STAT243 Problem Set 1

SID:26955248

September 8, 2015

Problem1

a

First we need to download the zip file to local directory using wget, and name it to be data.zip. Next, uncompress the file and name the dataset to be data.csv.

Code:

```
wget --output-document data.zip "http://data.un.org/Handlers/DownloadHan\
dler.ashx?DataFilter=itemC\
ode:526&DataMartId=FAO&Format=csv&c=2,3,4,5,6,7&s=co\
untryName:asc,elementCode:asc,year:desc"

unzip -c data.zip > data.csv
```

```
## --2015-09-08 14:13:29--  http://data.un.org/Handlers/DownloadHandler.ashx?DataFilter=itemCode:526
## Resolving data.un.org... 85.159.207.229
## Connecting to data.un.org|85.159.207.229|:80... connected.
## HTTP request sent, awaiting response... 200 OK
## Length: 68264 (67K) [application/zip]
## Saving to: 'data.zip'
##
##      0K ......... ......... ......... ......... ......... 75%  266K 0s
##     50K ......... ......                                  100% 94.0K=0.4s
##
## 2015-09-08 14:13:32 (183 KB/s) - 'data.zip' saved [68264/68264]
```

Now using the command grep to split the data of regions and countries to two files, and also fix the comma problem in countries. I subsititute comma space to -. E.g. China, Mainland becomes China-Mainland, and this step helps me in future steps while making comma to be the delimiter.

```
grep "+" data.csv > regions.csv
grep -v "+" data.csv > countries.csv
sed "s/, /-/g" countries.csv > countries_aprictos.csv
```

Now we are determining the five most land used countries in 2005, first use "grep" to find only the rows for Area Harvested in 2005. We cannot directly grep 2005 because 2005 could be the area used quantity of some countries, so I use "\"2005"\". Then we use sed to replace " " to nothing, which makes the numerical values evaluable in sort. Finally we sort the dataset and find the top 5 countires by the head command, and cut out the first and sixth columns so that it only displays the country name and the land use data.

The top 5 countries are Turkey, Iran, Pakistan, Uzbekistan, and Algeria.

Code:

```
grep -i "Area" countries_aprictos.csv | grep "\"2005\"" |\
sed 's/"//g' | sort -t',' -k6 -n -r| head -5 | cut -d"," -f1,6

## Turkey,60000.00000
## Iran-Islamic Republic of,49388.00000
## Pakistan,28884.00000
## Uzbekistan,28000.00000
## Algeria,22888.00000
```

Now we automate the analysis by writing a function called automate and set up a for loop, the i represents the year index. I also print the message to make the result looks nicer. The default interval of year change is 10 years in this function.

```
function automate(){
for ((i=1965;i<=2005;i=i+10))
do
printf "This is the rank for year $i \n"
grep -i "Area" countries_aprictos.csv | grep "\"$i\"" |\
sed 's/"//g' | sort -t',' -k6 -n -r | head -5 | cut -d"," -f1,6
printf "\n"
done
}
automate

## This is the rank for year 1965
## USSR,60000.00000
## Turkey,46500.00000
## United States of America,15460.00000
## Spain,15100.00000
## Tunisia,15000.00000
##
## This is the rank for year 1975
## USSR,71000.00000
## Turkey,41500.00000
## Spain,23300.00000
## Tunisia,18981.00000
## Italy,14000.00000
##
## This is the rank for year 1985
## Turkey,47250.00000
## USSR,45000.00000
## Spain,20000.00000
## Iran-Islamic Republic of,16504.00000
## Tunisia,15000.00000
##
## This is the rank for year 1995
## Turkey,57355.00000
## Iran-Islamic Republic of,28000.00000
## Spain,22500.00000
## Ukraine,18600.00000
## Tunisia,17000.00000
##
```

```
## This is the rank for year 2005
## Turkey,60000.00000
## Iran-Islamic Republic of,49388.00000
## Pakistan,28884.00000
## Uzbekistan,28000.00000
## Algeria,22888.00000
```

b

I write a function that takes numerical value of the code as an input, and read user's input to be the variable x by the "read" command, then I subsititute the 526 to $x in the http URL. Finally unzip the downloaded file and use the less statement to preview the csv in the compressed file. The output has a preview of CSV file, while requiring a user input, which is relatively long, so I disable the output for this part.

