Problem Set 1: STAT243

Cristobal Pais - cpaismz@berkeley.edu

September 8, 2017

Appendix

In this section we execute the codes used in this report, changing the sintax for matching the one implemented in *knitr*. Thus, $ is replaced by \$ and \ is replaced by “\”. In addition, extra *echo* commands are added just for visual purposes.

Problem 2

a) Download, extract and analyze

```
# Working directory
cd ~/latex

# Download the file in .zip format
wget -q -O apricots.zip "http://data.un.org/Handlers/DownloadHandler.ashx?"\
"DataFilter=itemCode:526&DataMartId=FAO&Format=csv&c=2,3,4,5,6,7&s=countryName:"\
"asc,elementCode:asc,year:desc"

# Unzip and delete the downloaded file
unzip -q apricots.zip
rm apricots.zip

# Change original name based on last file created
mv \$(ls -rt | tail -n 1) apricots.csv

# Generate regions file and keep the headers
head -n 1 apricots.csv > apricots_regions.csv

# Add regions file content
grep "+" apricots.csv >> apricots_regions.csv

# Adding footnotes
tail -n 7 apricots.csv >> apricots_regions.csv

# Generating countries file, footnotes included
grep -v "+" apricots.csv > apricots_countries.csv
```

```
# Filter by year and Area Harvested, then sort and show the top 5 (only names)
echo "Year 2005 Area Harvested ranking"
grep ""\""2005"\""" apricots_countries.csv | grep "Area Harvested" | \
sort --field-separator='"' --key=12 -nr | head -n 5 | awk -F "," '{print \$1}'
echo ""

# Individual codes: for each year
echo "Year 1965 Area Harvested ranking"
grep ""\""1965"\""" apricots_countries.csv | grep "Area Harvested" | \
sort --field-separator='"' --key=12 -nr | head -n 5 | awk -F "," '{print \$1}'
echo ""
echo "Year 1975 Area Harvested ranking"
grep ""\""1975"\""" apricots_countries.csv | grep "Area Harvested" | \
sort --field-separator='"' --key=12 -nr | head -n 5 | awk -F "," '{print \$1}'
echo ""
echo "Year 1985 Area Harvested ranking"
grep ""\""1985"\""" apricots_countries.csv | grep "Area Harvested" | \
sort --field-separator='"' --key=12 -nr | head -n 5 | awk -F "," '{print \$1}'
echo ""
echo "Year 1995 Area Harvested ranking"
grep ""\""1995"\""" apricots_countries.csv | grep "Area Harvested" | \
sort --field-separator='"' --key=12 -nr | head -n 5 | awk -F "," '{print \$1}'
echo ""
echo "Year 2005 Area Harvested ranking"
grep ""\""2005"\""" apricots_countries.csv | grep "Area Harvested" | \
sort --field-separator='"' --key=12 -nr | head -n 5 | awk -F "," '{print \$1}'
```

```
## Year 2005 Area Harvested ranking
## "Turkey"
## "Iran
## "Pakistan"
## "Uzbekistan"
## "Algeria"
##
## Year 1965 Area Harvested ranking
## "USSR"
## "Turkey"
## "United States of America"
## "Spain"
## "Tunisia"
##
## Year 1975 Area Harvested ranking
## "USSR"
## "Turkey"
## "Spain"
## "Tunisia"
## "Italy"
```

```
##
## Year 1985 Area Harvested ranking
## "Turkey"
## "USSR"
## "Spain"
## "Iran
## "Tunisia"
##
## Year 1995 Area Harvested ranking
## "Turkey"
## "Iran
## "Spain"
## "Ukraine"
## "Tunisia"
##
## Year 2005 Area Harvested ranking
## "Turkey"
## "Iran
## "Pakistan"
## "Uzbekistan"
## "Algeria"
```

```
# Working directory
cd ∼/latex

# Automated code for years 1965,...,2005 by 10
# For inside the specific set
for n in {1965..2005..10}
do
  # Print the year under study
  echo "Year "\${n}" Area Harvested ranking"

  # Print top five Area harvested ranking (structure of previous code)
  grep ""\${n}"" apricots_countries.csv | grep "Area Harvested" | \
  sort --field-separator=’"’ --key=12 -nr | head -n 5 | awk -F "," ’{print \$1}’
  echo ""
done


## Year 1965 Area Harvested ranking
## "USSR"
## "Turkey"
## "United States of America"
## "Spain"
## "Tunisia"
##
## Year 1975 Area Harvested ranking
## "USSR"
## "Turkey"
## "Spain"
## "Tunisia"
## "Italy"
##
## Year 1985 Area Harvested ranking
## "Turkey"
## "USSR"
## "Spain"
## "Spain"
## "Iran
##
## Year 1995 Area Harvested ranking
## "Turkey"
## "Iran
## "Spain"
## "Ukraine"
## "Tunisia"
##
## Year 2005 Area Harvested ranking
## "Turkey"
## "Iran
## "Pakistan"
## "Uzbekistan"
## "Algeria"
```

