Task A

1.
```
Edward Priyatna@LAPTOP-GG6FAS2B ~/HW3
$ ls -lh
total 4.6G
-rwxr-xr-x 1 Edward Priyatna None 2.3G Oct 18 13:04 dataset_TIST2015.tar
-rwxr-xr-x 1 Edward Priyatna None 2.1G Oct  6 18:53 dataset_TIST2015_Checkins_v2.txt
-rwxr-xr-x 1 Edward Priyatna None  25K Aug 12  2015 dataset_TIST2015_Cities.txt
-rwxr-xr-x 1 Edward Priyatna None 222M Aug 12  2015 dataset_TIST2015_POIs.txt
-rwxr-xr-x 1 Edward Priyatna None 2.0K Oct  6 18:59 dataset_TIST2015_readme_v2.txt
```
First unzip the file

ls-lh #ls list name of file, lh tells size of file

The size of the data files are 2.1GB, 25KB, 222MB and 2.0KB

2.
```
user_id venue_id        UTC_time        timezone_offset
50756   4f5e3a72e4b053fd6a4313f6        Tue Apr 03 18:00:06 +0000 2012  240
190571  4b4b87b5f964a5204a9f26e3        Tue Apr 03 18:00:07 +0000 2012  180
221021  4a85b1b3f964a520eefe1fe3        Tue Apr 03 18:00:08 +0000 2012  -240
66981   4b4606f2f964a520751426e3        Tue Apr 03 18:00:08 +0000 2012  -300
21010   4c2b4e8a9a559c74832f0de2        Tue Apr 03 18:00:09 +0000 2012  240
28761   4b4bade2f964a520cfa326e3        Tue Apr 03 18:00:09 +0000 2012  -240
39350   49bbd6c0f964a520f4531fe3        Tue Apr 03 18:00:09 +0000 2012  -240
1446    4e88cf4ed22d53877981fdab        Tue Apr 03 18:00:09 +0000 2012  -300
82296   4dfc825bc65b31579b2e7679        Tue Apr 03 18:00:11 +0000 2012  180
```
head dataset_TIST2015_Checkins_v2.txt | less #pipe into less to see the file

Then press /tab button

Tabs are used and there are 4 columns

3.
```
Edward Priyatna@LAPTOP-GG6FAS2B ~/HW3
$ head -1 dataset_TIST2015_Checkins_v2.txt
user_id venue_id        UTC_time        timezone_offset
```
head -1 dataset_TIST2015_Checkins_v2.txt #see the header

the column names are user_id, venue_id, UTC_time and timezone_offset

4.
```
Edward Priyatna@LAPTOP-GG6FAS2B ~/HW3_1
$ sort dataset_TIST2015_Checkins_v2.txt | uniq | wc -l
33253305
```
sort dataset_TIST2015_Checkins_v2.txt | uniq | wc -l #use this because there are some duplicate rows

use uniq to find unique value. Count how many unique value with wc -l.

```
Edward Priyatna@LAPTOP-GG6FAS2B ~/HW3_1
$ cut -f 1 dataset_TIST2015_Checkins_v2.txt | sort | uniq | wc -l
266910
```
cut -f 1 dataset_TIST2015_Checkins_v2.txt | sort | uniq | wc -l

#cut to column 1 user id then sort them then use uniq to find unique value. Count how many unique value with wc -l.

There are 266910 unique user id (including the header), so there are 266909 user id.

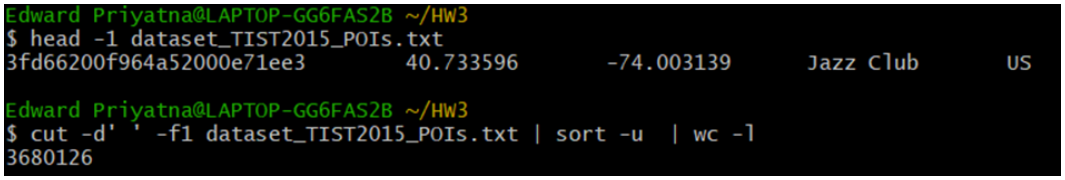
5.
```
Edward Priyatna@LAPTOP-GG6FAS2B ~/HW3
$ head -2 dataset_TIST2015_Checkins_v2.txt
user_id venue_id        UTC_time        timezone_offset
50756   4f5e3a72e4b053fd6a4313f6        Tue Apr 03 18:00:06 +0000 2012  240
```
head -2 dataset_TIST2015_Checkins_v2.txt

#see the head which is beginning of the file. -2 because -1 is header.

