# Lecture 5A: Filter Utilities

## Introduction

We have a lot of ways of working with data using the shell. Let's introduce a few important utilities that we will be using.

But first our sample data:

```
cat fsoss-example
```

```
  Brauer,Eric, volunteer,0.00, regular,M
De Boer,Vincent,attendee,50.00,regular,L
de Boer,Vincent,speaker,0.00,regular,L
Hong,Bruce,attendee,50.00, gluten free,S
Hernandez,Mark,speaker,0.00, regular,M
hernandez,mark,VIP,0.00, regular,m
Siegel,Les,contributor,250.00,vegetarian,L
Toombs,Laura,attendee,50.00,vegan,S
Patel,Ami,speaker,0.00,vegan,S
PATEL,AMI, CONTRIBUTOR,250.00,VEGAN,S
Wentz,Markus,volunteer,0.00,vegetarian,XXL
Leung,Bruce,volunteer,0.00,regular,M
hernandez,mark,VIP,0.00, regular,L
Johnson,Michael,speaker,0.00,beef & yogurt,L
McBoatface,Boaty,attendee,50.00,diesel,XXXXL
Smith,Tyler,speaker,0.00,regular,L
Liu,Glaser,volunteer,0.00,vegetarian,M
kim,jaeHyun,attendee,50.00,regular,M
```

Every year Seneca hosts a conference on Free Software. This event brings in hundreds of people from industry and academia, as well as students and enthusiasts.

This sample data is based on conference information we receive from an online registration web form. You can see that we have some regular attendees who have paid $50 to attend the conference, as well special contributors who pay a little more to show their support. Volunteers and Speakers get in free. In addition, we need to make note of people's dietary requirements and T-shirt size. Also note that some names are repeated. Some people have more than one role, or they may have forgotten that they already registered.

## How Many Registrations Have There Been?

Let's count how many times people have registered. Each registration is one line. To count things, we use `wc` .

```
wc fsoss-example
```

```
18  32 694 fsoss-example
```

The numbers we get are number of lines, number of words, number of characters. To get *only the number of lines*, use:

```
 wc -l fsoss-example
```

```
18 fsoss-example
```

So only 18 registrations so far… but assume that this file could be hundreds of lines long.

---

## Which Registrants Are Vegetarians?

What we need to do here is to print only the lines that contain the word ‘vegetarian.’ We have a fantastic tool called `grep` which will return lines that contain a search pattern, sort of like using Ctrl+F in your browser.

```
 grep "vegetarian" fsoss-example
```

```
 Siegel,Les,contributor,250.00,vegetarian,L
Wentz,Markus,volunteer,0.00,vegetarian,XXL
Liu,Glaser,volunteer,0.00,vegetarian,M
```

Note that the `-i` option is used to ignore case. For example:

```
 grep -i "contributor" fsoss-example
```

```
 Siegel,Les,contributor,250.00,vegetarian,L
PATEL,AMI, CONTRIBUTOR,250.00,VEGAN,S
```

---

## How Many Vegetarians Are There?

Instead of printing a list of names, we need to get a *number* of vegetarians so that we can organize meals during the conference. We know that `grep` will return a list of lines containing the word ‘vegetarian’, and we know that `wc -l` will return a number of lines. What we need is a way of connecting the *output* of a grep command into the *input* of a wc command.

## Introducing Pipes

The **|** symbol indicates a *pipe*. A pipe connects commands together in a simple way. Let’s try it with our example.

```
 grep "vegetarian" fsoss-example | wc -l
```

```
3
```

The lines we are counting here are from the output of the grep command, not from the file! Piping is incredibly useful. We should use similar commands to count the other types of meals required at our conference.

---

## Preparing Name Tags

In order to print name tags, we require people's names and their status in a csv file. We want to omit the money they spent, their dietary requirements and their shirt sizes since some people might be sensitive about that type of information.

