Problem 1

(a)

To extract the data for Republican and Democratic candidates of Senate into two files, *wget* is used to download the summary file and piping with *grep* to file output gives me the result.

```
# Download .csv from FEC
wget -q -O "CandidateSummary.csv" "http://www.fec.gov/data/CandidateSummary.do?format=csv"
# Extract Senate, and then split into 'REP' and 'DEM'
grep "\"S\"" CandidateSummary.csv | grep "\"REP\"" > REP.csv
grep "\"S\"" CandidateSummary.csv | grep "\"DEM\"" > DEM.csv
```

With pre-processing using *sed* commands given, we can then *cut* the fields and *sort* according to total contributions by printing the top 5 with *head*.

```
# Preprocessing before spliting fields
sed 's/\([^",]\),/\1/g' REP.csv | sed 's/[$"]//g' > REPclean.csv
sed 's/\([^",]\),/\1/g' DEM.csv | sed 's/[$"]//g' > DEMclean.csv
# Sort on total contribution and display the richest five
cut -d ',' -f 2,3,4,5,7,8,20 REPclean.csv | sort -n -r -t ',' -k 7 | head -n 5
cut -d ',' -f 2,3,4,5,7,8,20 DEMclean.csv | sort -n -r -t ',' -k 7 | head -n 5
```

(b)

After downloading the files, we *unzip* to get the **.txt* files. With stored *PCID*, we can extract the number of contributions for a given candidate. This part of work is focused on the current candidates for presidential race year 2012.

```
# Download from FEC
wget -q "ftp://ftp.fec.gov/FEC/2012/cn12.zip"
wget -q "ftp://ftp.fec.gov/FEC/2012/indiv12.zip"
unzip -q cn12 # get cn.txt
unzip -q indiv12 # get itcont.txt
# Clear cn.txt down to current candidates for presidential race year 2012
cut cn.txt -d '|' -f 1-6,9-11 | grep "|2012|US|P|C|" > pres_cn.txt
# Grab PC ID with last name, return the first one if duplicated
NAME="OBAMA"
PCID=$(grep "|${NAME}," pres_cn.txt | cut -d '|' -f 8 | head -n 1)
# Individual contribution counts (only over 200 is recorded, ingore negative number counts, treat \
    as input error)
# National counts
grep "${PCID}" itcont.txt | cut -d '|' -f 15 | wc -l
# California counts
grep "${PCID}" itcont.txt | grep "CA" | cut -d '|' -f 15 | wc -l
```

(c)

We functionalize the shell commands in (b) to get results for multiple candidates. A file of candidates names is generated for testing the functions.

```
# Get the last name(s) of the 2012 presidential candidates in the REP and DEM
# These are candidates of interest for my discussion
grep "|DEM\|REP|" pres_cn.txt | cut -d '|' -f 2 | cut -d ',' -f 1 > name_cn.txt
# Clean for split fields to get total contributions of candidates
sed 's/\([^",]\),/\1/g' CandidateSummary.csv | sed 's/[$"]//g' > CanSum_clean.csv
```

Three functions are built for getting the total contributions (*getTotalCont*), number of contributions (over $200) nationwide (*getContNation*) and in California (*getContCA*).

```
# Functions
function getTotalCont () {
    ID=$(grep "|${1}," pres_cn.txt | cut -d '|' -f 1 | head -n 1)
    NM=$(grep "${ID}" pres_cn.txt | cut -d '|' -f 2)
    TOTAL=$(grep "${ID}" CanSum_clean.csv | cut -d ',' -f 20)
}
function getContNation () {
    ID=$(grep "|${1}," pres_cn.txt | cut -d '|' -f 8 | head -n 1)
    NM=$(grep "${ID}" pres_cn.txt | cut -d '|' -f 2)
    NCOUNT=$(grep "${ID}" itcont.txt | cut -d '|' -f 15 | wc -l)
}
function getContCA () {
    ID=$(grep "|${1}," pres_cn.txt | cut -d '|' -f 8 | head -n 1)
    NM=$(grep "${ID}" pres_cn.txt | cut -d '|' -f 2)
    CACOUNT=$(grep "${ID}" itcont.txt | grep "CA" | cut -d '|' -f 15 | wc -l)
}
```

The loop-to-loop calling of the functions for multiple predefined candidates is realized with *for...do...done*.

```
# Call the function
cnName=$(cat name_cn.txt)
for name in $cnName
do
    getTotalCont $name
    getContNation $name
    getContCA $name
    echo "${NM} :"
    echo "${TOTAL}|${NCOUNT}|${CACOUNT}"
done
```

