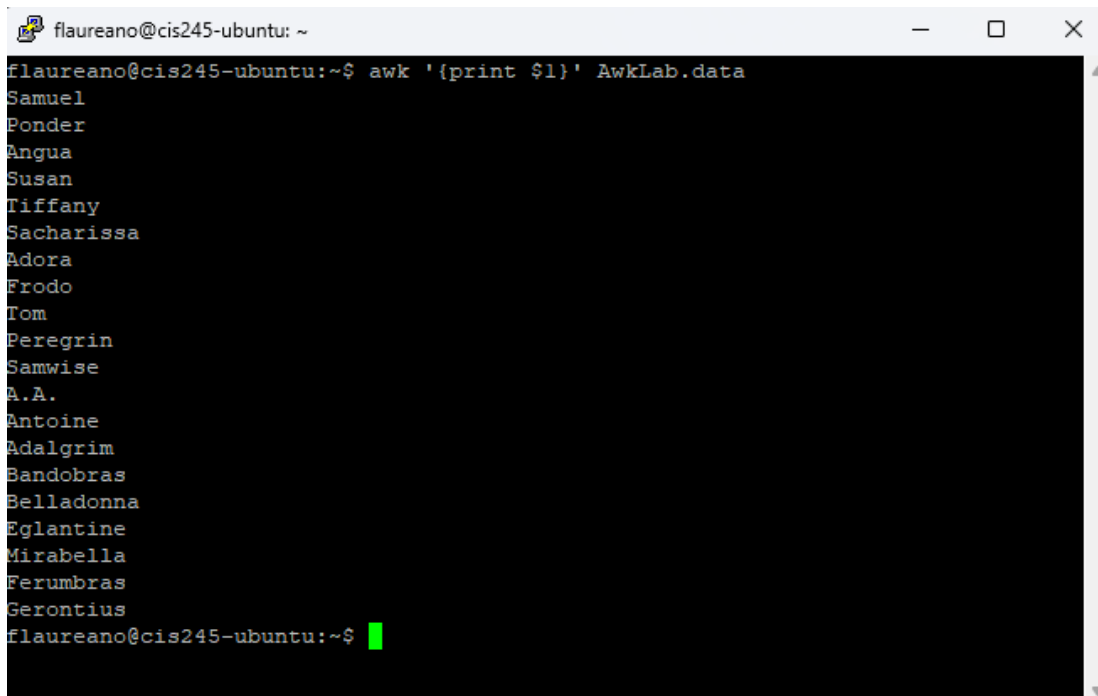


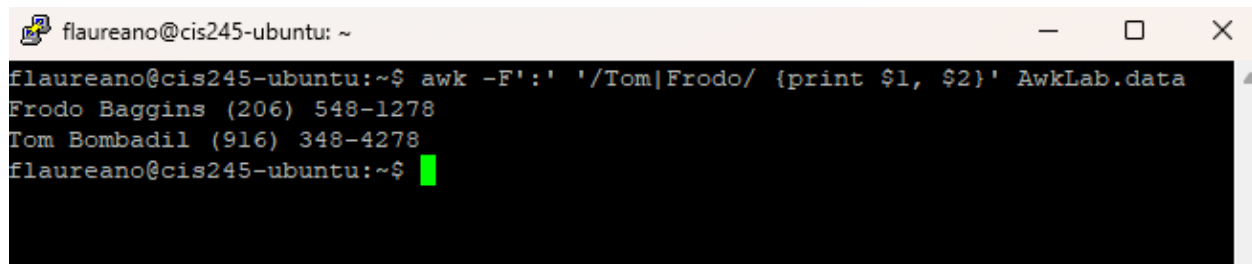
1. Print all the First Names.



```
flaureano@cis245-ubuntu: ~  
flaureano@cis245-ubuntu:~$ awk '{print $1}' AwkLab.data  
Samuel  
Ponder  
Angua  
Susan  
Tiffany  
Sacharissa  
Adora  
Frodo  
Tom  
Peregrin  
Samwise  
A.A.  
Antoine  
Adalgrim  
Bandobras  
Belladonna  
Eglantine  
Mirabella  
Ferumbbras  
Gerontius  
flaureano@cis245-ubuntu:~$
```

1a. Awk handles spaces as breaks in the fields, which is why I'm using '{print \$1}'. The \$ would pull the field of the number next to it. \$1 is the first field, \$2 the second and so on. As the first field for this file is the first name, we use \$1 which in turn provides us with the first names for these users.

