# STAT 243 - PS 1

*Cam Adams*

*9/8/2017*

1. This question is the class survey that you already filled out.
2. This problem provides practice in downloading and manipulating data using shell scripting

a. Download the data for apricotes

```
cd /Users/CamAdams/repos/STAT243/ps1/ #set workin
wget -O tmp.zip -nv "http://data.un.org/Handlers/DownloadHandler.ashx?DataFilter=itemCode:526&DataMartI
unzip tmp.zip #unzip data
sed "s/, /_/g"  UNdata_Export*.csv | tr -d '"' > tmp1.csv #replace , with _ inside fields
mv tmp1.csv apricots.csv #rename
rm UNdata_Export*.csv tmp.zip #remove unused files
```

```
## 2017-09-08 08:24:47 URL:http://data.un.org/Handlers/DownloadHandler.ashx?DataFilter=itemCode:526&Data
## Archive:  tmp.zip
##   inflating: UNdata_Export_20170908_172445169.csv
```

Extract the data for individual countries into one file and for regions of the world into another file.

```
grep -v "+" apricots.csv > apricots_Country.csv #create country only data
grep "+" apricots.csv > apricots_World.csv #create region data
```

Then subset the country-level data to the year 2005. Based on the "area harvested" determine the five countries using the most land to produce apricots. You'll need to look carefully at arguments to the sort utility.

```
#grep -w 2005 apricots_Country.csv | head -5 #preview 2005 country subset
grep -w 2005 apricots_Country.csv > apricots_Country_2005.csv #write 2005 country subset
grep "Area Harvested" apricots_Country_2005.csv | sort -t, -n -r -k6 | cut -d, -f1,3,4,6 | head -5 #fiv
```

```
## Turkey,Area Harvested,2005,60000.00000
## Iran_Islamic Republic of,Area Harvested,2005,49388.00000
## Pakistan,Area Harvested,2005,28884.00000
## Uzbekistan,Area Harvested,2005,28000.00000
## Algeria,Area Harvested,2005,22888.00000
```

Now automate your analysis and examine the top five countries for 1965, 1975, 1985, 1995, and 2005. Have the rankings changed?

```
#!/bin/bash
#prob2a.sh
#for loop to automate subseting top five countries with land available for harvest in years 1965-2005 i

for YEAR in 1965 1975 1985 1995 2005
  do
    echo "==== $YEAR ===="
    grep "Area Harvested" apricots_Country.csv | grep -w $YEAR | sort -t, -n -r -k6 | cut -d, -f1,3,4,6
done
```

```
## ==== 1965 ====
## USSR,Area Harvested,1965,60000.00000
## Turkey,Area Harvested,1965,46500.00000
## United States of America,Area Harvested,1965,15460.00000
```

```
## Spain,Area Harvested,1965,15100.00000
## Tunisia,Area Harvested,1965,15000.00000
## ==== 1975 ====
## USSR,Area Harvested,1975,71000.00000
## Turkey,Area Harvested,1975,41500.00000
## Spain,Area Harvested,1975,23300.00000
## Tunisia,Area Harvested,1975,18981.00000
## Italy,Area Harvested,1975,14000.00000
## ==== 1985 ====
## Turkey,Area Harvested,1985,47250.00000
## USSR,Area Harvested,1985,45000.00000
## Spain,Area Harvested,1985,20000.00000
## Iran_Islamic Republic of,Area Harvested,1985,16504.00000
## Tunisia,Area Harvested,1985,15000.00000
## ==== 1995 ====
## Turkey,Area Harvested,1995,57355.00000
## Iran_Islamic Republic of,Area Harvested,1995,28000.00000
## Spain,Area Harvested,1995,22500.00000
## Ukraine,Area Harvested,1995,18600.00000
## Tunisia,Area Harvested,1995,17000.00000
## ==== 2005 ====
## Turkey,Area Harvested,2005,60000.00000
## Iran_Islamic Republic of,Area Harvested,2005,49388.00000
## Pakistan,Area Harvested,2005,28884.00000
## Uzbekistan,Area Harvested,2005,28000.00000
## Algeria,Area Harvested,2005,22888.00000
```

