**Shell Programing**

#!/bin/sh : first line

chmod 700 file.sh : execution permissions for file

./file.sh with arguments(or not) : to run

Syntax:

**If :** if [ spaces ]

then

else

elif

fi

**While:** while [ spaces ]

do

done

**For:** for i in (range)

do

done

**Tests:**  **-d file** = True if the file exists and is a directory.  
**-f file** = True if the file exists and is a regular file  
**-p file** = True if the file exists and is a named pipe.  
**-r file** = True if the file exists and is readable.  
**-w file** = True if the file exists and is writable.  
**-x file** = True if the file exists and is executable.  
**-z string** = True if the length of the string is 0.  
**-n string** = True if the length of the string is non-zero.  
**string1 = string2** = True if the strings are equal.  
**string1 != string2** = True if the strings are not equal.  
**! expr** = True if the expr evaluates to false.  
e**xpr1 –a expr2** = True if both expr1 and expr2 are true.  
**expr1 –o expr2** = True is either expr1 or expr2 is true.

**nr -op nr : op =** lt(less than), gt(greater than), ge (grater equal), eq (equal)… MAN TEST

**CAREFULL AT SPACES!!!!!!**

$variable = value

$1,2,3,4,5,6,7,8,9 – command line parameters (use shift for accessing more)

$\* - all arguments

$# - nr of command line arguments

$$ - The PID number of the process executing the shell.  
**$?** - Exit status variable

Sum = $(($sum+1)) = `expr sum + 1` (must have spaces otherwise string)

Echo: prints

Read: to read an argument

Variable = $(grep, awk, sed, …)

Variable=value (no spaces!!!!!!)

**Tip and tricks:**

**Parse files in director as parameter:** for file in “$dir”/\*

$file

**Check nr of parameters:** [ “$#” -eq 0 or smt ]

**Find dimension:** du -ch (-b for bytes) file\_name | cut -d “.” -1 (for integer !!!!)

du -ch $file | tail -1 | cut -d "." -f 1

**Number of words of first line:** `cat $filename | head -1 | wc -w`

PROBLEMA 6????

**Word occurences in file:** grep -o -i $word $ file | wc -l

**Going throw arguments with while:** Eval “word=\${$i}”

**Get path:** readlink -f file.txt

**Append string:** latex=${latex}newstring

**Read line by line:** input: path

while IFS= read -r line

do

done < “$input”

**Get file name**: name=$(echo "$filename" | cut -f 1 -d '.') (- f 2 for file extension)

**Processes:** ps-ef

**Columns of processes:** (ps -ef | awk 'NR > 1{print $1","$2}' | sort) (column 1 and 2 separeted by ,)

**Symbcdkh link**: $(find -L "$1"/\*) ($1 is a dir) (test -e if valid link)

**Change permissions:** chmod u=rwx,g=rwx, o=rwx file\_name

**Sum of all words in c file:** for F in `find $1 –type f –name "\*.c"`; do

N=`grep -E "[^ ]" $F | wc -l`

**C files recursivlly:** find $1 -type f -name "\*.c"

**Get last word in file:** awk 'END {print $NF}' file

**Replace:** echo `sed "y/aeiou/AEIOU/" $file`

While true

Do

Read n

If n = stop

break

Done

**Skip first line:** Awk {NR != 1} skip first line

“``”

List[@]

1. **Write a bash script that calculates the sum of the sizes (in bytes) of all regular files in a folder given as a parameter. (use test to check if the folder exists and if a given file is a regular file)**

#!/bin/sh

if [ "$#" -eq 0 ]

then

echo Give parameters

break

fi

if [ ! -d $1 ]

then

echo Not a directory

break

fi

sum=0

for file in "$1"/\*

do

if [ ! -f $file ]

then

echo $file is not a file

else

dim=` du -ch $file | tail -1 | cut -d "." -f 1 `

sum=$(( $sum + $dim | bc ))

fi

done

echo Sum is $sum

1. **Write a script that reads filenames until the word "stop" is entered. For each filename, check if it is a text file and if it is, print the number of words on the first line. (Hint: test command to check if regular file; file command to check if text file)**

#!/bin/sh

echo Give parameter:

read n

while [ "$n" != "stop" ]

do

if [ -f $n ]

then

echo Numarul de cuvinte de pe prima linie a fisierul $n este: `cat $n | head -1 | wc -w`

else

echo Not a file

fi

echo Give parameter:

read n

done

echo Done

1. Write a script that receives as command line arguments pairs consisting of a filename and a word. For each pair, check if the given word appears at least 3 times in the file and print a corresponding message.