With the `cut` command we can specify which *fields* we want to display. We will also need to specify a *delimiter*. The comma (,) is used to distinguish one field from another. We want to print fields 1-3.

```
cut -f 1-3 -d',' fsoss-example
```

```
 Brauer,Eric, volunteer
De Boer,Vincent,attendee
de Boer,Vincent,speaker
Hong,Bruce,attendee
Hernandez,Mark,speaker
hernandez,mark,VIP
Siegel,Les,contributor
Toombs,Laura,attendee
Patel,Ami,speaker
PATEL,AMI, CONTRIBUTOR
Wentz,Markus,volunteer
Leung,Bruce,volunteer
hernandez,mark,VIP
Johnson,Michael,speaker
McBoatface,Boaty,attendee
Smith,Tyler,speaker
Liu,Glaser,volunteer
kim,jaeHyun,attendee
```

Let's also use a redirection to put this into a separate file:

```
cut -f 1-3 -d',' fsoss-example > nametags.csv
```

## Sorting The List

Right now it kind of seems like the list of registrations has no particular order. Let's put it into alphabetical order. We can use the `sort` command for this.

```
sort fsoss-example
```

```
 Brauer,Eric, volunteer,0.00, regular,M
De Boer,Vincent,attendee,50.00,regular,L
de Boer,Vincent,speaker,0.00,regular,L
Hernandez,Mark,speaker,0.00, regular,M
hernandez,mark,VIP,0.00, regular,L
hernandez,mark,VIP,0.00, regular,m
Hong,Bruce,attendee,50.00, gluten free,S
Johnson,Michael,speaker,0.00,beef & yogurt,L
kim,jaeHyun,attendee,50.00,regular,M
Leung,Bruce,volunteer,0.00,regular,M
Liu,Glaser,volunteer,0.00,vegetarian,M
McBoatface,Boaty,attendee,50.00,diesel,XXXXL
PATEL,AMI, CONTRIBUTOR,250.00,VEGAN,S
Patel,Ami,speaker,0.00,vegan,S
```

```
Siegel,Les,contributor,250.00,vegetarian,L
Smith,Tyler,speaker,0.00,regular,L
Toombs,Laura,attendee,50.00,vegan,S
Wentz,Markus,volunteer,0.00,vegetarian,XXL
```

By default, the `sort` command will sort by alphabetical order, starting from the beginning of the line to the end. Depending on the version of `sort`, you might find that Upper and Lower case are separated.

**Note:** If your output doesn't match this, try using `sort -f` to ignore case.

Let's also give each registration a number, something that we can use for a primary key if we have to. `cat` can provide us with line numbers with the `-n` option.

```
sort -f fsoss-example | cat -n
```

```
     1  Brauer,Eric, volunteer,0.00, regular,M
    2  De Boer,Vincent,attendee,50.00,regular,L
    3  de Boer,Vincent,speaker,0.00,regular,L
    4  Hernandez,Mark,speaker,0.00, regular,M
    5  hernandez,mark,VIP,0.00, regular,L
    6  hernandez,mark,VIP,0.00, regular,m
    7  Hong,Bruce,attendee,50.00, gluten free,S
    8  Johnson,Michael,speaker,0.00,beef & yogurt,L
    9  kim,jaeHyun,attendee,50.00,regular,M
   10  Leung,Bruce,volunteer,0.00,regular,M
   11  Liu,Glaser,volunteer,0.00,vegetarian,M
   12  McBoatface,Boaty,attendee,50.00,diesel,XXXXL
   13  PATEL,AMI, CONTRIBUTOR,250.00,VEGAN,S
   14  Patel,Ami,speaker,0.00,vegan,S
   15  Siegel,Les,contributor,250.00,vegetarian,L
   16  Smith,Tyler,speaker,0.00,regular,L
   17  Toombs,Laura,attendee,50.00,vegan,S
   18  Wentz,Markus,volunteer,0.00,vegetarian,XXL
```