Shell script running results

```
-----------------------------1.(a)-----------------------------
The five Senate candidates with the largest total contributions
---[ID,Name,Office,State,Party,Description,TotalContribution]---

in the Republican Party :
S0MA00109,BROWN SCOTT P,S,MA,REP,INCUMBENT,18188937.01
S6TN00216,CORKER ROBERT P JR,S,TN,REP,INCUMBENT,8724077.00
S2OH00170,MANDEL JOSH,S,OH,REP,CHALLENGER,8393673.00
S2TX00361,DEWHURST DAVID H,S,TX,REP,OPEN,8054403.00
S2TX00312,CRUZ RAFAEL EDWARD TED,S,TX,REP,OPEN,8016399.00

in the Democratic Party :
S2MA00170,WARREN ELIZABETH,S,MA,DEM,CHALLENGER,27955729.00
S0NY00410,GILLIBRAND KIRSTEN ELIZABETH,S,NY,DEM,INCUMBENT,13251208.00
S6OH00163,BROWN SHERROD,S,OH,DEM,INCUMBENT,10494500.78
S8FL00166,NELSON BILL,S,FL,DEM,CHALLENGER,10083866.48
S2VA00142,KAINE TIMOTHY MICHAEL,S,VA,DEM,OPEN,10005026.00

-------------------------------------------------------------

-----------------1.(b) with test case OBAMA----------------------
The number of contributions above 200 nationwide for OBAMA is :
```

```
188584

The number of contributions above 200 in California for OBAMA is :
61091

----------------------------------------------------------------------

----------------------1.(c) with REP,DEM|P----------------------
For candidates in presidential race of year 2012,
---[the total contributions|
number of contributions above 200 nationwide|
number of contributions above 200 in California (CA)]---

RICHARDSON, DARCY G :
2443.00|3|1
HERMAN, RAPHAEL :
251018.00|24|0
DAVIS, L JOHN JR :
13907.47|11|3
CISNEROS, CESAR :
|0|0
KARGER, FRED :
588139.57|1615|1582
BLANKENSHIP, JARED :
40666.17|50|10
SANTORUM, RICHARD J. :
22482279.25|16794|3700
DRUMMOND, KEITH :
625.00|1|0
LAWSON, EDGAR A :
17800.00|2|2
GINGRICH, NEWT :
23755104.35|19330|4359
PAUL, RON :
39827022.82|38907|10735
OBAMA, BARACK :
266173662.74|188584|61091
ROMNEY, MITT :
167495762.19|128688|31078

----------------------------------------------------------------------
```

Problem 2

(a)

The shell function *remoteRJobs* is realized with *ps* remotely after *ssh* to the specified machine. The jobs are sorted according to CPU usage percentage and will be displayed with the given number of lines. The default is to display all the jobs.

```
# Find R jobs and CPU usage on remote machines
function remoteRJobs () {
               if [ $# == "2" ]
               then
                  echo "The top ${2} %CPU usage of R jobs running on ${1}:"
                  echo "PID UID %CPU CMD"
                  ssh $1 ps -C R -o pid,user,%cpu,comm --sort=-%cpu | grep -v PID | head -n $2
                  echo ""
               elif [ $# == "1" ]
               then
                  echo "The %CPU usage of R jobs running on ${1}:"
                  echo "PID UID %CPU CMD"
                  ssh $1 ps -C R -o pid,user,%cpu,comm --sort=-%cpu | grep -v PID
                  echo ""
               else
                  echo "ERROR: **************************"
                  echo "remoteRJobs MACHINE [number]"
                  echo ""
               fi
}
```

(b)

