

Programare sh

Contents

1. Ce este un script sh (fișier de comenzi shell)?	1
2. Directivele (structurile de control) folosite în programarea shell	2
3. Exemplul 1: Verificare perechi fișier - lungime	3
4. Exemplul 2: Numărul mediu de linii:	3
5. Exemplul 3: Supravegherea conținutului unui grup de directoare	3
6. Exemplul 4: Selectare numai cuvinte din litere mici	4
7. Exemplul 5: Primul cuvânt de cel puțin 5 caractere.....	5
8. Exemplul 6: Lista fișierelor cu anumite drepturi:	5
9. Exemplul 7: Primele linii din fișiere cu cuvinte lungi:.....	6
10. Exemplul 8: Redenumirea fișierelor de un anumit tip și numărarea aparițiilor unui cuvânt	6
11. Exemplul 9: Apariția unor cuvinte în linii din fișiere	7
12. Exemplul 10: Reunirea fișierelor text.....	7
13. Analizați textul sursă al unui script.....	9
14. Probleme propuse	10

1. Ce este un script sh (fișier de comenzi shell)?

Un **script** (fișier de comenzi) este un fișier text care conține în el:

- comenzi Unix;
- directive (ale interpretorului de comenzi shell) de control al fluxului execuției acestor comenzi.

Unscript se comporta, la randul lui, ca și o comandă shell. **Numele unui fișier script nu trebuie să respecte nici o cerință sintactică!**

Noi vom adăuga la numele de scripturi sufixul **.sh** ca o convenție proprie de a ilustra conținutul. În general, un script este folosit pentru a manevra fișiere din sistemul de fișiere. Dacă se dorește intervenția în interiorul fișierelor, de regulă a celor text, se folosesc comenzi filtru auxiliare: `grep`, `sed`, `cut`, `awk` etc.

Pentru testarea unor condiții în structurile de control `if` sau `while` **se exploatează codul de retur al terminării comenzii precedente. Valoarea 0 a codului de retur reprezintă valoare de adevăr (TRUE), iar o valoare nenulă înseamnă FALSE.** De multe ori în condițiile din `if` sau `while` se folosește, comanda `test expresie` (notat uneori mai elegant `[expresie]`) care întoarce codul de retur în funcție de valoarea de adevăr a condiției din `test`.

Orice comandă shell se poate rula:

1. Direct la prompter în linia de comandă, așa cum am văzut în seminarul precedent
2. Comanda se scrie într-un script urmând a fi rulată odată cu execuția scriptului, de care ne ocupăm în cele ce urmează.

Dacă `script` este numele unui fișier de comenzi din directorul curent, rularea acestuia se poate face:

1. `./script ...` sau `caleabsoluta/script ...` dacă fișierul `script` are drepturi de execuție. Pentru fixarea drepturilor, în particular și a celor de execuție, se folosește comanda `chmod`.
2. `script ...` dacă `script` are drepturi de execuție și dacă directorul curent este înscris în `PATH`
3. `sh script ...` sau `sh caleabsoluta/script ...` indiferent dacă are sau nu drepturi de execuție.

Prin ... am notat: argumente ale comenzii, opțiuni, fișiere, expresii, redirectari: `< > >> <& >&`

Dacă o comandă trebuie reprezentată pe două (sau mai multe) linii consecutive, toate liniile componente, cu excepția ultimei, se încheie cu `\<ENTER>` (succesiunea de caractere `'\ ' \n'`).

O succesiune de două comenzi se separă prin: `;` `&&` `|| \n` (linie nouă) `|` (conectarea pipe a două comenzi) `&` (execuție în background). Mai multe comenzi pot fi grupate prin `(comenzi)` sau `{ comenzi }`

Într-o linie, dacă apare caracterul `#` (diez), atunci tot restul liniei este interpretat ca și **comentariu**. Este indicat (nu obligatoriu) ca scriptul shell să înceapă cu un comentariu special

```
#!/bin/sh  sau  #!/bin/bash  sau  #!/bin/ksh . . .
```

Prin aceasta se indică sub controlul cărui Shell se va interpreta scriptul: `sh`, `bash`, `ksh` etc. În cele ce urmează vom folosi numai scriptul `sh`.

2. Directivele (structurile de control) folosite în programarea shell

Structuri de control ale interpretorului `sh` (directivele) sunt: `if`, `case`, `for`, `while`, `until`. Pentru specificarea acestor structuri se folosesc cuvintele rezervate `if`, `then`, `elif`, `else`, `fi`, `case`, `in`, `esac`, `for`, `do`, `done`, `while`, `until`. Sintaxele acestor construcții sunt:

```
if listaCom then listaCom [elif listaCom then listaCom ]* [else listaCom ]? fi
case cuvânt in [ cuvânt [ | cuvânt ]* ) listaCom ;; ]+ esac
for nume do listaCom done
for nume in [ cuvânt ]+ do listaCom done
while listaCom do listaCom done
until listaCom do listaCom done
```

Prin *listaCom* am notat, generic, orice succesiune de comenzi, legate între ele prin `;` `|` `||` `&&`

În strânsă legătură cu structurile de control acest context se pot folosi comenzile speciale `true`, `false`, `break`, `continue`. De asemenea, o serie de comenzi standard Unix sunt utile în principal în scripturi shell: `shift`, `read`, `readonly`, `sleep`, `exit`, `echo`, `test` (echivalent cu `"[...]"`), `export`, `expr`, `basename`, (apostroafele inverse)

Semantica acestor construcții este similară cu cea întâlnită în limbajele de programare. Exemplele care urmează vor ilustra funcționarea lor.