Code:

```
function httpcode(){
echo -n "what is your code for the Agriculture Product?"
read x
wget --output-document data$x.zip "http://data.un.org/Handlers/Downloa\
dHandler.ashx?DataFilter=itemCode:$x&Dat\
aMartId=FAO&Format=csv&c=2,3,4,5,6,7&s=cou\
ntryName:asc,elementCode:asc,year:desc"
unzip -c data$x.zip > data$x.csv
head -10 data$x.csv
}
```

c

I opened the item code page "http://faostat.fao.org/site/384/default.aspx", and oberseved that the pattern <\td><td> only exits in the table rows. First I downloaded the the html file, and greped the lines with <\td><td>, then I used the sed command to replace those <.>s to be "-" to be my future delimiters. Then I cut out the second and the fourth column respect to code and product names. Then I use sort -u to avoid repeated names, and finally I named this output to be codename.csv. For these steps, I just extracted a table from an html file.

```
wget --output-document codename.html "http://faostat.fao.org/site/384/default.aspx"
grep "</td><td>" codename.html | sed 's/td//g' |\
sed -e 's/<.><>/-/g' | cut -d'-' -f2,4 | sort -u -t'-' > codename.csv

## --2015-09-08 14:13:32--  http://faostat.fao.org/site/384/default.aspx
## Resolving faostat.fao.org... 193.43.36.221
## Connecting to faostat.fao.org|193.43.36.221|:80... connected.
## HTTP request sent, awaiting response... 200 OK
## Length: 166670 (163K) [text/html]
## Saving to: 'codename.html'
##
##      0K .......... .......... .......... .......... .......... 30% 56.0K 2s
##     50K .......... .......... .......... .......... .......... 61%  275K 1s
##    100K .......... .......... .......... .......... .......... 92%  271K 0s
```

```
##    150K .......... ..                                       100% 80.9M=1.3s
##
## 2015-09-08 14:13:34 (129 KB/s) - 'codename.html' saved [166670/166670]
```

Then I worte a function called name to code, which refined the function I wrote in part B. First define a variable x to take the input of product names, and referring to the first column-which was the code of the product, and I need to have a dollar sign after $1 so that it will strictly find the name. For instance, when we search "Cherries", it will return "Cherries" instead of "Cherries, Sour". Then the rest of the function was just the same as I did in part b. I previewed the first ten lines of the product "wheat", which corresponed to the code "15".

Code:

```
function nametocode(){
x=$(grep $1$ codename.csv | cut -d'-' -f1)
wget --output-document data$x.zip "http://data.un.org/Handlers/DownloadHan\
dler.ashx?DataFilter=itemCode:$x&Data\
MartId=FAO&Format=csv&c=2,3,4,5,6,7&s=countr\
yName:asc,elementCode:asc,year:desc"
unzip -c data$x.zip > data$x.csv
head -10 data$x.csv
}
nametocode Wheat

## --2015-09-08 14:13:34--  http://data.un.org/Handlers/DownloadHandler.ashx?DataFilter=itemCode:15&
## Resolving data.un.org... 85.159.207.229
## Connecting to data.un.org|85.159.207.229|:80... connected.
## HTTP request sent, awaiting response... 200 OK
## Length: 169459 (165K) [application/zip]
## Saving to: 'data15.zip'
##
##      0K .......... .......... .......... .......... .......... 30% 41.5K 3s
##     50K .......... .......... .......... .......... .......... 60% 92.9K 1s
##    100K .......... .......... .......... .......... .......... 90% 94.6K 0s
##    150K .......... .....                                      100%  156K=2.4s
##
## 2015-09-08 14:13:41 (69.8 KB/s) - 'data15.zip' saved [169459/169459]
##
## Archive:  data15.zip
##   inflating: UNdata_Export_20150908_231333625.csv
## "Country or Area","Element Code","Element","Year","Unit","Value","Value Footnotes"
## "Afghanistan","111","Seed","2007","tonnes","181815.00000","Fc"
## "Afghanistan","111","Seed","2006","tonnes","209610.00000","Fc"
## "Afghanistan","111","Seed","2005","tonnes","207740.00000","Fc"
## "Afghanistan","111","Seed","2004","tonnes","199070.00000","Fc"
## "Afghanistan","111","Seed","2003","tonnes","160480.00000","Fc"
## "Afghanistan","111","Seed","2002","tonnes","197200.00000","Fc"
## "Afghanistan","111","Seed","2001","tonnes","148070.00000","Fc"
```

Problem 2

First I download the html file and name it to be climate.html, then we extract the txt files names to a variable called txtnames. First we grep the lines containing .txt, then I eliminate all expressions before href=" by sed command. Finally I replace all codes after txt'> to be txt, the I got a list of txt file names. Then I use the command printf to show the status message indicating which file is being downloaded. Again the output contains the downloading status, so I added -nv to disable system downloading status.