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



```
flaureano@cis245-ubuntu: ~  
flaureano@cis245-ubuntu:~$ awk -F':' '/Tom|Frodo/ {print $1, $2}' AwkLab.data  
Frodo Baggins (206) 548-1278  
Tom Bombadil (916) 348-4278  
flaureano@cis245-ubuntu:~$
```

2a. By using the -F option, I can tell awk to set the field separator as colon. /Tom|Frodo/ is stating that I am searching the file specifically for instances where the field Tom or Frodo are present and asking for the first and second fields to be printed (\$1 \$2). The comma after the \$1 is simply to create more clarity as it provides a line break (inserts a space), making the output more readable.

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

```
flaureano@cis245-ubuntu: ~  
flaureano@cis245-ubuntu:~$ awk -F ':' '/Peregrin/ {print $1, $2}' AwkLab.data  
Peregrin Took (510) 548-5258  
flaureano@cis245-ubuntu:~$
```

3a. Similar to question 2, I targeted the first and second fields, separating the fields by a manually set parameter of a colon. The /Peregrin/ entry allows me to focus my search only to lines that contain the text that matches Peregrin, so in this case it only prints out one line. The \$1 and \$2 grabs just the first and second field, discarding the rest of the data in the line which prints just the Full name + phone number.

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

```
flaureano@cis245-ubuntu: ~  
flaureano@cis245-ubuntu:~$ awk '/(123)/ {print}' AwkLab.data  
Antoine de Saint-Exupery:(123) 978-6432:250:100:175  
Belladonna Took:(123) 978-5754:356:247:175  
Eglantine Took:(123) 978-3574:473:475:4367  
flaureano@cis245-ubuntu:~$
```

4a. This time we are targeting all 123 area codes. To achieve this, I filter out the lines that don't contain a 123 area code with /(123)/, which searches for keyword matches and discards any lines that don't contain a match, then print the lines with {print}.

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

```
flaureano@cis245-ubuntu:~$ awk -F ':' '/ T| D/ {print $1}' AwkLab.data  
Adora Belle Dearheart  
Peregrin Took  
Adalgrim Took  
Bandobras "Bullroarer" Took  
Belladonna Took  
Eglantine Took  
Mirabella Took  
Ferumbras III Took  
Gerontius Took  
flaureano@cis245-ubuntu:~$
```

5a. To find names where the last name ends in T or D, you want to define your key word, which in our scenario here it's / T| D/. The spaces before T and D help match the results to the last name. Once our keyword is set, we want to set our field separator to isolate the names away from the phone numbers.

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

```
flaureano@cis245-ubuntu: ~$ awk 'length($1)<=4{print $1}' AwkLab.data
Tom
A.A.
```

6a. The length function takes the numerical value that is passed to it (in this case  $\leq 4$ ) and applies it to the first field (\$1). The function then scans the file and finds and lines that have a first field with 4 or less characters.

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

```
flaureano@cis245-ubuntu: ~$ awk -F'[: ]' '/(916)/{print $1 $3}' AwkLab.data
Sacharissa(916)
Tom(916)
A.A.(916)
flaureano@cis245-ubuntu:~$
```

7a. To find just the first names you need to filter by the specific keyword which in this instance is the area code 916 which we accomplish with `/(916)/`. Then we set our field separators to both an empty space and a colon to filter out all text after the first name, excluding our area code, and all text after our area code which we accomplish with `-F '[: ]'`. Then we print out our results by looking at the two fields we need, \$1 which is the first name and \$3 which is the area code.

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

```
flaureano@cis245-ubuntu: ~$ awk -F':' '/Sacharissa/ {print $1 " " $3 " " $4 " " $5}' AwkLab.data
Sacharissa Cripslock $250 $100 $175
flaureano@cis245-ubuntu:~$
```

8a. To add dollar symbols next to the value for Sacharissa's contributions, we must first filter for lines with her contributions via `/Sacharissa/`. Once found, we set our field separator to `:` via `-F':'` and print fields 1 (\$1 her name), 3 (\$3 for \$250), 4 (\$4 for \$100), and 5 (\$5 for \$175) while attaching a " character to the front of each field as we're adding text in front of it. This allows us to print each field but with the required \$ sign in front.