#!/bin/sh

i=1

echo $#

while [ $i -lt $# ]

do

eval "word=\${$i}"

i=$(($i+1))

eval "file=\${$i}"

aparitii=`grep -o -i $word $file | wc -l`

if [ $aparitii -ge 3 ]

then

echo $word apare de cel putin 3 ori in $file

fi

i=$(($i+1))

done

1. Write a bash script that sorts all files given as command line arguments descending by size.(first check if an argument is a file)

#!/bin/bash

echo "" > out.txt

for file in $@; do

if [ -f "$file" ]; then

echo "$file" du -b "$file" >> out.txt

fi

done

1. Write a script that extracts from all the C source files given as command line arguments the included libraries and saves them in a file.(use the file command to check if a file is a C source file)

#!/bin/sh

if [ $# -lt 1 ]

then

echo Provide parameters pls

else

for file in $@

do

if [ ! -f $file ]

then

echo $file not a file

else

ext=$(echo $file | cut -d "." -f 2)

if [ $ext != c ]

then

echo $file not a c file

else

echo $(grep "#" $file | cut -d "<" -f 2 | cut -d ">" -f 1) >> output.txt

fi

fi

done

fi

sort -u output.txt >> output.txt

1. Write a script that monitors the state of a given folder and prints a message when something changed.
2. Find recursively in a given directory all the symbolic links, and report those that point to files/directories that no longer exist. Use option -L to test if a path is a symbolic link, and option -e to test if it exists (will return false if the target to which the link points does not exist)

#!/bin/sh

if [ $# -lt 1 ]

then

echo Provide parameters pls

else

for file in $@

do

if [ ! -f $file ]

then

echo $file not a file

else

ext=$(echo $file | cut -d "." -f 2)

if [ $ext != c ]

then

echo $file not a c file

else

echo $(grep "#" $file | cut -d "<" -f 2 | cut -d ">" -f 1) >> output.txt

fi

fi

done

fi

sort -u output.txt >> output.txt

1. Write a bash script that receives a folder name as argument. Find recursively in the folder the number of times each file name is repeated.

#!/bin/sh

if [ $# -lt 1 ]

then

echo 0 parameters given

else

if [ ! -d $1 ]

then

echo Not a directory

else

for file in "$1"/\*

do

if [ ! -f $file ]

then

echo $file not a file

else

echo $file

fi

done | sort | uniq -c

fi

fi

1. Calculate the average of all process ids in the system per user.

#!/bin/sh

sum=0

nr=0

prev=""

for pidd in $(ps -ef | awk 'NR > 1{print $1","$2}' | sort)

do

curr=$(echo $pidd | cut -d "," -f 1 )

id=$(echo $pidd | cut -d "," -f 2 )

if [ "$prev" != "$curr" ]

then

if [ $nr -gt 0 ]

then

echo The average for $prev is $((sum/nr))

fi

prev=$curr

nr=0

sum=0

else

sum=$(($sum+$id))

nr=$(($nr+1))

fi

done

1. Write a script that receives program/process names as command line arguments. The script will monitor all the processes in the system, and whenever a program with one of those names is run, the script will kill it and display a message. (see commands ps, kill, killall). Alternativ, comenzile pkill/pgrep pot fi folosite.
2. Write a script that receives a directory as a command line argument. The script will delete all the C source files from the directory and will display all other text files sorted alphabetically.

#!/bin/bash

if [ ! -d $1 ]

then

echo Not a directory

else

for file in "$1"/\*

do

extension=$(echo $file | cut -d "." -f 2)

echo $extension

if [ "$extension" = c ]

then

rm $file

fi

done

fi

1. Write a script that finds recursively in the current folder and displays all the regular files that have write permisions for everybody (owner, group, other). Then the script removes the write permissions from everybody. Hint: use chmod's symbolic permissions mode (see the manual).

#!/bin/sh

for file in "."/\*

do

if [ ! -f $file ]

then

echo $file not a file

else

perm=$(ls -l $file | cut -d " " -f 1 | grep -E "[w]{3}")

echo $perm

if [ "$perm" != "" ]

then

echo $perm

chmod u=rx,g=rx,o=rx $file

fi

fi

done

1. Consider a file containing a username on each line. Generate a comma-separated string with email addresses of the users that exist. The email address will be obtained by appending "@scs.ubbcluj.ro" at the end of each username. Make sure the generated string does NOT end in a comma.

#!/bin/sh

mail="@scs.ubbcluj.ro"

input="/home/scs/licenta/an1/gr913/gaie2925/shellprog/f5.txt"

stringfinal=""

i=0

virg=", "

while IFS= read -r line

do

if [ $i -eq 0 ]

then

stringfinal=${stringfinal}$line$mail

else

stringfinal=${stringfinal}", "$line$mail

fi

i=`expr i + 1`

done < "$input"

echo $stringfinal

echo Done

1. Write a shell script that recieves any number of words as command line arguments, and continuously reads from the keyboard one file name at a time. The program ends when all words received as parameters have been found at least once across the given files.
2. Dir with subdir with more than n files difrent than empty files