3. Exemplul 1: Verificare perechi fisier - lungime

Sa se scrie un fisier de comenzi care primeste ca parametri perechi formate din nume de fisier si un numar. Pentru fiecare astfel de pereche se va verifica daca dimensiunea fisierului coincide cu numarul respectiv si se va afisa un mesaj corespunzator.

```
#!/bin/sh
# Sa se scrie un fisier de comenzi care primeste ca parametri perechi formate
# din nume de fisier si un numar. Pentru fiecare astfel de pereche se va
# verifica daca dimensiunea fisierului coincide cu numarul respectiv, si se va
# afisa un mesaj corespunzator.
while true; do
    if [ -z $1 ] || [ -z $2 ]; then break; fi
    fisier=$1
    numar=$2
    if [ ! -f $fisier ]; then continue; fi # nu e fisier
    if [ `echo $numar | egrep -c "[0-9]+"` -eq 0 ]; then continue; fi # NaN
    #lung=`ls -l $fisier | cut -d" " -f5`
    lung=`ls -l $fisier | awk '{print $5;}'`
    if [ `echo $lung | egrep -c "[0-9]+"` -eq 0 ]; then continue; fi # NaN
    if [ $numar -eq $lung ]; then echo "$fisier are lungimea $numar"; fi
    shift 2
done
```

4. Exemplul 2: Numărul mediu de linii:

Sa se scrie un script shell care primeste ca parametru un nume de director si va determina numarul mediu de linii din toate fisierele text din acest director si din toate subdirectoarele acestuia.

```
#!/bin/sh
# Sa se scrie un script shell care primeste ca parametru un nume de director si
# va determina numarul mediu de linii din toate fisierele text din acest
# director si din toate subdirectoarele acestuia.
if [ $# -ne 1 ]; then echo "trebuie dat un director" >&2; exit 1; fi
if [ ! \( -d $1 \) ]; then echo "$1 nu exista sau nu este director" >&2; exit 2; fi
TotalLinii=0
TotalFisiere=0
find $1 -type f -print | sort | while read Fisier t; do
    if [ `file $Fisier | grep -ci "ASCII text"` -eq 0 ]; then continue; fi
    linii=`wc -l <$Fisier` # cu $Fisier, da la iesire si numele fisierului
    TotalLinii=`expr $linii + $TotalLinii`
    TotalFisiere=`expr $TotalFisiere + 1`
    echo $TotalFisiere >/tmp/${LOGNAME}TotalFisiere
    echo $TotalLinii >/tmp/${LOGNAME}TotalLinii
done
TotalFisiere=`cat /tmp/${LOGNAME}TotalFisiere`
TotalLinii=`cat /tmp/${LOGNAME}TotalLinii`
rm /tmp/${LOGNAME}TotalFisiere /tmp/${LOGNAME}TotalLinii
medie=`expr $TotalLinii / $TotalFisiere`
echo "Linii" $TotalLinii "Fisiere" $TotalFisiere "Medie" $medie
```

5. Exemplul 3: Supravegherea continutului unui grup de directoare

Sa se scrie un script shell care monitorizeaza mai multe directoare. Monitorizarea se refera la aparitia unui fisier in aceste directoare. Numele directoarelor se dau la linia de comanda, iar numele fisierului urmarit se da de la tastatura. De asemenea, se da de la tastatura timpul, in secunde, intre doua cautari.

```
#!/bin/sh
# Sa se scrie un script shell care monitorizeaza mai multe directoare.
# Monitorizarea se refera la aparitia unui fisier in aceste directoare.
# Numele directoarelor se dau la linia de comanda, iar
# numele fisierului urmarit se da de la tastatura.
# De asemenea, se da de la tastatura timpul, in secunde, intre doua cautari.
echo -n "Introduceti numele fisierului: "; read nume
echo -n "Introduceti timpul (sec) intre cautari: "; read timp
while true; do
    find $* -type f -print | while read af; do
        nf=`echo $af | awk -F/ '{NF>1}{print $NF;}'`
        if [ "$nf" = "$nume" ]; then echo $nume "in" $af; fi
    done
    sleep $timp
done
```

Remarcam utilizarea in find a variabilei **\$*** care reuneste toate argumentele liniei de comanda, privite ca o lista de nume (**\$@** vede aceeasi lista ca un singur string).