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

```
flaureano@cis245-ubuntu:~$ awk -F':' '({split($1, name, " "); print name[length(name)] ", " $2 })' AwkLab.data
Vimes, (510) 548-1278
Stibbons, (408) 538-2358
Überwald, (206) 654-6279
Helit, (206) 548-1348
Aching, (206) 548-1278
Cripslock, (916) 343-6410
Dearheart, (406) 298-7744
Baggins, (206) 548-1278
Bombadil, (916) 348-4278
Took, (510) 548-5258
Gamgee, (408) 926-3456
Milne, (916) 440-1763
Saint-Exupery, (123) 978-6432
Took, (345) 978-7684
Took, (453) 978-3534
Took, (123) 978-5754
Took, (123) 978-3574
Took, (345) 978-2677
Took, (563) 978-753
Took, (574) 978-8535
flaureano@cis245-ubuntu:~$
```

9a. To accomplish this task we creation two functions, one splits the first field into an array called name with the strings separated by a space. The other function prints the last string of the name array [length(name)]. Because the string is always the last value in the array, the last name is always printed. After we simply add a comma and print the second field which provides us with the phone number to attach.

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

```
flaureano@cis245-ubuntu: ~
flaureano@cis245-ubuntu:~$ awk -F':' '($5>110){print $1, $5}' AwkLab.data
Samuel Vimes 175
Ponder Stibbons 201
Susan Sto Helit 175
Tiffany Aching 150
Sacharissa Cripslock 175
Adora Belle Dearheart 275
Tom Bombadil 175
Peregrin Took 135
Samwise Gamgee 200
A.A. Milne 300
Antoine de Saint-Exupery 175
Adalgrim Took 467
Bandobras "Bullroarer" Took 4673
Belladonna Took 175
Eglantine Took 4367
Mirabella Took 175
Ferumbras III Took 3457
Gerontius Took 4562
flaureano@cis245-ubuntu:~$
```

10a. To solve this, we must first set our field separator to colon. Once set, we want the last donation which in this case is field 5 (\$5). So we search all lines fifth field for values larger than 110 and if found they will be included in the print output. The output contains \$1 which is the first and last name field and \$5 which is the final donation field.

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

```
flaureano@cis245-ubuntu:~$ awk -F'[: ]' '$5<150{print $2, $5}' AwkLab.data
Aching 15
Took 50
flaureano@cis245-ubuntu:~$
```

11a. To solve this, we must first set our field separators to both colon and a space so that we can split out this data into fields. Once set, we search our fifth field which is the first contribution field (\$5) for values less than 150. Finally, we print the second field which newly contains the last names along with the first contribution value.

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

```
flaureano@cis245-ubuntu:~$ awk -F':' '$3>=10 && $3<=200 {print $1, $3}' AwkLab.data
Ponder Stibbons 155
Tiffany Aching 15
Peregrin Took 50
A.A. Milne 175
flaureano@cis245-ubuntu:~$
```

12a. To solve this we must first set our field separator to colon. Once set, we search all lines where our third field (initial contributions) is both higher than 10 but also lower than 200. To facilitate the and aspect of this query, we included && which represents and. After this, we simply print \$1 which is the name field and \$3 which is the initial donation field.

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

```
flaureano@cis245-ubuntu:~$ awk -F':' '($3 + $4 + $5) < 700 {print $1, $3+$4+$5}' AwkLab.data
Samuel Vimes 525
Ponder Stibbons 446
Angua von Überwald 360
Susan Sto Helit 525
Tiffany Aching 353
Sacharissa Cripslock 525
Frodo Baggins 405
Tom Bombadil 525
Peregrin Took 280
Samwise Gamgee 618
A.A. Milne 550
Antoine de Saint-Exupery 525
flaureano@cis245-ubuntu:~$
```

13a. First set the field separator, then prepare our query where we add \$3+\$4+\$5 into a value for each line which then gets checked if it's less than 700. For printing, we print the initial field (name) then we combine fields 3,4, and 5 into one field to represent the total contributions.