#!/bin/sh

if [ $# -lt 2 ]

then

echo Provide at least 2 arguments

exit 1

fi

nr\_subdir=$1

shift 1

while true

do

count=0

if [ ! -d "$1" ]

then

echo $1 not a directory

break

fi

for subdir in "$1"/\*

do

if [ -d $subdir ]

then

if [ ! -z $subdir ];then

count=$((count+1))

fi

fi

done

if [ $count -ge $nr\_subdir ];then

echo "$dir": has at least $nr\_subdir subdirectories

fi

if [ $# -ge 2 ];then

shift 1

else

break

fi

done

echo Done

**17.**

#!/bin/sh

if [ $# -lt 1 ];then

echo provide arguments

exit 1

fi

if [ $(($# % 3)) != 0 ];then

echo Provide triplets

exit 1

fi

while true

do

F=$1

N=$2

L=$3

if [ ! -f $F ];then

echo $F not a regular file

else

count=0

for line in `awk '{ print NF}' $F`;do

count=$((count+1))

if [ $count -eq $L ];then

#nr\_words=`grep -E " " $line | `

#echo $nr\_words

if [ "$line" = "$N" ];then

echo $F

break

fi

else

if [ $count -gt $L ];then

break

fi

fi

done

fi

if [ $# -ge 6 ];then

shift 3

else

break

fi

done

echo Done

**18.**

#!/bin/sh

if [ $# -lt 1 ]

then

echo Provide arguments

exit 1

elif [ $# -lt 2 ]

then

echo Provide a number and at least one directory

exit 1

fi

dimension=$1

shift 1

while true

do

for file in "$1"/\*

do

if [ ! -d "$1" ]

then

echo $1 not a directory

break

else

if [ ! -f $file ]

then

echo $file not a file

else

actual\_size=`du -b $file | awk '{print $1}'`

if [ $actual\_size -gt $dimension ]

then

echo $file

fi

fi

fi

done

if [ $# -ge 2 ]

then

shift 1

else

break

fi

done

echo Done

**17.Display all the mounted file systems who are either smaller than than 1GB or have less than 20% free space.**

#!/bin/sh

for dir in `find . -type d`

do

size=`du -sh $dir | awk '{print $1}' | cut -d "." -f 1 | cut -d "K" -f 1`

if [ $size -lt 1 ];then

echo $dir

else

free=`df -T $dir | awk '{print $6}'`

if [ "$free" != "Use% 80%" ];then

echo $dir

fi

fi

done

**18. Write a script that finds in a given directory hierarchy, all duplicate files and displays their paths. Hint: use checksums to detect whether two files are identical.**

#!/bin/sh

if [ $# -lt 1 ];then

echo No arg provided

exit 1

fi

if [ ! -d $1 ];then

echo Not a dir

exit 1

fi

for file in `find $1 -type f`

do

continut1=`cat $file`

for file2 in `find $1 -type f`

do

if [ "$file" != "$file2" ];then

continut2=`cat $file2`

if [ "$continut1" = "$continut2" ];then

echo `readlink -f $file`

fi

fi

done

done

echo Done

**19. model alina**

#!/bin/sh

while true

do

param=$1

if [ -d $param ];then

echo Size of directory $param is `du -b $param`

else

if [ -f $param ];then

p=$(ls -l $file | cut -d " " -f 1 | grep -E "[wr]")

if [ -n "$p" ];then

sed -e 's/\(-[0-9]\+\)/"ERROR"/g' $param >> $param

fi

else

#valid=`grep "[a-z]{4}[0-9]{4}" $param`

#if [ -n $valid ];then

if [[ $param =~ ^[a-z]{4}[0-9]{4} ]];then

fullname=`cat /etc/passwd | grep "$param" | cut -d":" -f 5`

if [ "$fullname" != "" ];then

echo Full name of $param is $fullname

login=`last $param | head -1 |grep "in"`

echo $login

if [ "$login" != "" ];then

echo Number of processes is `ps -ef | grep -o $param | wc -l`

else

echo `last $param | head -1 | awk '{print $7,$3,$10}'`

fi

else

echo user not found

fi

fi

fi

fi

if [ $# -ge 2 ];then

shift 1

else

echo Done

break

fi

done

**20.model**

#!/bin/sh

while true

do

param=$1

if [ -d $param ];then

echo Permissions for directory $param are `ls -l $param | cut -d " " -f 1`

else

if [ -f $param ];then

p=$(ls -l $file | cut -d " " -f 1 | grep -E "[wr]")

if [ -n "$p" ];then

sed -e 's/\(-[0-9]\+\)/"ERROR"/g' $param >> $param

fi

else

if [[ $param =~ ^[a-z]{4}[0-9]{4} ]];then

fullname=`cat /etc/passwd | grep "$param" | cut -d":" -f 5`

if [ "$fullname" != "" ];then

echo Full name of $param is $fullname

login=`last $param | head -1 |grep "in"`

echo $login

if [ "$login" != "" ];then

echo Number of processes is `ps -ef | grep -o $param | wc -l`

else

echo `last $param | head -1 | awk '{print $7,$3,$10}'`

fi

else

echo user not found

fi

fi

fi

fi

if [ $# -ge 2 ];then

shift 1

else

echo Done

break

fi

done

**21. model**

#!/bin/sh

if [ $# -lt 1 ];then

echo No arguments provided

exit 1

fi

if [ ! -d $1 ];then

echo Not a valid directory

exit 1

fi

for file in `find $1 -type f`

do

extensie=$(echo "$file" | cut -f 2 -d '.')

if [ "$extensie" != "c" ];then

echo $file

else

echo `grep -v "^#include" $file`

fi

done

echo Done