6. Exemplul 4: Selectare numai cuvinte din litere mici

Sa se scrie un script care primeste la linia de comanda doua nume de fisiere: *intrare* *iesire*. Scriptul preia *intrare* si pune in *iesire* liniile din *intrare* din care se retin numai cuvintele formate din litere mici. In *iesire* se vor ordona alfabetic liniile si se vor elimina dublurile.

Evident, pot fi mai multe solutii. Noi vom da doua solutii, ambele folosind expresii regulate. O solutie foloseste awk, cealalta sed.

```
#!/bin/sh
# Sa se scrie un script care primeste la linia de comanda doua nume
# de fisiere: intrare iesire
# Scriptul preia intrare si pune in iesire liniile din intrare
# din care se retin numai cuvintele formate din litere mici.
# iesire se vor ordona alfabetic liniile si se vor elimina dublurile.

# varianta awk sort
awk <$1 '{for (i=1; i<=NF; i++)\
if ($i !~ /^[a-z]+$/) $i="";\
linie = ""; for (i=1; i<=NF; i++) if($i != "") linie = linie $i " ";\
if (linie != "") print substr(linie,1,length(linie)-1);}' |\
sort -u >$2.awk

# varianta sed sort
sed <$1 -e 's/\t/ /g'| # InlocuiesteTAB cu un spatiu \
sed -e 's/$/ /g'| # Adauga spatiu la sfarsit \
sed -e 's/^[^ ]*[a-z ][^ ]* / /g'| # Primul cuvant din linie \
sed -e 's/[^ ]*[a-z ][^ ]*$ / /g'| # Ultimul cuvant din linie \
sed -e 's/[^ ]*[a-z ][^ ]* / /g'| # Cuvintele terminate de spatiu \
sed -e 's/ / /g'| # Reduce numarul de spatii. De ce trebuie repetata? \
sed -e 's/ / /g'| # Reduce numarul de spatii \
sed -e 's/ / /g'| # Reduce numarul de spatii \
sed -e 's/ / /g'| # Reduce numarul de spatii \
```

```
sed -e 's/ / /g' | # Reduce numarul de spatii \
sed -e 's/^ //g' | # Sterge spatiul de la inceput \
sed -e 's/ $/g' | # Sterge spatiul de la sfarsit \
sed -e '/^$/d' | # Sterge liniile goale \
sort -u >$2.sed
```

7. Exemplul 5: Primul cuvânt de cel puțin 5 caractere

Din directorul curent, să se determine primul fișier text care conține o linie al cărei prim cuvânt are cel puțin 5 caractere. Exemplul este dat pentru a testa `break 2` (ieșirea din două cicluri interioare), care se pare că nu merge. Este suplinit prin `break` - uri succesive.

Selectarea primului cuvânt din linie se poate face în două moduri: folosind **cut** sau fara el.

```
#!/bin/sh
# Sa se caute in directorul curent primul fisier text care contine o linie
# in care primul cuvânt este mai scurt de 5 caractere.

for x in * ; do
    if [ `file $x | grep -ci "ASCII text" -eq 0 ] ; then continue ; fi
    #Variabila cuv1 retine primul cuvânt de pe o linie, delimitator spatiu
    cat $x | while read cuv1 t ; do
        #cat $x | cut -d" " -f1 | while read cuv1 ; do
            #verificam daca linia nu e vida, respectiv lungimea primului cuvânt
            if [ ! -z $cuv1 ] && [ `expr length $cuv1` -ge 5 ] ; then
                echo In $x s-a gasit $cuv1 cu lungimea `expr length $cuv1`
                #break 2 # Se iese din doua cicluri
                break # Se iese din ciclul while
            fi
        done
        break # Se iese din ciclul for
    done
done
```

8. Exemplul 6: Lista fișierelor cu anumite drepturi:

Sa se construiască un fișier de comenzi care primește ca parametru un nume de director (sa-l numim D) și un număr întreg (sa-l numim N). Pentru fiecare fișier din directorul D sau din subdirectoarele acestuia, pentru care userul are drepturi de citire și de execuție, sa se afișeze (maximum) primele N linii.

```
#!/bin/sh
# Sa se construiască un fișier de comenzi care primește ca
# parametru un nume de director (sa-l numim D) și un număr întreg (sa-l numim N).
# Pentru fiecare fișier din directorul D sau din subdirectoarele acestuia,
# pentru care userul are drepturi de citire și de execuție,
# sa se afișeze (maximum) primele N linii.

# verificam daca exista 2 parametrii in linia de comanda;
if [ ! $# -eq 2 ] ; then echo "usage: shell1.sh director numar"; exit 1; fi
# verificam daca primul parametru este nume de director
if [ ! -d $1 ] ; then echo "$1 nu este director\!"; exit 1; fi
D=$1 # Numai pentru a fi in ton cu enuntul. Putem folosi si $1
N=$2 # Numai pentru a fi in ton cu enuntul. Putem folosi si $2

# find $D -perm -u=rx -type f
```

```
# va afisa toate fisierele din directorul $D si din subdirectoare
# pentru care user-ul (u) are drept de citire (r) si de executie (x).
# O alternativa pentru comanda find ar fi test cu optiunile -f,-r, -x

for fis in `find $D -perm -u=rx -type f`; do
    echo $fis          # afisam numele fisierului
    head -$N $fis      # afisam primele $N linii din fisier
done
```