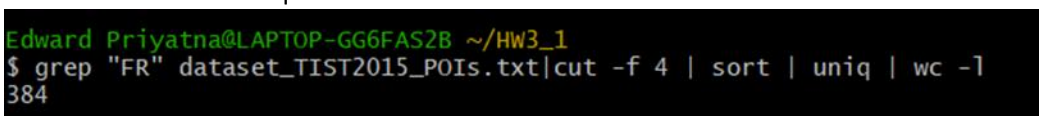
April 3 2012 is the first date

```
Edward Priyatna@LAPTOP-GG6FAS2B ~/HW3
$ tail -1 dataset_TIST2015_Checkins_v2.txt
22704   50df4ee5e4b0c48b5a1c2968        Mon Sep 16 23:24:15 +0000 2013  180
```
tail -1 dataset_TIST2015_Checkins_v2.txt #see the tail which is end of file

September 16 2013

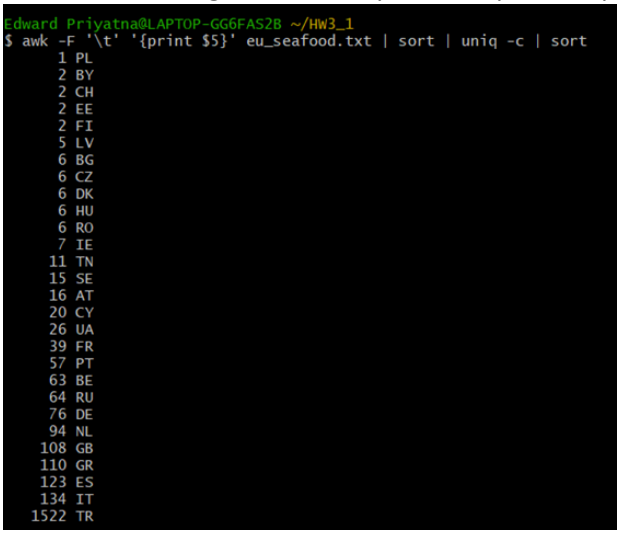
6. 

Head -1 dataset_TIST2015_POIs.txt #see the file so we can know which column is venue id
Cut -d' ' -f1 dataset_TIST2015_POIs.txt | sort -u | wc -l
#cut to column 1 then sort the unique value then count how many line are there
There are 3680126 unique venue ID.

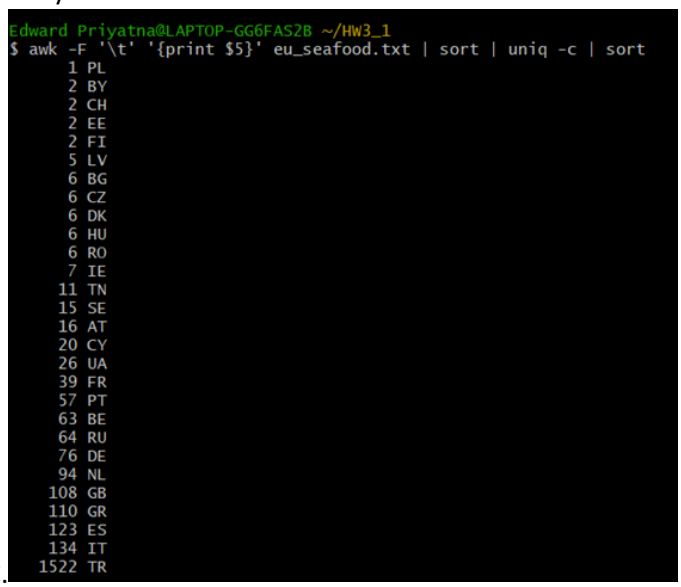
7. 

grep "FR" dataset_TIST2015_POIs.txt|cut -f 4 | sort | uniq | wc -l
#search for country code FR, then cut to column 4, then sort the values, then uniq, then count how many lines are there
There are 384 unique venue categories in France.

8. A. awk -F "\t" '{ if($2>=35 && $2<=72 && $3>=-25 && $3<=45) {print}}' dataset_TIST2015_POIs.txt > POIeu.txt
#use awk, file is delimited by tab. Column 2 between 35 to 72. Column 3 between -25 to 75. Because that is the latitude and longitude of Europe. Then {print} to print all the values. > to write the data to a new txt.