14. 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

```
flaureano@cis245-ubuntu:~$ awk -F':' '($3 + $4 + $5) / 3 > 300 {split($1, name, " "); print name[1], substr(name[2], 1, 1), ($3 + $4 + $5) / 3}' AwkLab.data
Adora B 341.667
Adalgrim T 1746.67
Bandobras " 3931.33
Eglantine T 1771.67
Mirabella T 507
Ferumbas I 1269
Gerontius T 1925
```

14a. First we set the field separator, then combine fields \$3, \$4, \$5 for each line individually, averaging, then determining if that line's average contribution is greater than 300. Then we use the split function to split the first field (full name) into an array called 'name' with two different strings (String 1 is first name & String 2 last name) separated by a space, then printing the first string of the array. Using the substr function, we extract just the first character from the second string (name[2], which is the last name) by specifying the starting position and length.

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

```
flaureano@cis245-ubuntu:~$ awk -F'[(:)]' '($3 != "916") {split($1, name, " "); print name[2], $3}' AwkLab.data
Vimes 510
Stibbons 408
von 206
Sto 206
Aching 206
Belle 406
Baggins 206
Took 510
Gamgee 408
de 123
Took 345
"Bullroarer" 453
Took 123
Took 123
Took 345
III 563
Took 574
flaureano@cis245-ubuntu:~$
```

15a. First we set the field separator which for this case is both colon and parentheses because of the format of the area code, as they are wrapped in (). Then we check the third field (which is the area code field) to make sure that its not a match of 916 (!=). After, we split the first field into an array called name, separated by the space character (" " means the space between characters). As we're using two functions it requires a break so we insert a semi-colon and move onto printing the second string in our name array and 3 characters length (enough for a area code).

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

```
flaureano@cis245-ubuntu:~$ awk '{print NR, $0}' AwkLab.data
1 Samuel Vimes:(510) 548-1278:250:100:175
2 Ponder Stibbons:(408) 538-2358:155:90:201
3 Angua von Überwald:(206) 654-6279:250:60:50
4 Susan Sto Helit:(206) 548-1348:250:100:175
5 Tiffany Aching:(206) 548-1278:15:188:150
6 Sacharissa Cripslock:(916) 343-6410:250:100:175
7 Adora Belle Dearheart:(406) 298-7744:450:300:275
8 Frodo Baggins:(206) 548-1278:250:80:75
9 Tom Bombadil:(916) 348-4278:250:100:175
10 Peregrin Took:(510) 548-5258:50:95:135
11 Samwise Gamgee:(408) 926-3456:250:168:200
12 A.A. Milne:(916) 440-1763:175:75:300
13 Antoine de Saint-Exupery:(123) 978-6432:250:100:175
14 Adalgrim Took:(345) 978-7684:4673:100:467
15 Bandobras "Bullroarer" Took:(453) 978-3534:6753:368:4673
16 Belladonna Took:(123) 978-5754:356:247:175
17 Eglantine Took:(123) 978-3574:473:475:4367
18 Mirabella Took:(345) 978-2677:783:563:175
19 Ferumbras III Took:(563) 978-753:250:100:3457
20 Gerontius Took:(574) 978-8535:535:678:4562
flaureano@cis245-ubuntu:~$
```

16a. To complete this task we first print the number of records (NR) which will list the number of line entries and include a \$0 to print the entire line worth of data. Combining these, you get the number of records + the line data.

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

```
flaureano@cis245-ubuntu:~$ awk -F':' '{print $1, $3+$4+$5 }' AwkLab.data
Samuel Vimes 525
Ponder Stibbons 446
Angua von Überwald 360
Susan Sto Helit 525
Tiffany Aching 353
Sacharissa Cripslock 525
Adora Belle Dearheart 1025
Frodo Baggins 405
Tom Bombadil 525
Peregrin Took 280
Samwise Gamgee 618
A.A. Milne 550
Antoine de Saint-Exupery 525
Adalgrim Took 5240
Bandobras "Bullroarer" Took 11794
Belladonna Took 778
Eglantine Took 5315
Mirabella Took 1521
Ferumbras III Took 3807
Gerontius Took 5775
```

17a. To complete the task we need to set our field separator and print the first field (full name). After we can use fields 3, 4, and 5 to generate our total contributions. To do so, we simply add \$3, \$4, and \$5.