9. Exemplul 7: Primele linii din fişiere cu cuvinte lungi:

Sa se creeze un fisier care contine numele tuturor fisierelor text, dintr-un director dat ca parametru si din subdirectoarele lui, care au cuvinte mai lungi de **n** caractere, unde **n** se citeşte de la tastatura. Lista rezultata va fi ordonată alfabetic.

```
#!/bin/sh
# Sa se creeze un fisier care contine numele tuturor fisierelor text dintr-un
# director dat ca parametru si din subdirectoarele lui, care au cuvinte mai
# lungi de n caractere, unde n se citeşte de la tastatura.
# Fisierul rezultat va fi ordonat alfabetic.

if [ $# -lt 1 ] ; then echo "introduceti cel putin un argument" ; exit 1; fi
if [ ! -d $1 ] ; then echo "$1 nu e director" ; exit 2; fi
rm /tmp/${LOGNAME}numefisiere /tmp/${LOGNAME}rezultat >/dev/null 2>&1
echo -n "Introduceti n: "
read n
for fis in `find $1 -type f -print`; do
    if [ `file $fis | grep -c text` -ne 1 ] ; then continue; fi
    # variabilele n si fis din interiorul lui awk sunt altele decat cele din sh
    # La fel si $1 $2 . . .
    awk -v n=$n -v fis=$fis '\
{ for (i=1; i<=NF; i++) if (length($i) > n) print fis; }'\
<$fis >>/tmp/${LOGNAME}numefisiere # De ce am folosit /tmp ??
done
sort -u </tmp/${LOGNAME}numefisiere >/tmp/${LOGNAME}rezultat
cat /tmp/${LOGNAME}rezultat
```

10. Exemplul 8: Redenumirea fişierelor de un anumit tip si numararea aparitiilor unui cuvânt

Sa se scrie un script shell care primeşte 4 parametri: director, extensie1, extensie2, cuvânt. Scriptul va redenumi toate fisierele cu extensia extensie1 (dupa .) din director si subdirectoarele acestuia, dandu-le extensia extensie2. Va numara de cate ori apare cuvânt in fiecare fisier.

```
#!/bin/sh
# Sa se scrie un script shell care primeşte 4 parametri:
# director, extensie1, extensie2, cuvânt
# Scriptul va redenumi toate fisierele cu extensia extensie1 (dupa .)
# din director si subdirectoarele acestuia, dandu-le extensia extensie2.
# Va numara de cate ori apare cuvânt in fiecare fisier.
if [ $# -ne 4 ]; then echo "director, extensie1, extensie2, cuvânt" >&2; exit 1; fi
if [ ! \(`-d $1` \) ]; then echo "$1 nu exista sau nu este director" >&2; exit 2; fi
Total=0
```

```

find $1 -type f -print | sort | while read Fisier; do
    if [ `file $Fisier | grep -ci "ASCII text" -eq 0 `]; then continue; fi
    ext=`echo $Fisier | awk -F. '{NF>1{print $NF;}}`
    if [ -z $ext ] || [ $ext != $2 ]; then continue; fi
    nume=`echo $Fisier | awk -F. '{NF>1{print substr($0,1,length($0)-length($NF));}}`
    apare=`grep -ci $4 $Fisier`
    Total=`expr $Total + $apare`
    echo $Total>/tmp/${LOGNAME}Total
    echo "In" $Fisier $4 "apare de" $apare "ori"
    echo $Fisier "se va redenumi in" $nume$3
    #mv $Fisier $nume$3 # Aici re face redenumirea
done
echo $4 "apare in total de" `cat /tmp/${LOGNAME}Total` "ori"
rm /tmp/${LOGNAME}Total

```

Remarcam separarea partilor din numele absolut al unui fisier folosind awk.

11. Exemplul 9: Aparitia unor cuvinte in linii din fisiere

Sa se scrie un script care primeste la linia de comanda triplete: fisier, cuvant numar
Pentru fiecare astfel de triplet, se vor afisa toate liniile din fisier care contin cuvant exact de numar ori.

```

#!/bin/sh
# Sa se scrie un script care primeste la linia de comanda triplete:
# fisier, cuvant numar
# Pentru fiecare astfel de triplet, se vor afisa toate liniile
# din fisier care contin cuvant exact de numar ori.
while true; do
    if [ -z $1 ] || [ -z $2 ] || [ -z $3 ]; then break; fi
    fisier=$1
    cuvant=$2
    numar=$3
    if [ ! -f $fisier ]; then continue; fi # nu e fisier
    if [ `file $fisier | grep -c "text" -eq 0 `]; then continue; fi
    if [ `echo $numar | egrep -c "[0-9]+" -eq 0 `]; then continue; fi # NaN
    awk '{k=0; for (i=1; i<=NF; i++) if ($i == cuv) k++;\
        if (k==num) print FILENAME ":" $0;}' num=$numar cuv=$cuvant $fisier
    shift 3
done