Code

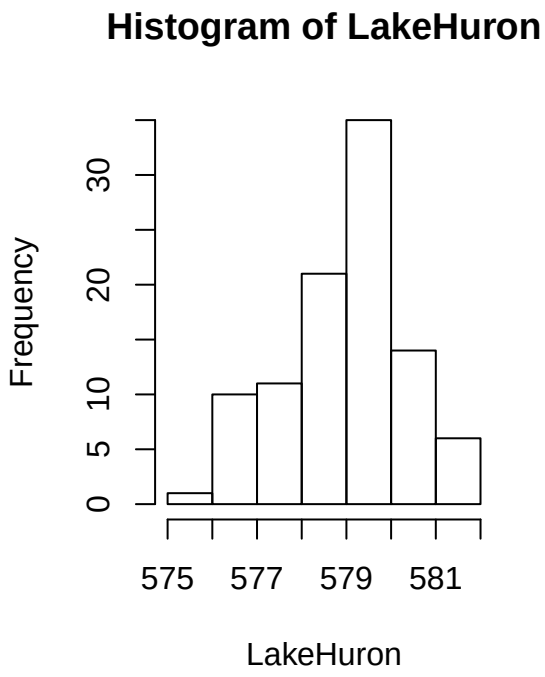
```
wget --output-document climate.html "http://www1.ncdc.noaa.gov/pub/data/ghcn/daily/"
txtnames=$(grep .txt climate.html | sed 's/.*href="//g' | sed 's/txt">.*/txt/g')
for i in $txtnames;
do
printf "\n You are downloading the text file $i \n"
wget -nv "http://www1.ncdc.noaa.gov/pub/data/ghcn/daily/$i"
done
```

```
## --2015-09-08 14:13:41--  http://www1.ncdc.noaa.gov/pub/data/ghcn/daily/
## Resolving www1.ncdc.noaa.gov... 205.167.25.102, 2610:20:8040:2::102
## Connecting to www1.ncdc.noaa.gov|205.167.25.102|:80... connected.
## HTTP request sent, awaiting response... 200 OK
## Length: 4888 (4.8K) [text/html]
## Saving to: 'climate.html'
##
##      0K ....                                                  100% 7.64M=0.001s
##
## 2015-09-08 14:13:41 (7.64 MB/s) - 'climate.html' saved [4888/4888]
##
##
##  You are downloading the text file ghcnd-countries.txt
## 2015-09-08 14:13:41 URL:http://www1.ncdc.noaa.gov/pub/data/ghcn/daily/ghcnd-countries.txt [3670/3
##
##  You are downloading the text file ghcnd-inventory.txt
## 2015-09-08 14:13:54 URL:http://www1.ncdc.noaa.gov/pub/data/ghcn/daily/ghcnd-inventory.txt [262899
##
##  You are downloading the text file ghcnd-states.txt
## 2015-09-08 14:13:54 URL:http://www1.ncdc.noaa.gov/pub/data/ghcn/daily/ghcnd-states.txt [1086/1086
##
##  You are downloading the text file ghcnd-stations.txt
## 2015-09-08 14:13:56 URL:http://www1.ncdc.noaa.gov/pub/data/ghcn/daily/ghcnd-stations.txt [8431010
##
##  You are downloading the text file ghcnd-version.txt
## 2015-09-08 14:13:57 URL:http://www1.ncdc.noaa.gov/pub/data/ghcn/daily/ghcnd-version.txt [270/270]
##
##  You are downloading the text file readme.txt
## 2015-09-08 14:13:57 URL:http://www1.ncdc.noaa.gov/pub/data/ghcn/daily/readme.txt [24088/24088] ->
##
##  You are downloading the text file status.txt
## 2015-09-08 14:13:57 URL:http://www1.ncdc.noaa.gov/pub/data/ghcn/daily/status.txt [29430/29430] ->
```

Problem3

The height of the water level in Lake Huron flucuates over time. Here I analyze the variation using R. I show a histogram of the lake levels for the lake levels for the period 1875 and 1972.

```
hist(LakeHuron)
```




```
lowHi<-c(which.min(LakeHuron),which.max(LakeHuron))
yearExtrema <-attributes(LakeHuron)$tsp[1]-1 + lowHi
```