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

```
flaureano@cis245-ubuntu:~$ awk -F':' ' /Tiffany Aching/ {print $1, $3 + 10}' AwkLab.data
Tiffany Aching 25
```

18a. First we set our field separator and search for our keyword /Tiffany Aching/. Then we simply print our first field containing the name of the user (print \$1) and add 10 dollars to her first contribution with \$3+10.

19. Change Samwise Gamgee's name to Sean Astin

```
flaureano@cis245-ubuntu:~$ awk '{sub(/Samwise Gamgee/, "Sean astin"); print}' AwkLab.data
Samuel Vimes:(510) 548-1278:250:100:175
Ponder Stibbons:(408) 538-2358:155:90:201
Angua von Überwald:(206) 654-6279:250:60:50
Susan Sto Helit:(206) 548-1348:250:100:175
Tiffany Aching:(206) 548-1278:15:188:150
Sacharissa Cripslock:(916) 343-6410:250:100:175
Adora Belle Dearheart:(406) 298-7744:450:300:275
Frodo Baggins:(206) 548-1278:250:80:75
Tom Bombadil:(916) 348-4278:250:100:175
Peregrin Took:(510) 548-5258:50:95:135
Sean astin:(408) 926-3456:250:168:200
A.A. Milne:(916) 440-1763:175:75:300
Antoine de Saint-Exupery:(123) 978-6432:250:100:175
Adalgrim Took:(345) 978-7684:4673:100:467
Bandobras "Bullroarer" Took:(453) 978-3534:6753:368:4673
Belladonna Took:(123) 978-5754:356:247:175
Eglantine Took:(123) 978-3574:473:475:4367
Mirabella Took:(345) 978-2677:783:563:175
Ferumbas III Took:(563) 978-753:250:100:3457
Gerontius Took:(574) 978-8535:535:678:4562
flaureano@cis245-ubuntu:~$
```

19a. Using the sub function, we can replace text in a file with the following command {sub/oldtext/, "newtext"}. With this, we can take Samwise Gamgee and replace their name with Sean astin.

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

(b) Print the full names and contributions of anyone who contributed

between \$10 and \$200 in the last contribution

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



```
flaureano@cis245-ubuntu: ~  
GNU nano 7.2 awkscrip  
#1/user/bin/awk -F:  
awk -F':' ' /Took/ {print $1, $3+$4+$5}' AwkLab.data  
awk -F':' ' ($5 > 10 && $5 < 200) {print $1, $3, $4, $5}' AwkLab.data  
awk -F':' ' (if (($3+$4+$5)/3 < 300) print $1, (($3+$4+$5)/3))' AwkLab.data
```

20a. The first line searches for any tooks and then prints the first field (their name) and then combine their contributions ( $\$3+\$4+\$5$ ) into one value.

20b. The second line prints the last contribution (signified by  $\$5$ ) if its greater than 10 but less than 200. It then prints the first field (name), third fourth and fifth fields which are all of their contributions. Something that is key is to make sure that you segment the math here properly because you don't want, for example, to have  $\$5$ 's value to bleed into being divided too early because you made a mistake with parentheses.

20c. This entry checks the lines and sees IF any of the contributions combined, then divided are less than 300. For this, the if statement lays out a condition that if met, the printing of the first field ( $\$1$ ) will occur. Because there is no else if, either the condition is met or the print never occurs.

### Work Cited

<https://www.computerworld.com/video/509010/how-to-use-the-awk-command-2-minute-linux-tips.html>

<https://www.youtube.com/watch?v=9YOZml-zWok>

<https://www.grymoire.com/Unix/Awk.html#uh-45>

[https://www.gnu.org/software/gawk/manual/html\\_node/Options.html](https://www.gnu.org/software/gawk/manual/html_node/Options.html)

<https://flylib.com/books/en/4.356.1.52/1/>

<https://www.aholdengouveia.name/LinuxAdmin/Awk.html>