```

12. Exemplul 10: Reunirea fişierelor text

Se cere un script **sh** care primeste la linia de comandă un nume de director. Se cere ca toate fişierele cu conţinut text din acest director şi din descendenţii lui să fie concatenate într-unul singur, într-o formă tipăribilă. Fişierul în care se reunesc să înceapă cu un cuprins, în ordine alfabetică, care reperează fişierele componente făcând repertoriizarea la nivel de linie sursă. Iată cum apare o porţiune din fişierul reuniune.

```

0 linii pana la ./b: ASCII text, with CRLF, LF line terminators
6 linii pana la ./c: ASCII text
9 linii pana la ./cl: POSIX shell script, ASCII text executable
38 linii pana la ./f: ASCII text
42 linii pana la ./f5: POSIX shell script, ASCII text executable
68 linii pana la ./lr: POSIX shell script, ASCII text executable
98 linii pana la ./pall: POSIX shell script, ASCII text executable

```

135 linii pana la ./pall.sh: POSIX shell script, ASCII text executable
 Total general: 172 linii in 8 fisiere.

```
=====
|| 0 linii pana la ./b: ASCII text, with CRLF, LF line terminators
=====
Sa se scrie un fisier de comenzi care preia un fisier de intrare dat ca
parametru si creeaza din el un alt fisier (al carui nume este dat ca
parametru) in care pastreaza doar cuvintele care contin litere mici. Fisierul
se va ordona alfabetic. Daca in rezultat exista linii consecutive identice, se
va pastra doar una dintre ele.

]======

=====
|| 6 linii pana la ./c: ASCII text
=====
Sa se scrie un script shell care monitorizeaza aparitia in toate directoarele
date ca si parametru in linia de comanda a unui nume de fisier citit de la
tastatura.
]======

=====
|| 9 linii pana la ./cl: POSIX shell script, ASCII text executable
=====

- - - - -

=====
|| 135 linii pana la ./pall.sh: POSIX shell script, ASCII text executable
=====

- - - - -

]======
Total general: 172 linii in 8 fisiere.
=====
```

Sursa scriptului este:

```
#!/bin/sh
# Sa se reuneasca intr-un fisier text, intr-o forma tiparibila,
# toate fisierele text din directorul $1.
# Fisierul text va incepe cu un cuprins al fisierelor continute.
if [ $# -ne 1 ]; then echo "folosire: pall director" >&2; exit 1; fi
if [ ! \( -d $1 \) ]; then echo "$1 nu exista sau nu este director" >&2; exit 2; fi
rm /tmp/${LOGNAME}Afisari /tmp/${LOGNAME}DeListat>/dev/null 2>&1
rm /tmp/${LOGNAME}/Total >/dev/null 2>&1
TotalLinii=0
find $1 -type f -print | sort | while read FISIER; do
    if [ `file $FISIER | grep -ci "ASCII text" ` -eq 0 ] >/dev/null 2>&1
    #if file $FISIER | egrep "exec|data|empty|reloc|cannot open" >/dev/null 2>&1
    then continue
    fi
    NrLinii=`wc -l <"$FISIER" ` # De ce este necesara delimitarea cu " ?
    Linie=${TotalLinii} linii pana la "`file $FISIER`
    echo $Linie >/dev/tty
    echo $Linie >> /tmp/${LOGNAME}DeListat
```



```

echo "\n\n===== " >> /tmp/${LOGNAME}Afisari
echo "|| "$Linie >> /tmp/${LOGNAME}Afisari
echo "===== [" >> /tmp/${LOGNAME}Afisari
cat $FISIER >> /tmp/${LOGNAME}Afisari
#pr -f $FISIER >> /tmp/${LOGNAME}Afisari
echo "]"===== " >> /tmp/${LOGNAME}Afisari
TotalLinii=`expr $TotalLinii + $NrLinii`
echo $TotalLinii >/tmp/${LOGNAME}Total # De ce este necesar ?
done
TotalLinii=`cat </tmp/${LOGNAME}Total`
NrFisiere=`wc -l </tmp/${LOGNAME}DeListat`
Linie="Total general: $TotalLinii linii in $NrFisiere fisiere."
echo $Linie >/dev/tty
echo $Linie >>/tmp/${LOGNAME}DeListat
echo $Linie >>/tmp/${LOGNAME}Afisari
echo "===== " >> /tmp/${LOGNAME}Afisari
cat /tmp/${LOGNAME}Afisari >>/tmp/${LOGNAME}DeListat
rm /tmp/${LOGNAME}Afisari /tmp/${LOGNAME}Total

