Ceng328 Operating System_Lab4

Sed Statement

sed is basically a find and replace program. It reads text from standard input (e.g from a pipe) and writes the result to stdout (normally the screen).

The search pattern is a regular expression. This search pattern should not be confused with shell wildcard syntax.

To replace the string linuxfocus with LinuxFocus in a text file use:
cat text.file | sed 's/linuxfocus/LinuxFocus/' > newtext.file

This replaces the first occurance of the string linuxfocus in each line with LinuxFocus. If there are lines where linuxfocus appears several times and you want to replace all use:

```
cat text.file | sed 's/linuxfocus/LinuxFocus/g' > newtext.file
```

Cut Statement

```
Usage: cut [OPTION]... [FILE]...

Print selected parts of lines from each FILE to standard output.

 -b, --bytes=LIST    output only these bytes

 -c, --characters=LIST  output only these characters

 -d, --delimiter=DELIM  use DELIM instead of TAB for field delimiter

 -f, --fields=LIST    output only these fields; also print any line

       that contains no delimiter character, unless

       the -s option is specified

 -n           with -b: don't split multibyte characters

 -s, --only-delimited  do not print lines not containing delimiters

  --output-delimiter=STRING use STRING as the output delimiter

       the default is to use the input delimiter

  --help   display this help and exit

  --version output version information and exit
```

With no FILE, or when FILE is -, read standard input.

Examples:

1) $cut –c 4-7 file2

2) $cat > courses

```
    ceng112;ceng114;ceng102;

    ceng218;ceng212

    ^D

     $cut –f2 –d ';' courses

      ceng114

      ceng212

    $
```

3)

```
  a) # wc puts some space behind the output this is why we need sed:

    numofchar=`echo -n "Ceng328" | wc -c | sed 's/ //g' `

    # now cut out the last char

    rval=`echo -n "$1" | cut -b $numofchar`

  b) numofcharminus1=`expr $numofchar "-" 1`

    # now cut all but the last char:

    rval=`echo -n "$1" | cut -b 0-${numofcharminus1}`
```

Functions in Shell Script

```
functionname(){

    # inside the body $1 is the first argument given to the function

    # $2 the second...

    body

}
```

You need to "declare" functions at the beginning of the script before you use them.

Example: Write Script to find out biggest number from given three numbers. Numbers are supplies as command line argument. Print error if sufficient arguments are not supplied.

```
help(){

  cat << HELP

  findBiggest – find the biggest of the three numbers

  USAGE: findBiggest #1 #2 #3

  EXAMPLE: findBiggest 12 5 33

  HELP

  exit 0

}

# we have less than 3 arguments. Print the help text:

if [ $# -lt 3 ] ; then

   help

else

   max=$1

   for i in $2 $3

   if test $i –gt $max

   then

      max=$i

   fi

   echo "Biggest number is $max"

fi
```

Exercise1: Write a shell script that writes "I love operating system Lectures" into a file then ask user, the word that will be changed and the new word that will be replaced with the old one.

Ans:

```
echo –n "input file name:"; read inputfile

echo –n "output filename:"; read outputfile

echo "I love operating system lecture" > $inputfile

echo –n "Enter the word that you want to replace: "

read word1

echo –n "Enter the word that will be replaced with old one: ";

read word2

for I in 'cat $inputfile'

do

    if test $i=$word1

    then

         echo –n $word2 >> $outputfile

    else

         echo –n $i >> $outputfile

    fi

    echo –n '' >> $outputfile

done

echo –n "." >> $outputfile
```

Exercise2: Write a shell script that renames multiple files.

Ans:

```
OLD="$1"

NEW="$2"

# The shift command removes one argument from the list of

# commandline arguments.

shift

shift

# $* contains now all the files:

for file in $*; do

    if [ -f "$file" ] ; then

      newfile=`echo "$file" | sed "s/${OLD}/${NEW}/g"`

      if [ -f "$newfile" ]; then

        echo "ERROR: $newfile exists already"

      else

        echo "renaming $file to $newfile ..."

        mv "$file" "$newfile"

      fi

    fi

done
```

Question: Write a shell script that converts the binary numbers into its decimals equivalents.