Week 4 study diary

Exercises:

1. ASCII art. Script for printing cat (in this case script and shortcat.txt are in the same directory):

```
#!/bin/bash

# Script that accepts a single numerical (greater than 0) command line parameter
and prints a cat made from ASCII characters, that has a stomach length equivalent
to the given parameter.

num=

if [ "$num" -lt "1" ]; then
   echo You have to give number greater than 0
    (exit 1)
elif [ "$num" == "1" ]; then
   echo $'\nShortcat\n\n\n'
   cat shortcat.txt
else
   echo $'\nLongcat' "$num" $'\n\n\n'
   head -9 shortcat.txt
 for ((i=0;i<num;i++))
  do
   sed -n '10p' shortcat.txt
 done

   tail -6 shortcat.txt
fi
```

1. Script for creating random numbers

```
#!/bin/bash

# Script takes a numerical parameter n and produces n lines with two random numbers.

count=

 for ((i=0;i<count;i++))
  do
    echo $RANDOM $RANDOM
 done
```

- Creating random-data file:

```
fuksi@dhcp-asv-103:~/LinuxFundamentals2016/Week4$ ./create-random-data.sh 100 | cat
> random-data.txt
```

- gnuplot script for printing data points and its output (output might look wrong in this file, but works 100% in terminal):

```
# gnuplot script for printing datapoints from random-data.txt file

set terminal dumb
set title "Datapoints"
set style data points
plot "random-data.txt" using 1:2 title ''
```

```
fuksi@dhcp-asv-103:~/LinuxFundamentals2016/Week4$ gnuplot datapoints.gnuplot


                                Datapoints

  35000 ++--------+--------+---------+--------+---------+--------+--------++
        +         +        +         +        A     A   +        +         +
        |      A   A      A      A             A    AA                      |
  30000 ++                                     A        A              A   ++
        |          A A                   A    A                         A    |
  25000 ++      A            A  A             A                            ++
        |                A   A      A           A            A  A          |
        A     A                 A      A        A                          |
  20000 ++                    A      A  A     A          A        AA       ++
        |A             A              A  A          A             AA       |
  15000 ++ A   A                     AA  A    A                           ++
        |  A         A           A    A   AA  A               A           |
        | A           A        A        A       A      A       A          |
  10000 ++                              A     A                        A   ++
        |         A  A    A                             A                 |
   5000 ++          A               A     A       AA             AA      ++
        |           A  A AA            A                 A A              |
        AA  A     +        +         +        A A     A +        +         +
      0 ++---------+A-------+---------+----A---+---------+--------A---------++
        0         5000    10000     15000    20000     25000    30000     35000
```

1. Let's plot some real data points. BASH and gnuplot scripts for assignment:

- BASH script

```
#!/bin/bash

# Script for finding maximum processor temperatures by date from November 2011
using lost24/monitor dataset.

tempfile="$(mktemp)"
dayfiles=~/lost24/monitor/2011.11.*

# loop for going through days of month
for dayfile in $dayfiles
  do
timefiles=${dayfile}/*/*temps.txt
max=0

# loop for going through times of a day
for timefile in $timefiles
  do
```

```
temp=$(grep 'PRO' $timefile | cut -c32-33)
if [ $max -lt $temp ]; then
  max=$temp
fi

done
echo $(echo $dayfile| cut -c30-) $max >> $tempfile
done

gnuplot -e "filename='$tempfile'" max-temps-2011-11.gnuplot
```

- gnuplot script

```
# gnuplot script for printing daily maximum processor temperatures from chosen data
set.

set terminal postscript eps size 3.5,2.62 enhanced color \
    font 'Helvetica,20' linewidth 2
set output 'max-temps-2011-11.eps'

set title "Daily maximum temperatures"
set xlabel 'Date'
set timefmt "%Y.%m.%d"
set xdata time
set xrange [ "11.11.01":"11.11.30" ]
set yrange [ 0 : ]
set ylabel 'Temperature'
set style data linespoints
plot filename using 1:2 notitle
```

- Let's put some context. As in previous assignment BASH and gnuplot scripts for temperatures (this time for minimum and maximum):
- BASH script

```
#!/bin/bash

# Script for finding minimum and maximum processor temperatures by date from
November 2011 using lost24/monitor dataset.

tempfile="$(mktemp)"
dayfiles=~/lost24/monitor/2011.11.*

# loop for going through days of month
for dayfile in $dayfiles
  do
timefiles=${dayfile}/*/*temps.txt
max=0
min=100

# loop for going through times of a day
for timefile in $timefiles
  do
temp=$(grep 'PRO' $timefile | cut -c32-33)
```

```
if [ $min -gt $temp ]; then
  min=$temp
fi

if [ $max -lt $temp ]; then
  max=$temp
fi

done
  echo $(echo $dayfile| cut -c30-) $min $max >> $tempfile
done

gnuplot -e "filename='$tempfile'" min-max-temps-2011-11.gnuplot
```

- gnuplot script

```
# gnuplot script for printing daily maximum processor temperatures from chosen data
set.

set terminal postscript eps size 3.5,2.62 enhanced color \
    font 'Helvetica,20' linewidth 2
set output 'min-max-temps-2011-11.eps'

set title "Daily minimum and maximum temperatures"
set xlabel 'Date'
set timefmt "%Y.%m.%d"
set xdata time
set xrange [ "11.11.01":"11.11.30" ]
set yrange [ 0 : ]
set ylabel 'Temperature'
set style data linespoints
plot filename using 1:2 title 'min' with linespoints,\
filename using 1:3 title 'max' with linespoints
```