B. 

awk -F '\t' '{print $5}' POIeu.txt | sort | uniq -c | sort
#use awk, file is delimited by tab. Print column 5. Pipe the result to sort, sort the value. Pipe to uniq -c count frequency of each unique value. Then pipe to sort to be sorted.
Turkey has the most venues

C. 

grep -w 'Seafood Restaurant' POIeu.txt > eu_seafood.txt #grep -w search for ‘Seafood Restaurant’. Then > put it to eu_seafood.txt
awk -F '\t' '{print $5}' eu_seafood.txt | sort | uniq -c |sort
use awk, file is delimited by tab. Print column 5 Pipe the result to sort, sort the value. Pipe to uniq -c count frequency of each unique value. Then pipe to sort to be sorted.

```
Edward Priyatna@LAPTOP-GG6FAS2B ~/HW3_1
$ awk -F '\t' '{print $4}' eu_restaurant.txt | sort | uniq -c | sort
     23 Filipino Restaurant
     27 Mongolian Restaurant
     37 Peruvian Restaurant
     51 Gluten-free Restaurant
     53 Malaysian Restaurant
     54 New American Restaurant
     57 Southern / Soul Food Restaurant
     67 Australian Restaurant
     67 Indonesian Restaurant
     72 Cajun / Creole Restaurant
     77 Ethiopian Restaurant
     89 South American Restaurant
     95 Cuban Restaurant
     95 Latin American Restaurant
     96 Dim Sum Restaurant
    126 Molecular Gastronomy Restaurant
    129 Dumpling Restaurant
    130 Paella Restaurant
    137 Caribbean Restaurant
    137 Swiss Restaurant
    150 Moroccan Restaurant
    179 Afghan Restaurant
    181 Arepa Restaurant
    185 Brazilian Restaurant
    207 Korean Restaurant
    313 Argentinian Restaurant
    322 African Restaurant
    326 Scandinavian Restaurant
    338 Vietnamese Restaurant
    422 Portuguese Restaurant
    522 Vegetarian / Vegan Restaurant
    580 Falafel Restaurant
    753 Thai Restaurant
    756 Mexican Restaurant
   1096 German Restaurant
   1288 American Restaurant
   1371 Indian Restaurant
   1456 Tapas Restaurant
   1497 Greek Restaurant
   1689 Eastern European Restaurant
   1749 Japanese Restaurant
   1911 Spanish Restaurant
   2072 Mediterranean Restaurant
   2122 Sushi Restaurant
   2219 Chinese Restaurant
   2411 Asian Restaurant
   2529 Seafood Restaurant
   2713 Middle Eastern Restaurant
   2861 French Restaurant
   7659 Italian Restaurant
   8632 Fast Food Restaurant
  10093 Turkish Restaurant
  15196 Restaurant
```

D.
grep -w 'Restaurant' POIeu.txt > eu_restaurant.txt # grep -w search for ‘Restaurant’. Then > put it to eu_restaurant.txt
awk -F '\t' '{print $4}' eu_restaurant.txt | sort | uniq -c | sort
#Use awk, file is delimited by tab. Print column 4. Then pipe to sort where it sort the values. Pipe to uniq -c counts the frequency of each unique values. Pipe to sort to sort the values.
Restaurant appears the most. But ‘Restaurant’ might mean unclassified restaurant. So maybe the correct answer is ‘Turkish Restaurant’.