b) Bash function

```
#!/bin/sh
# Working directory
cd ~/latex

# Function definition
print_item_data () {
# A regular expression is defined for checking input
re='^[0-9]+$'

# If flag --help or -h is given, output help msg
if [ "\$1" == "--help" ] || [ "\$1" == "-h" ]; then
    echo "Usage: print_item_data [INTEGER]"
    echo "Downloads and prints the data on agricultural production for item"
    echo "with ID INTEGER"
    echo "Information Provided by United Nations Food and Agriculture"
    echo "Organization (FAO)"
# If the input is not a number, output error msg 1
elif ! [[ \$1 =~ \$re ]]; then
    # Print error msg 1
    echo "error: Input should be only an integer value"
    echo "use print_item_data -h or --help for more information"

# If we have more than 1 input, output error msg 2
elif [ \$# != "1" ]; then
    # Print error msg 2
    echo "error: Function needs only one integer as input"
    echo "use print_item_data -h or --help for more information"

else
    # Download the file in .zip format
    wget -q -O FAO_\${1}_data.zip "http://data.un.org/Handlers/DownloadHandler."\
"ashx?DataFilter=itemCode:"\${1}"&DataMartId=FAO&Format=csv&c=2,3,4,5,6,7&s="\
"countryName:asc,elementCode:asc,year:desc"

 # Unzip the downloaded file
    unzip -q FAO_\${1}_data.zip

    # Change original name based on last file created
    mv \$(ls -rt | tail -n 1) FAO_\${1}_data.csv

    # Remove the zip file (optional)
    rm FAO_\${1}_data.zip

    # Prints out to the screen
    cat FAO_\${1}_data.csv
fi
}

# If needed, use the help flag for information
print_item_data -h

echo ""

# Print out the apricots data (ID=526)
print_item_data 526 | head -n 5

## Usage: print_item_data [INTEGER]
## Downloads and prints the data on agricultural production for item
## with ID INTEGER
## Information Provided by United Nations Food and Agriculture
## Organization (FAO)
##
## "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 "
```

Problem 3

b) Download files by extension from url

```
#!/bin/sh
# Working directory
cd ∼/latex

# Get index.html file without output
wget -q "https://www1.ncdc.noaa.gov/pub/data/ghcn/daily/"

# Get the name of the .txt files and save them to a text file
egrep -oe ’>[ˆ<].*\.txt’ index.html | tr -d ">" > TextList.txt

# Output current status
echo "Download starts"

# Read txt with names line by line and save it to filename var
while IFS= read -r filename
do
    # Indicates the user the file being downloaded
    echo "File \${filename} is being downloaded"

    # Download the file
    wget -q "https://www1.ncdc.noaa.gov/pub/data/ghcn/daily/"\${filename}

    # Tells the user that it has been downloaded
    echo "File \${filename} has been downloaded"
    echo ""

# Declare the input file for reading on while loop
done < TextList.txt

# Remove the index.html and TextList files
rm index.html TextList.txt

## Download starts
## File ghcnd-countries.txt is being downloaded
## File ghcnd-countries.txt has been downloaded
##
## File ghcnd-inventory.txt is being downloaded
## File ghcnd-inventory.txt has been downloaded
##
## File ghcnd-states.txt is being downloaded
## File ghcnd-states.txt has been downloaded
##
## File ghcnd-stations.txt is being downloaded
## File ghcnd-stations.txt has been downloaded
```

```
##
## File ghcnd-version.txt is being downloaded
## File ghcnd-version.txt has been downloaded
##
## File readme.txt is being downloaded
## File readme.txt has been downloaded
##
## File status.txt is being downloaded
## File status.txt has been downloaded
```