For extra function on the *remoteRJobs*, the 2.0 version utilizes a *mysum* function for summing up the CPU and MEM usage. The calculation is done purely in shell without porting to R. Also the total CPU usage is averaged to the number of cores on the given remote machine, while the MEM usage is just simple sum up.

```
# Extend 2.(a) to add up the CPU and memory use of all R jobs
function mysum () {
               sum=0
               for num in $(cat $1)
               do
                   sum=$((sum+=num))
               done
}

function remoteRJobs2 () {
               if [ $# == "2" ]
               then
                  echo "The top ${2} %CPU usage of R jobs running on ${1}:"
                  echo "PID UID %CPU %MEM CMD"
                  ssh $1 ps -C R -o pid,user,pcpu,pmem,comm --sort=-pcpu | grep -v "%CPU" | head \
                      -n $2

                  cpunum=$(ssh $1 grep processor /proc/cpuinfo | wc -l)
                  # calculate cpu and memory usage
                  ssh $1 ps -C R -o pcpu --sort=-pcpu | grep -v "%CPU" | head -n $2 | sed 's/ //' \
                      | sed 's/\.//' | sed 's/^0//' > cpu.txt
                  ssh $1 ps -C R -o pmem --sort=-pmem | grep -v "%MEM" | head -n $2 | sed 's/ //' \
                      | sed 's/\.//' | sed 's/^0//' > mem.txt
```

```
    mysum cpu.txt;sum=$((sum/=10));sum=$((sum/=cpunum))
    echo "The top ${2} total %CPU used by R jobs on ${1} is ${sum}% for ${cpunum} \
        CPUs"
    mysum mem.txt;sum=$((sum/=10))
    echo "The top ${2} total %MEM used by R jobs on ${1} is ${sum}%"

    echo ""
  elif [ $# == "1" ]
  then
    echo "The %CPU usage of R jobs running on ${1}:"
    echo "PID UID %CPU %MEM CMD"
    ssh $1 ps -C R -o pid,user,pcpu,pmem,comm --sort=-pcpu | grep -v "%CPU"

    cpunum=$(ssh $1 grep processor /proc/cpuinfo | wc -l)
    # calculate cpu and memory usage
    ssh $1 ps -C R -o pcpu --sort=-pcpu | grep -v "%CPU" | sed 's/ //' | sed \
        's/\.//' | sed 's/^0//' > cpu.txt
    ssh $1 ps -C R -o pmem --sort=-pmem | grep -v "%MEM" | sed 's/ //' | sed \
        's/\.//' | sed 's/^0//' > mem.txt
    mysum cpu.txt;sum=$((sum/=10));sum=$((sum/=cpunum))
    echo "The total %CPU used by R jobs on ${1} is ${sum}% for ${cpunum} CPUs"
    mysum mem.txt;sum=$((sum/=10))
    echo "The total %MEM used by R jobs on ${1} is ${sum}%"

    echo ""
  else
    echo "ERROR: **************************"
    echo "remoteRJobs MACHINE [number]"
    echo ""
  fi
}
```

Shell script running results

```
------------------------------2.(a)-----------------------------
remoteRJobs - query a machine to find all R jobs running
              and returns the CPU usage of those jobs with
              the most intensive jobs listed first

remoteRJobs MACHINE [number]
MACHINE    remote machine name
[number]   number of jobs listed
************************************************************
Test Case 1: remoteRJobs beren 10
The top 10 %CPU usage of R jobs running on beren:
PID UID %CPU CMD
13254 3262 76.4 R
13255 3262 76.3 R
13256 3262 76.2 R
11027 kpkim 65.6 R
11025 kpkim 64.0 R
11033 kpkim 60.5 R
11026 kpkim 60.2 R
15912 scf 59.6 R
11029 kpkim 58.7 R
11032 kpkim 57.8 R
```