- Let's generalize. Data that i downloaded is not named the same way as in assigment (example: ...../lost24/monitor/2011/10 for October 2011) but folder names are e.g. (.../lost24/monitor/2011.10.20) and (.../lost24/monitor/2011.11.21) so theres no folder structure for months, so for example these two are in the same directory. So instead after getting stuck for a while i chose to make script that takes the year and month parameter in yyyy.mm form. The BASH and gnuplot scripts for it:
- BASH script

```
#!/bin/bash

# Script for finding minimum and maximum processor temperatures by date from given
month of given year using lost24/monitor dataset. $yearmonth is parameter given in
form yyyy.mm

tempfile="$(mktemp)"
yearmonth=
outputtime=$(echo $yearmonth | sed 's/\./\-/')

dayfiles=~/lost24/monitor/$yearmonth*
```

```
# loop for going through days of month
for dayfile in $dayfiles
  do
timefiles=${dayfile}/*/*temps.txt
max=0
min=100

# loop for going through times of a day
for timefile in $timefiles
  do
temp=$(grep 'PRO' $timefile | cut -c32-33)

if [ $min -gt $temp ]; then
  min=$temp
fi

if [ $max -lt $temp ]; then
  max=$temp
fi

done
  echo $(echo $dayfile| cut -c30-) $min $max >> $tempfile
done

gnuplot -e "filename='$tempfile'; outp='min-max-temps-$outputtime.eps'"
min-max-temps-of-month.gnuplot
```

- gnuplot script

```
# gnuplot script for printing daily maximum processor temperatures from chosen data
set.

set terminal postscript eps size 3.5,2.62 enhanced color \
    font 'Helvetica,20' linewidth 2
set output outp

set title "Daily minimum and maximum temperatures"
set xlabel 'Date'
set timefmt "%Y.%m.%d"
set xdata time
set xrange [ * : * ]
set yrange [ * : ]
set ylabel 'Temperature'
set style data linespoints
plot filename using 1:2 title 'min' with linespoints,\
filename using 1:3 title 'max' with linespoints
```

1. Let's make more refined commands. BASH and gnuplot scripts for assignment:

- BASH script:

```
#!/bin/bash

# Script for finding minimum and maximum processor temperatures by date from given
month of given year using lost24/monitor dataset. $yearmonth is parameter given in
```

```
form yyyy.mm

tempfile1="$(mktemp)"
tempfile2="$(mktemp)"
yearmonth=${@: -1}
outputtime=$(echo $yearmonth | sed 's/\./\-/')

dayfiles=~/lost24/monitor/$yearmonth*

# loop for going through days of month
for dayfile in $dayfiles
  do
timefiles=${dayfile}/*/*temps.txt
max=0
min=100

# loop for going through times of a day
for timefile in $timefiles
  do
temp=$(grep 'PRO' $timefile | cut -c32-33)

if [ $min -gt $temp ]; then
  min=$temp
fi

if [ $max -lt $temp ]; then
  max=$temp
fi
done
  echo $(echo $dayfile| cut -c30-) $min >> $tempfile1
  echo $(echo $dayfile| cut -c30-) $max >> $tempfile2
done

# getopts arguments
while getopts "c:w:b:a:h" opt; do
  case $opt in
    c)
    gnuplot -e "filename1='$tempfile1'; outp='min-max-temps-$outputtime.eps';
terma='postscript default'" min-max-temps-of-month-getopts.gnuplot
      ;;
    w)
     gnuplot -e "filename2='$tempfile2'; outp='min-max-temps-$outputtime.eps';
terma='postscript default'" min-max-temps-of-month-getopts.gnuplot
      ;;
    b)
     gnuplot -e "filename1='$tempfile1'; filename2='$tempfile2';
outp='min-max-temps-$outputtime.eps'; terma='postscript default'"
min-max-temps-of-month-getopts.gnuplot
      ;;
    a)
         gnuplot -e "filename1='$tempfile1'; filename2='$tempfile2'; termd='dumb'"
min-max-temps-of-month-getopts.gnuplot
      ;;
    h)
      echo "Ohjelma tulostaa prosessorin lämpötilatiedot annetulta kuukaudelta
       (parametri oltava muotoa yyyy.kk) kaaviona .eps-tiedostoon .
        Argumentti vaihtoehdot:
```

```
        -c = kylmimmät
        -w = lämpimimmät
        -b = molemmat
        -a = kaavio tulostetaan ASCII-merkistönä terminaaliin
        (annettava ensimmäisenä jos annetaan useampi argumentti)
        -h = ohjeet

        esimerkkisyöte: ./min-max-temps-of-month-withgetopts.sh -a -b 2012.07"
      ;;
  esac
done
```

- gnuplot script

```
# gnuplot script for printing daily maximum processor temperatures from chosen data
set.

if (!exists ("termd")) set output outp

if (exists ("termd")) {
set terminal dumb
} else {set terminal postscript default}

set title "Daily minimum and maximum temperatures"
set xlabel 'Date'
set timefmt "%Y.%m.%d"
set xdata time
set xrange [ * : * ]
set yrange [ * : ]
set ylabel 'Temperature'
set style data linespoints

if (exists ("filename1") && exists ("filename2")) plot filename1 using 1:2 notitle
with linespoints,\
filename2 using 1:2 notitle with linespoints

if (exists ("filename1") && !exists ("filename2")) plot filename1 using 1:2 notitle
with linespoints

if (!exists ("filename1") && exists ("filename2")) plot filename2 using 1:2 notitle
with linespoints
```