The ranking have changed some over time. The USSR was at the top in 1965 and 1975, but fell off the list, and teh US was in the top 5 only in 1965. European countries (Italy, Spain, Ukraine) were in the top five until 2005. Turkey has always been near or at the top of the rankings. More non-European countires have been ranked in the top 5 after 1985, and include Iran, Tunisia, Algeria, and Uzbekistan.

b. Write a bash function that takes as input a single item code (e.g., 526 for apricots, 572 for avocados) and prints out to the screen the data stored in the CSV file, such that the information could be piped on to UNIX another operation. Your function should detect if the user provides the wrong number of arguments and return a useful error message. It should also give useful help information if the user invokes the function as: "myfun -h".

```
#!/bin/bash
#function that will
  #1. takes as input a single item code (e.g., 526 for apricots, 572 for avocados)
  #2. prints out to the screen the data stored in the CSV file, such that the information could be pipe
  #3. it will detect if the user provides the wrong number of arguments and return a useful error messa
  #4. help info is avaiable with "-h"

func() {

    #inputs
    VAR=$1
    VAR_LENGTH=${#VAR}

    echo "================="
    echo "Input: $VAR"

    #help info
```

```
    if  [[ "$1" == "-h" || "$1" == "--help" ]]    #help file
    then
        printf "This function downloads historical agricultural data from the UN website. \n"
        printf "It will only accept a three digit integer: func 576 \n"
        printf "Any other entry will return an error. \n"
        return
    fi

    #check input formatting
    if ! [[ VAR_LENGTH -eq 3 ]]                   #input==3 char
    then
        echo "Sorry, 3 digit integers only"
        return
    fi

    if ! [[ $VAR =~ ^[0-9]+$ ]]                   #numeric digits only
    then
        echo "Sorry, 3 digit integers only"
        return
    fi

    #retrieve data from UN website using input $VAR
    wget -O tmp.zip -nv "http://data.un.org/Handlers/DownloadHandler.ashx?DataFilter=itemCode:$VAR&Datal
    #unzip and remove .zip file
    unzip tmp.zip
    rm tmp.zip
    #rename file using user input
    mv UN*.csv $VAR.csv
    #print resulting .csv to screen
    cat $VAR.csv | head -5

    return
}
func
func -h
func --help
func 0
func 999999
func abc
func a
func 526
func 572
```

```
## ================
## Input:
## Sorry, 3 digit integers only
## ================
## Input: -h
## This function downloads historical agricultural data from the UN website.
## It will only accept a three digit integer: func 576
## Any other entry will return an error.
## ================
## Input: --help
## This function downloads historical agricultural data from the UN website.
```

```
## It will only accept a three digit integer: func 576
## Any other entry will return an error.
## ================
## Input: 0
## Sorry, 3 digit integers only
## ================
## Input: 999999
## Sorry, 3 digit integers only
## ================
## Input: abc
## Sorry, 3 digit integers only
## ================
## Input: a
## Sorry, 3 digit integers only
## ================
## Input: 526
## 2017-09-08 08:24:49 URL:http://data.un.org/Handlers/DownloadHandler.ashx?DataFilter=itemCode:526&Data
## Archive:  tmp.zip
##   inflating: UNdata_Export_20170908_172447982.csv
## "Country or Area","Element Code","Element","Year","Unit","Value","Value Footnotes"
## "Afghanistan","31","Area Harvested","2007","Ha","3400.00000","F "
## "Afghanistan","31","Area Harvested","2006","Ha","8030.00000",""
## "Afghanistan","31","Area Harvested","2005","Ha","5200.00000","F "
## "Afghanistan","31","Area Harvested","2004","Ha","5200.00000","F "
## ================
## Input: 572
## 2017-09-08 08:24:51 URL:http://data.un.org/Handlers/DownloadHandler.ashx?DataFilter=itemCode:572&Data
## Archive:  tmp.zip
##   inflating: UNdata_Export_20170908_172449659.csv
## "Country or Area","Element Code","Element","Year","Unit","Value","Value Footnotes"
## "Africa +","31","Area Harvested","2007","Ha","62426.00000","A "
## "Africa +","31","Area Harvested","2006","Ha","59938.00000","A "
## "Africa +","31","Area Harvested","2005","Ha","60691.00000","A "
## "Africa +","31","Area Harvested","2004","Ha","60225.00000","A "
```

3. Your task here is to automatically download all the files ending in .txt from this National Climate Data Center website: http://www1.ncdc.noaa.gov/pub/data/ghcn/daily/...