## Who Gets In First?

Let's sort by the amount of money that people paid, from highest to lowest. That way we can let those people in first. `sort` can do this. We can use `-k` to specify a *key*, `-nr` to do a numerical sort in reverse order, and `-t','` to specify the comma as our delimiter. Why does `cut` use `-d` and `sort` uses `-t` ? I don't know ask the programmers.

```
sort -t',' -nr -k4 fsoss-example
```

```
 Siegel,Les,contributor,250.00,vegetarian,L
PATEL,AMI, CONTRIBUTOR,250.00,VEGAN,S
Toombs,Laura,attendee,50.00,vegan,S
McBoatface,Boaty,attendee,50.00,diesel,XXXXL
kim,jaeHyun,attendee,50.00,regular,M
Hong,Bruce,attendee,50.00, gluten free,S
De Boer,Vincent,attendee,50.00,regular,L
Wentz,Markus,volunteer,0.00,vegetarian,XXL
Smith,Tyler,speaker,0.00,regular,L
Patel,Ami,speaker,0.00,vegan,S
Liu,Glaser,volunteer,0.00,vegetarian,M
Leung,Bruce,volunteer,0.00,regular,M
```

```
Johnson,Michael,speaker,0.00,beef & yogurt,L
hernandez,mark,VIP,0.00, regular,m
hernandez,mark,VIP,0.00, regular,L
Hernandez,Mark,speaker,0.00, regular,M
de Boer,Vincent,speaker,0.00,regular,L
Brauer,Eric, volunteer,0.00, regular,M
```

## Who Wins The Raffle?

Each registration earns the registrant a chance to win the big raffle. Fortunately we can also do this using `sort`. Using `-R` will sort the lines in a *random* order. Then we can use `head` to specify the first name on the list.

```
sort -R fsoss-example | head -1
```

```
Patel,Ami,speaker,0.00,vegan,S
```

Congratulations, Ami! You win!

## How Many People Will Be Attending?

Let’s remove all the repeated names in our list, so we get a clearer picture about people who are attending. `uniq` will print only unique lines, that is, it will remove the second instance of a line if they are the same and next to each other.

```
sort -f fsoss-example | uniq
```

```
  Brauer,Eric, volunteer,0.00, regular,M
De Boer,Vincent,attendee,50.00,regular,L
de Boer,Vincent,speaker,0.00,regular,L
Hernandez,Mark,speaker,0.00, regular,M
hernandez,mark,VIP,0.00, regular,m
Hong,Bruce,attendee,50.00, gluten free,S
Johnson,Michael,speaker,0.00,beef & yogurt,L
kim,jaeHyun,attendee,50.00,regular,M
Leung,Bruce,volunteer,0.00,regular,M
Liu,Glaser,volunteer,0.00,vegetarian,M
McBoatface,Boaty,attendee,50.00,diesel,XXXXL
PATEL,AMI, CONTRIBUTOR,250.00,VEGAN,S
Patel,Ami,speaker,0.00,vegan,S
Siegel,Les,contributor,250.00,vegetarian,L
Smith,Tyler,speaker,0.00,regular,L
Toombs,Laura,attendee,50.00,vegan,S
Wentz,Markus,volunteer,0.00,vegetarian,XXL
```

This only removed one line, the second instance of ‘hernandez,mark’. Let’s use `cut` to specify only the first and last names.

```
sort -f fsoss-example | cut -d',' -f1,2 | uniq
```

```
  Brauer,Eric
De Boer,Vincent
de Boer,Vincent
Hernandez,Mark
hernandez,mark
```

```
Hong,Bruce
Johnson,Michael
kim,jaeHyun
Leung,Bruce
Liu,Glaser
McBoatface,Boaty
PATEL,AMI
Patel,Ami
Siegel,Les
Smith,Tyler
Toombs,Laura
Wentz,Markus
```

Closer, but not good enough. Since we got our list from a web form, it isn't very "clean". Some people left their Caps Lock on, some didn't capitalise their name. We can `tr` to `translate` letters upper case or lower case only. I'll choose upper case.

**Note**: `tr` can't take filenames the same way our other utilities can. (Why not? Ask the programmer, etc.). One way around that is to use `cat`, then pipe that into `tr`, then everything else.