```

Pentru a nu perturba culegerea de fișiere, fișierele auxiliare necesare sunt memorate în directorul /tmp, prefixate cu numele userului (/tmp este directorul tuturor userilor). Mai întâi se verifică corectitudinea parametrului, apoi se sterg eventualele fișiere temporare vechi. Construcția >/dev/null 2>&1 ascunde fișierele de ieșire și de erori standard. Lista tuturor fișierelor din director este dată de find, iar în ciclul while se lipesc fișierele de tip text. Decizia că un fișier conține text este obținută prin file și grep. Câteva variabile de mediu împreună cu efectul comenzii wc realizează calculele de numere de linii aferente. Derularea comenzii se vede pe tty, indiferent de ieșirea standard a scriptului. Fișierul /tmp/\${LOGNAME}DeListat conține reuniunea fișierelor cu conținut text din \$1.

13. Analizați textul sursă al unui script

Considerând că directorul **DIR** conține o ierarhie de subdirectoare și fișiere text răspundeți la următoarele întrebări despre scriptul Shell UNIX de mai jos:

- Ce va conține fișierul 1.txt după rularea scriptului?
- Ce va conține fișierul 2.txt după rularea scriptului?
- Explicați în detaliu expresiile regulate de pe liniile 2 și 5.

1	for f in `find DIR -type f`; do
2	if grep -q "^[^0-9]" \$f; then
3	echo \$f >> 1.txt
4	fi
5	if ! grep -q "[a-z]\$" \$f; then
6	echo \$f > 2.txt
7	fi
8	done

Răspuns:

- Căile (relative la DIR) ale tuturor fișierelor care conțin linii care nu încep cu cifră.
- Calea (relativă la DIR) a ultimului fișier găsit de comanda find care nu conține linii terminându-se cu literă mică.

Linia 2: început de linie, interval de cifre negat. Linia 5: interval de litere mici, sfârșit de linie

14. Probleme propuse

1. Răspundeți la următoarele întrebări, considerând o rulare a scriptului shell de mai jos:

- De câte ori se afișează "OK"? Justificați răspunsul.
- Care e valoarea variabilei f?
- Care e valoarea variabilei d?
- Care e valoarea variabilei x?
- Care e valoarea variabilei y?

```

1 f=`find . -type f`
2 d=`find . -type d`
3 for x in $f; do
4     for y in $d; do
5         if [ $x = $y ]; then
6             echo "OK"
7         fi
8     done
9 done

```

2. Se d fișierul **abc.sh** conținând scriptul Shell UNIX de mai jos. Răspundeți la următoarele întrebări:

- Ce se întâmplă dacă scriptul este rulat fără argumente?
- Ce va tipări pe ecran rularea `./abc.sh f3` și ce fișiere (nume și conținut) va crea, dacă f3 conține "abc 74 2-8 aa 3a =c b2" și rularea se face într-un director conținând doar fișierele `abc.sh` și `f3`?
- Dați un exemplu de fișier f3 astfel încât rularea de la punctul precedent să creeze 4 fișiere noi astfel încât numele niciunuia să nu aibă prefixul f3.

```

1 n=0
2 for i in `cat $1`; do
3     c=`echo $i|cut -c1`
4     if echo $i|grep -q "^[0-9][0-9]*$"; then
5         echo $i >> $1.nr
6     elif echo $c|grep -q "[A-Za-z]"; then
7         echo $i >> $c
8     else
9         n=`expr $n + 1`
10    fi
11 done
12 echo $n

```

3. Sa se creeze doua fisiere, unul care va contine lista tuturor fisierelor dintr-un director si din subdirectoarele acestuia, iar al doilea lista tuturor subdirectoarelor. Fisierele vor fi ordonate dupa dimensiune, iar directoarele alfabetic.

4. Sa se calculeze suma fiecărei coloane de numere din oricate fisiere. Pentru fiecare fisier se vor afisa aceste sume, numarul maxim de coloane si numarul de linii.

5. Sa se scrie un fisier de comenzi care preia un fisier de intrare dat ca parametru si creeaza din el un alt fisier (al carui nume este dat ca parametru) in care pastreaza doar vocalele. Daca in rezultat exista linii consecutive identice, se va pastra doar una dintre ele.

6. Sa se scrie un fisier de comenzi care are ca parametri triplete formate dintr-un nume de fisier si doua cuvinte. Pentru fiecare astfel de triplet, se va inlocui in fisier ultima aparitie din fiecare linie a primului cuvânt cu cel de-al doilea cuvânt.