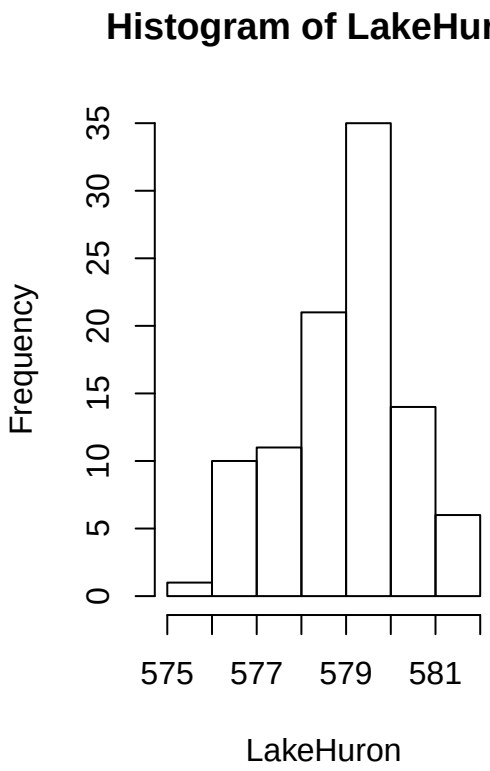
```
echo "download html"
wget -nv https://www1.ncdc.noaa.gov/pub/data/ghcn/daily/ #download html of the data
grep .txt index.html | cut -d'"' -f8 > index.html.txtFileNames #get .txt file names from index.html

#while loop to extract each text file from website
while read FILE
  do
    echo "================="
    echo "Dowloading: $FILE"
    wget -nv https://www1.ncdc.noaa.gov/pub/data/ghcn/daily/$FILE #download text file
done < index.html.txtFileNames
```

```
## download html
## 2017-09-08 08:24:51 URL:https://www1.ncdc.noaa.gov/pub/data/ghcn/daily/ [5670/5670] -> "index.html"
## ================
## Dowloading: ghcnd-countries.txt
## 2017-09-08 08:24:52 URL:https://www1.ncdc.noaa.gov/pub/data/ghcn/daily/ghcnd-countries.txt [3670/367
```

```
## ================
## Dowloading: ghcnd-inventory.txt
## 2017-09-08 08:25:01 URL:https://www1.ncdc.noaa.gov/pub/data/ghcn/daily/ghcnd-inventory.txt [27401648/
## ================
## Dowloading: ghcnd-states.txt
## 2017-09-08 08:25:01 URL:https://www1.ncdc.noaa.gov/pub/data/ghcn/daily/ghcnd-states.txt [1086/1086] -
## ================
## Dowloading: ghcnd-stations.txt
## 2017-09-08 08:25:07 URL:https://www1.ncdc.noaa.gov/pub/data/ghcn/daily/ghcnd-stations.txt [8914416/89
## ================
## Dowloading: ghcnd-version.txt
## 2017-09-08 08:25:08 URL:https://www1.ncdc.noaa.gov/pub/data/ghcn/daily/ghcnd-version.txt [270/270] -
## ================
## Dowloading: readme.txt
## 2017-09-08 08:25:08 URL:https://www1.ncdc.noaa.gov/pub/data/ghcn/daily/readme.txt [24179/24179] -> "
## ================
## Dowloading: status.txt
## 2017-09-08 08:25:09 URL:https://www1.ncdc.noaa.gov/pub/data/ghcn/daily/status.txt [30827/30827] -> "
```

4. Your task in this problem is to produce a single-page of PDF that looks like the last page of this assignment, using either the knitr package in R with LATEX or R Markdown. If you are a Statistics student you should use LATEX.

```
hist(LakeHuron)
```




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

The height of the water level in Lake Huron fluctuates over time. Here I 'analyze' the variation using R. I show a histogram of the lake levels for the period 1876 to 1964.