For example:

`cat fsoss-example | tr 'a-z' 'A-Z'` OR

`tr 'a-z' 'A-Z' < fsoss-example`

```
 BRAUER,ERIC, VOLUNTEER,0.00, REGULAR,M
DE BOER,VINCENT,ATTENDEE,50.00,REGULAR,L
DE BOER,VINCENT,SPEAKER,0.00,REGULAR,L
HONG,BRUCE,ATTENDEE,50.00, GLUTEN FREE,S
HERNANDEZ,MARK,SPEAKER,0.00, REGULAR,M
HERNANDEZ,MARK,VIP,0.00, REGULAR,M
SIEGEL,LES,CONTRIBUTOR,250.00,VEGETARIAN,L
TOOMBS,LAURA,ATTENDEE,50.00,VEGAN,S
PATEL,AMI,SPEAKER,0.00,VEGAN,S
PATEL,AMI, CONTRIBUTOR,250.00,VEGAN,S
WENTZ,MARKUS,VOLUNTEER,0.00,VEGETARIAN,XXL
LEUNG,BRUCE,VOLUNTEER,0.00,REGULAR,M
HERNANDEZ,MARK,VIP,0.00, REGULAR,M
JOHNSON,MICHAEL,SPEAKER,0.00,BEEF & YOGURT,L
MCBOATFACE,BOATY,ATTENDEE,50.00,DIESEL,XXXXL
SMITH,TYLER,SPEAKER,0.00,REGULAR,L
LIU,GLASER,VOLUNTEER,0.00,VEGETARIAN,M
KIM,JAEHYUN,ATTENDEE,50.00,REGULAR,M
```

Behold! This is an example STDIN redirection.

Now to plus this into our chain:

`sort -f fsoss-example | tr 'a-z' 'A-Z' | cut -d',' -f1,2 | uniq`

```
 BRAUER,ERIC
DE BOER,VINCENT
HERNANDEZ,MARK
HONG,BRUCE
JOHNSON,MICHAEL
KIM,JAEHYUN
LEUNG,BRUCE
```

```
LIU,GLASER
MCBOATFACE,BOATY
PATEL,AMI
SIEGEL,LES
SMITH,TYLER
TOOMBS,LAURA
WENTZ,MARKUS
```

And finally, use `wc -l` to count the unique visitors to the conference.

---

# Summary

## Filter Commands

- `head <-#>` : **Prints number of lines from the beginning of a file, specified by ‘#’. Default is 10.**
- `tail <-#>` : **Prints number of lines from the end of a file, specified by ‘#’. Default is 10.**
- `uniq` : Prints only unique lines. Use after `sort` .
- `wc` : Word Count. Counts lines, words, or characters. Default is all three.
  - `-l` : **count lines**
  - `-w` : count words (delimited by whitespace)
  - `-m` : count characters
  - `-c` : count bytes
- `tr` ‘X’ ‘Y’: Translates any instance of X into Y. Here are some examples:
  - `tr ‘a-z’ ‘A-Z’: **Converts lower case to upper case.**
  - `tr -s ' ' '\t'` : Converts one or more spaces into only one tab. `-s` means ‘squeeze SET1’.
- `cut` : Selects fields or characters from files or standard input. Default delimiter is the *tab*. Here’s a complete list of how we can use `cut` :
  - 1–10: first 10
  - 3–8: 3rd to 8th
  - –10: up to 10th
  - 2–: from 2nd until the end of line
  - 1–3,4,10–: combination of above
  - `-c` : cut characters
  - `-f` : **cut fields**
  - `-d"X"` : **specify a delimiter (X)**
- `sort` : Sorts single files or standard input. This will also merge and sort multiple files.
  - `-f` : **ignore case in comparisons**
  - `-n` : **numeric sort**
  - `-h` : human readable sort, such as file sizes (10M, 1G, 4k)
  - `-u` : display unique entries
  - `-r` : **reverse sort**
  - `-k` : **specify a key to use to sort. This works the same way as the `-f` in the `cut` command, but the default delimiter here is the space.**

   - `-t` : **specify a delimiter**

**Note**: You can access fsoss-example to help you practice. Use this command:

```
cp ~eric.brauer/uli101/fsoss-example ~
```