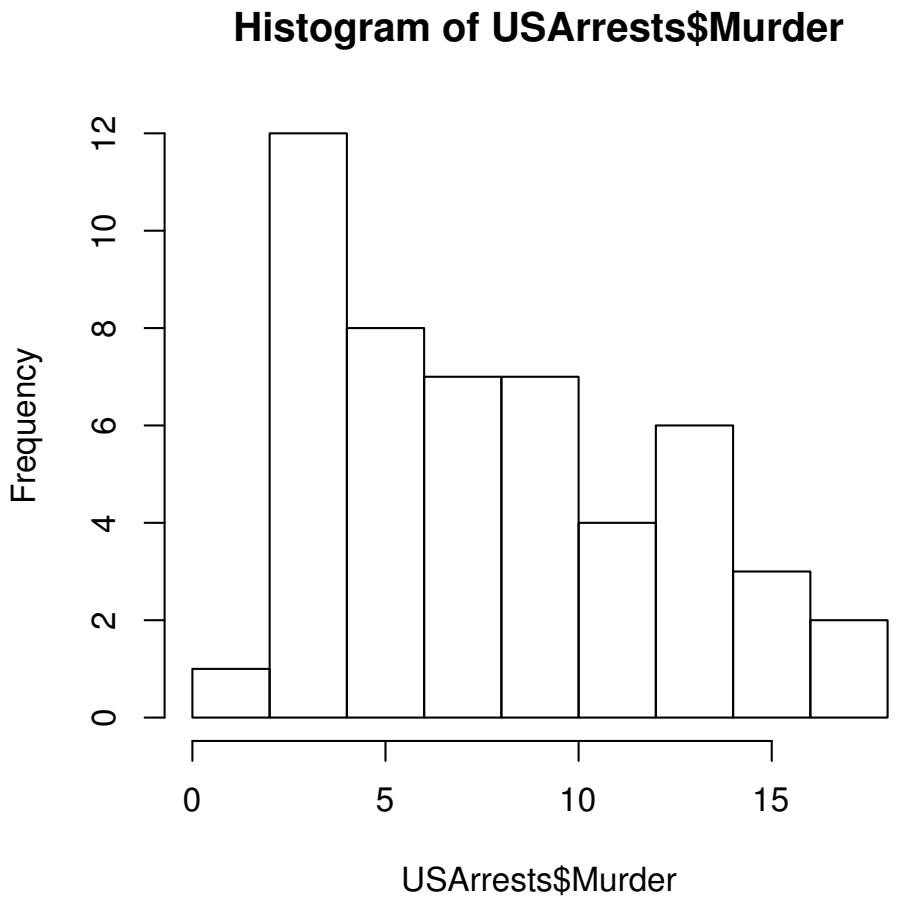
```
Test Case 2: remoteRJobs beren
The %CPU usage of R jobs running on beren:
PID UID %CPU CMD
13254 3262 76.4 R
13255 3262 76.3 R
13256 3262 76.2 R
11027 kpkim 65.5 R
11025 kpkim 63.8 R
11033 kpkim 60.3 R
11026 kpkim 60.1 R
15912 scf 59.6 R
11029 kpkim 58.5 R
11032 kpkim 57.7 R
11028 kpkim 56.2 R
11030 kpkim 55.0 R
 3498 decker 3.0 R
13262 kpkim 0.8 R
 5331 scf 0.0 R
13372 yuvalb 0.0 R
13236 3262 0.0 R
22774 vincent 0.0 R

Test Case 3: remoteRJobs
ERROR: **************************
remoteRJobs MACHINE [number]

------------------------------------------------------------

------------------------------2.(b)-----------------------------
remoteRJobs2 - query a machine to find all R jobs running
               and returns the CPU usage of those jobs with
               the most intensive jobs listed first;
               also add up CPU and memory use of these jobs

remoteRJobs2 MACHINE [number]
MACHINE    remote machine name
[number]   number of jobs listed
************************************************************
Test Case 1: remoteRJobs2 beren 10
The top 10 %CPU usage of R jobs running on beren:
PID UID %CPU %MEM CMD
13254 3262 76.4 0.0 R
13255 3262 76.3 0.0 R
13256 3262 76.2 0.0 R
11027 kpkim 65.6 1.4 R
11025 kpkim 63.8 1.4 R
11033 kpkim 60.3 1.4 R
11026 kpkim 60.0 1.4 R
15912 scf 59.6 1.0 R
11029 kpkim 58.5 1.4 R
11032 kpkim 57.7 1.4 R
The top 10 total %CPU used by R jobs on beren is 81% for 8 CPUs
The top 10 total %MEM used by R jobs on beren is 14%

Test Case 2: remoteRJobs2 beren
The %CPU usage of R jobs running on beren:
PID UID %CPU %MEM CMD
13254 3262 76.4 0.0 R
13255 3262 76.3 0.0 R
```

```
13256 3262 76.2 0.0 R
11027 kpkim 65.9 1.4 R
11025 kpkim 63.7 1.4 R
11033 kpkim 60.3 1.4 R
11026 kpkim 60.0 1.4 R
15912 scf 59.6 1.0 R
11029 kpkim 58.5 1.4 R
11032 kpkim 57.7 1.4 R
11028 kpkim 56.2 1.4 R
11030 kpkim 55.4 1.4 R
 3498 decker 3.0 1.7 R
13262 kpkim 0.8 1.3 R
 5331 scf 0.0 1.1 R
13372 yuvalb 0.0 0.0 R
13236 3262 0.0 0.1 R
22774 vincent 0.0 0.0 R
The total %CPU used by R jobs on beren is 96% for 8 CPUs
The total %MEM used by R jobs on beren is 16%

Test Case 3: remoteRJobs2 beren arwen 10
ERROR: **************************
remoteRJobs MACHINE [number]

------------------------------------------------------------
```

Problem 3

Crime rates in the US are high compared to European countries. Here I 'analyze' the variation in murder across US states using R. I show a histogram of rates of arrest for murder for the 50 states and find the states with the lowest and highest murder arrest rates.

```
hist(USArrests$Murder)
```




```
lowHi <- c(which.min(USArrests$Murder), which.max(USArrests$Murder))
attributes(USArrests)$row.names[lowHi]

## [1] "North Dakota" "Georgia"
```

The state with the lowest rate is North Dakota. The state with the highest rate is Georgia.