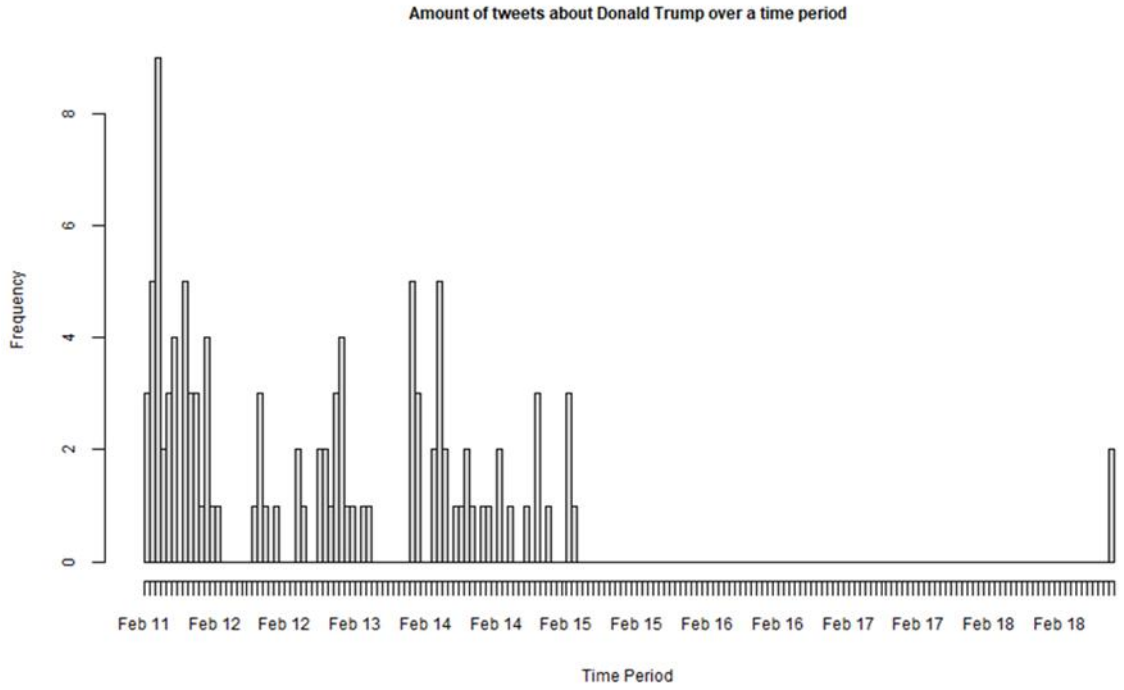
Task B

1.
```
Edward Priyatna@LAPTOP-GG6FAS2B ~/HW3_2
$ grep -o 'Donald Trump' Donald_Trump.txt | wc -l
114
```
grep -w 'Donald Trump' Twitter_Data_1 > Donald_Trump.txt
#grep searches for ‘Donald Trump’ w means only when the whole word matches. > to pass it to new txt Donald_Trump.txt
grep -o 'Donald Trump' Donald_Trump.txt | wc -l

2. taf = read.table("timestamps.txt",header=FALSE,sep=",")
Read.table will read the timestamp.csv file into a dataframe and store it in a variable (here is taf), header=FALSE is to indicate that there is no header, sep=’,’ to represent the delimiter of the csv file

taf[['V1']] <- strptime(taf[['V1']],format="%a %b %d %T %z %Y")

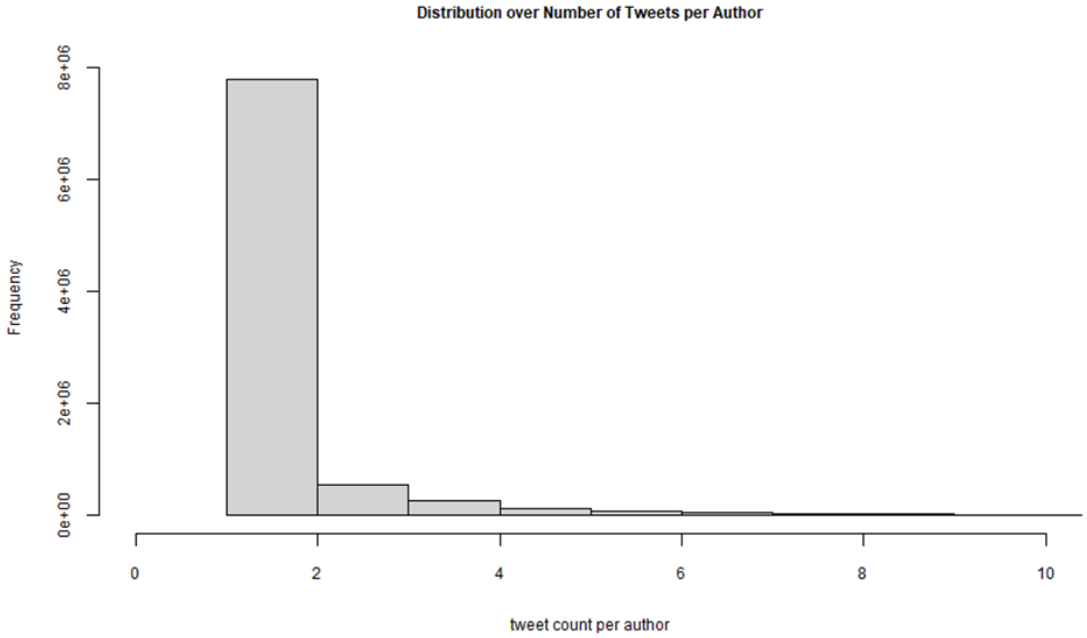
strptime to convert the timestamps into datetime format. The format is telling the program that the column is following certain format. After converting it, save it back to the original column.