7. Sa se scrie un fisier de comenzi care preia un fisier de intrare dat ca parametru si creeaza din el un alt fisier (al carui nume este dat ca parametru) in care pastreaza doar cuvintele care contin litere mici. Fisierul se va ordona alfabetic, si se vor semnala liniile consecutive identice.
8. Sa se afiseze, pentru fiecare fisier din linia de comanda, cuvantul care apare de cele mai multe ori. Afisarea se va face in ordine descrescatoare a numarului de aparitii.
9. Sa se scrie un fisier de comenzi care va afisa toate fisierele dintr-un director si din subdirectoarele acestuia asupra carora au drepturi de scriere toate cele trei categorii de utilizatori. Aceste fisiere vor fi apoi redenumite, adaugandu-se sufixul .all.
10. Sa se afiseze, pentru fiecare fisier din linia de comanda, numarul liniei care apare de cele mai multe ori, afisarea facandu-se in ordinea descrescatoare a numarului de aparitii.
11. Sa se scrie un fisier de comenzi care va afisa toate numele de fisiere dintr-un director dat ca parametru si din subdirectoarele sale, care au numele mai scurte de 8 caractere. Pentru acestea, daca sunt fisiere text, li se vor afisa primele 10 linii.
12. Sa se afiseze, pentru fiecare fisier din linia de comanda, cuvantul care apare de cele mai multe ori intr-o aceeași linie. Afisarea se va face in alfabetica dupa numele fisierului.
13. Sa se scrie un fisier de comenzi care creeaza un fisier care va contine toate fisierele dintr-un director dat ca parametru si din subdirectoarele sale pentru care membrii grupului nu au nici un fel de drept. Apoi, pentru aceste fisiere, se va da drept de scriere pentru membrii grupului.
14. Sa se scrie un fisier de comenzi care are ca parametri triplete formate dintr-un nume de fisier, un cuvânt si un numar k. Pentru fiecare astfel de triplet, se vor afisa toate liniile fisierului care contin cuvântul respectiv exact de k ori.
15. Pentru toate fisierele text date in linia de comanda, sa se elimine toate liniile care contin un anumit cuvânt apoi sa se afiseze liniile comune acestor fisiere.
16. Pentru fiecare parametru din linia de comanda, daca el reprezinta un fisier, se va afisa cuvântul de lungime maxima. Afisarea se va face in ordine descrescatoare a lungimii cuvântului. Ceilalti parametri se vor scrie intr-un fisier cu numele 'eronat'.
17. Se cere un fisier de comenzi care afiseaza toate numele de fisiere dintr-un director dat ca parametru si din subdirectoarele acestuia al caror nume se termina in .me. Afisarea se va face in ordinea creerii lor. Pentru toate aceste fisiere, daca sunt fisiere executabile, se vor lansa in executie.
18. Sa se scrie un program de supraveghere care afiseaza pe ecran toti utilizatorii care lanseaza comanda 'who'.
19. Sa se creeze un fisier care contine numele tuturor fisierelor text dintr-un director dat ca parametru si din subdirectoarele acestuia care au cuvinte mai lungi de 15 caractere. Fisierul va fi ordonat alfabetic.
20. Sa se scrie un fisier de comenzi care are ca parametri cvadruple formate dintr-un nume de fisier, doua cuvinte si un numar k. Pentru fiecare astfel de cvadruplu, se va inlocui in fisier a k-a aparitie din fiecare linie a primului cuvânt cu cel de-al doilea cuvânt.

21. Se da un fisier care contine pe fiecare rand un nume urmat de 5 note. Se cere sa se construiasca un al doilea fisier care contine numele urmat de medie. Acest fisier va fi ordonat descrescator in functie de medie. Se va semnala daca un anumit nume apare pe mai multe linii in fisierul initial.
22. Sa se afiseze, dintr-o lista de fisiere, numele celui care contine numarul maxim de cuvinte distincte pe o linie. Totodata, se va crea un fisier care contine, pentru fiecare fisier, numarul liniei care are lungime maxima. Liniile fisierului vor fi ordonate descrescator dupa lungimea liniei.
23. Sa se scrie un program de supraveghere, care semnaleaza momentul in care se sterge un fisier dintr-un director dat ca parametru si din subdirectoarele acestuia. Se va afisa ora stergerii si dimensiunea fisierului.
24. Sa se afiseze, pentru fiecare fisier din linia de comanda, numarul de cuvinte care au lungimea mai mare decat un numar k , citit de la tastatura. Afisarea se va face decrescator dupa numarul de cuvinte.
25. Sa se scrie un fisier de comenzi care primeste ca parametri fie nume de fisiere care exista, fie numere. Fiecare fisier se va afisa pe ecran (daca este fisier text, in caz contrar se va da un mesaj), iar pentru fiecare numar n se va crea un fisier 'f.n' care contine textul "Acest fisier nu a existat" urmat de patratul numarului respectiv.
26. Pentru fiecare fisier din linia de comanda se vor afisa toate liniile care sunt mai lungi de 10 caractere. De asemenea, se vor inlocui toate cifrele cu caracterul 'a'. Liniile unui fisier vor fi precedate de numele fisierului. Se va face o situatie finala cu numarul liniilor afisate din fiecare fisier, in ordine descrescatoare in functie de numarul liniilor.
27. Sa se scrie un fisier de comenzi care primeste ca parametri perechi formate din nume de fisier si un numar. Pentru fiecare astfel de pereche se va verifica daca dimensiunea fisierului coincide cu numarul respectiv, si se va afisa un mesaj corespunzator.
28. Sa se scrie un fisier de comenzi care primeste ca parametri perechi formate din nume de fisier si un numar. Pentru fiecare astfel de pereche se vor afisa toate liniile care au lungimea mai mare decat numarul respectiv. Liniile unui fisier vor fi precedate de numele fisierului. Se va face o situatie cu numarul liniilor afisate din fiecare fisier, ordonat descrescator in functie de acest numar.
29. Se da un fisier de intrare care contine cuvinte. Unele cuvinte sunt delimitate de caracterele "<" ">". Sa se afiseze fisierul, mai putin cuvintele marcate intre "<" si ">".
30. Se da un fisier care are pe fiecare linie mai multe cuvinte. Fiecare dintre aceste cuvinte contine 0 sau mai multe caractere "." Sa se elimine din cuvintele de rang par caracterele ".". Cuvintele sunt despartite de spatii si pot contine orice caracter.
31. Sa se determine folosind comanda ping toate calculatoarele din reseaua scs care sunt pornite stiind ca aceste calculatoare au adresa ip de forma 192.168.144.x unde $x = 1, 254$.
32. Sa se determine din utilizatorii intrati in sistem in ultima luna, pe aceia care sunt in grupul ccubb (se poate folosi comanda last)
33. Sa se scrie un program shell care prelucreza un fisier si pastreaza in acest fisier doar cuvintele care contin doar litere mici.
34. La toate fisierele dintr-un director dat ca parametru sa li se adauge la sfarsit o linie care sa contina dimensiunea fisierului (impreuna cu dimensiunea informatiei care se adauga) si data ultimei modificari

