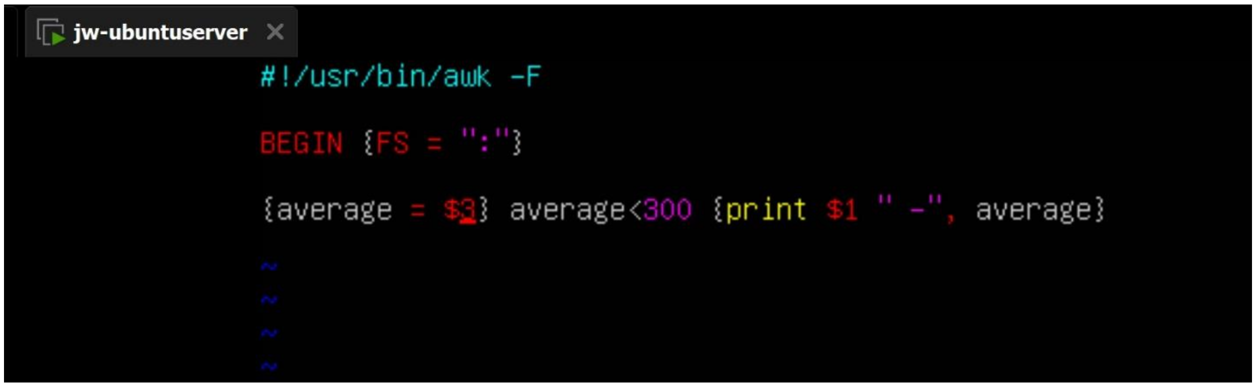
Use the command: *awk -f script.awk AwkLab.data* to run the script



```
[joew@localhost ~]$ awk -f scriptb.awk AwkLab.data
Ponder Stibbons-155
Tiffany Aching-15
Peregrin Took-50
A.A. Milne-175
[joew@localhost ~]$
```

(c) Print the full names and average contribution of those who contributed less than $300 on average.

The third awk script: script.awk will print full names and average contribution of those who contributed less than 300. Please refer to the screenshot.

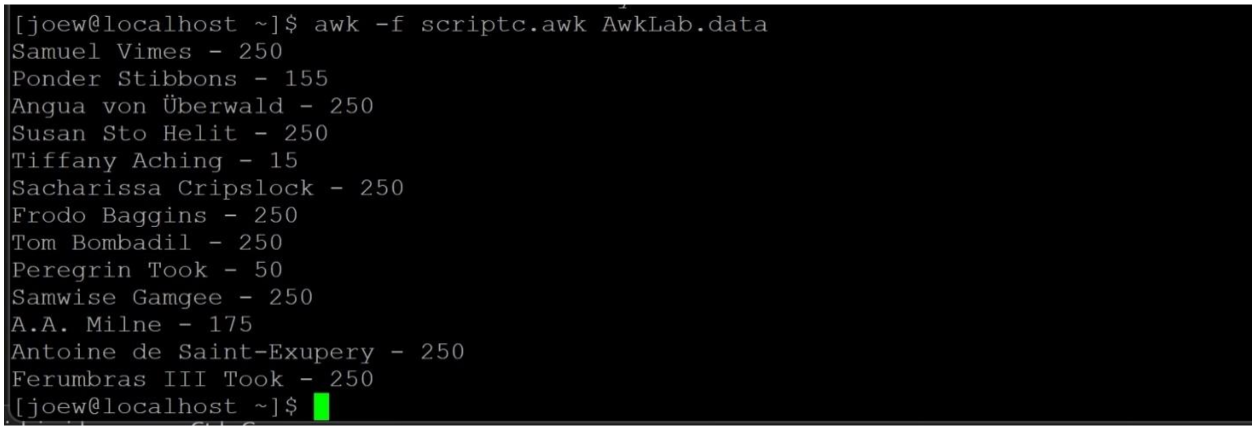


```
#!/usr/bin/awk -F

BEGIN {FS = ":"}

{average = $3} average<300 {print $1 " -", average}
~
~
~
~
```

Use the awk command: *awk. -f script.awk AwkLab.data* to run the script



```
[joew@localhost ~]$ awk -f scriptc.awk AwkLab.data
Samuel Vimes - 250
Ponder Stibbons - 155
Angua von Überwald - 250
Susan Sto Helit - 250
Tiffany Aching - 15
Sacharissa Cripslock - 250
Frodo Baggins - 250
Tom Bombadil - 250
Peregrin Took - 50
Samwise Gamgee - 250
A.A. Milne - 175
Antoine de Saint-Exupery - 250
Ferumbras III Took - 250
[joew@localhost ~]$
```

References:

Learn How to Use Awk Variables, Numeric Expressions and Assignment Operators - Part 8 (tecmint.com)

By Aaron Kili

Executable Scripts (The GNU Awk User’s Guide)

Test if the first character is a number using awk - Stack Overflow

6.9. awk Commands in a Script File | UNIX Shells by Example (4th Edition) (flylib.com)