3. par(mar=c(5, 4, 2, 1)) #setting margin so the graph is nice

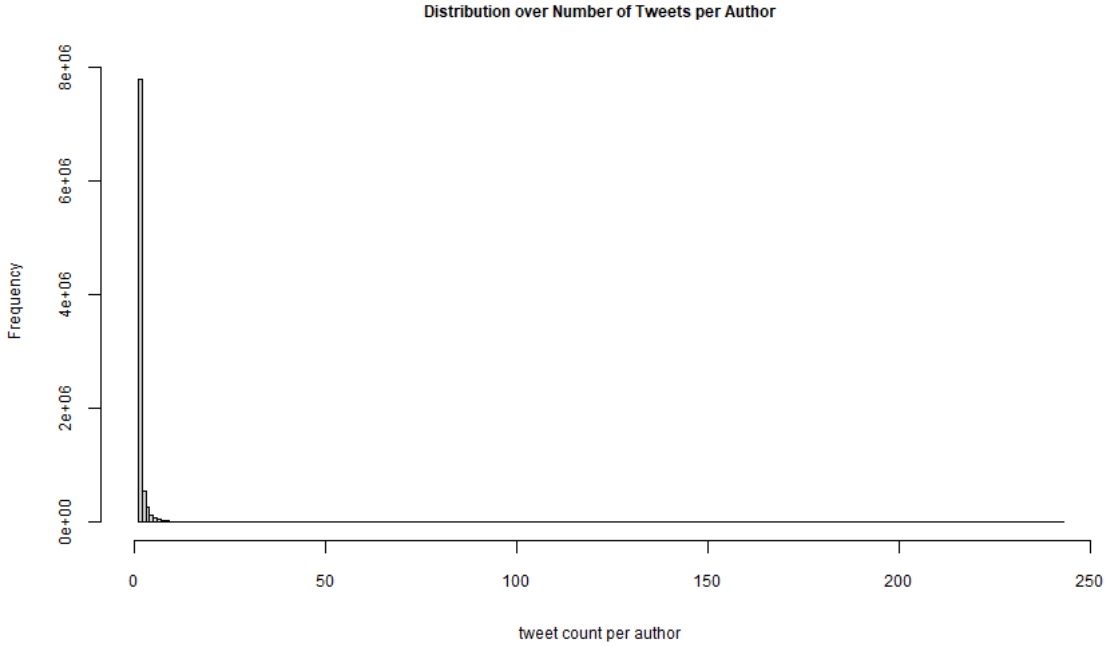
hist(taf[["V1"]],breaks='hours',main='Amount of tweets about Donald Trump over a time period',freq=TRUE,xlab='Time Period',cex.main=0.75, cex.lab=0.75, cex.axis=0.75)
# hist is to create a histogram by taking a column (here is taf[["V1"]]), breaks is to set the number of bins (day means to break by the date of the dataset), main is the title of the histogram. freq is to set the number of frequency to be a integer, xlab is to set the name of x-axis, the last three code is to adjust the size of title, the name of x-axis, and axes.

4. cut -f 3,4 Donald_Trump.txt | less #pipe the file to less to see the file
then type /wind farm
From the image we can see a lot of wind farms before February 15. I searched the internet and found this article: https://www.theguardian.com/world/2014/feb/11/donald-trump-loses-windfarm-scottish-golf-resort. It seems that he opposes the building of a wind farm near his golf course. However, he lost the legal battle and a wind farm was build near his golf course.



5.

hist(tweet_count[['V1']],breaks=260,main='Distribution over Number of Tweets per Author',xlab='tweet count per author',freq=TRUE,cex.main=0.75, cex.lab=0.75, cex.axis=0.75, xlim=c(0,10))
# hist is to create a histogram by taking a column (here is df2[["V1"]]), breaks is to set the number of bins, main is the title of the histogram. freq = TRUE is to set the number of frequency to be a integer, xlab is to set the name of x-axis, the last three code is to adjust the size of title, the name of x-axis and axes, and xlim limit the output of x-axis.



hist(tweet_count[['V1']],breaks=260,main='Distribution over Number of Tweets per Author',xlab='tweet count per author',freq=TRUE,cex.main=0.75, cex.lab=0.75, cex.axis=0.75)
# hist is to create a histogram by taking a column (here is tweet_count[['V1']]), breaks is to set the number of bins, main is the title of the histogram. freq = TRUE is to set the number of frequency to be a integer, xlab is to set the name of x-axis, the last three code is to adjust the size of title, the name of x-axis and axes.
This is the full graph. It is hard to see.
It is very hard to find the correct x and y limit of the graph because the graph drops drastically after x=1. Most twitter accounts only sends one tweet and never send another.