(ziua, ora si anul). Sa se testeze pentru directorul respectiv daca dupa o perioada de timp sau modificat fisiere.

35. Sa se sorteze toti utilizatori conectati in sistem descrescator dupa idle (timpul care au fost inactivi).

36. Sa se determine daca exista doi utilizatori conectati la sistem de la o aceeaasi statie.

37. Folosind comanda last sa se determine numarul de statii distincte de la care sau realizat conectari la sistem, si numarul de conectarii de la fiecare statie.

38. Sa se determine numarul de utilizatori distincti care lucreaza in sistem si numarul de sesiuni ale fiecarui utilizator.

39. Folosind comanda df sa se determine spatiul ocupat pe disc (pe o anumita partitie). Sa se determine numarul de fisiere de pe partitia corespunzatoare si dimensiunea medie a unui fisier.

40. Sa se faca topul primilor 20 de utilizatori din sistem din punct de vedere al dimensiunii cutiei postale. Acestora sa li se trimita un mail de atentionare.

41. Sa se determine utilizatorii din sistem care au drepturi pe directorul personal si pentru "others". Daca exista cel putin doua din cele trei drepturi setate sa li se trimita acestor utilizatori un mesaj de avertizare.

42. Sa se determine utilizatorii din sistem care au in directorul personal un director cu numele public_html. Sa se afiseze numarul total de utilizatori care au acest director si cat la suta din utilizatorii din sistem au acest director.

43. Sa se determine numarul de utilizatori distincti care au intrat in sistem in ultima luna, precum si numarul de conectari pentru fiecare. Sa se sorteze utilizatori dupa numarul de conectari.

44. Sa se determine numarul de utilizatori distincti care au intrat in sistem in ultima luna, precum si timpul total de conectare pentru fiecare. Sa se sorteze utilizatori dupa acest timp.

45. Sa se determine subdirectoarele dintr-un director dat ca parametru care contin un fisier care ocupa mai mult de jumatate din dimensiunea subdirectorului respectiv.

46. Sa se faca topul subdirectoarelor dintr-un director dat ca parametru dupa raportul "numar de fisiere continute" / "dimensiune subdirector".

47. Sa se scrie un program shell care pentru fiecare fisier cu drepturile 755 dintr-un director (si subdirectoarele sale) dat ca parametru ii schimba drepturile de acces in 744. Inainte de a schimba drepturile de acces, pogramul cere confirmare din partea utilizatorului (pentru fiecare fisier in parte).

48. Sa se scrie un script care primeste ca si parametru un nume de utilizator si cauta toate legaturile simbolice detinute de acel utilizator. Programul va numara aceste legaturi simbolice, afisand numarul lor pe ecran si din multimea aceasta de legaturi simbolice va redenumi in 'leg-simbolica' jumatate dintre ele (de pilda cele pare).

49. Sa se scrie un shell care primeste ca parametrii in linia de comanda un nume de director si oricate fisiere. Programul va verifica care dintre aceste fisiere sunt detip text si pentru acestea din urma va afla numarul de caractere pe care le contine. Daca numarul de caractere este par, programul va muta fisierul respectiv in directorul desemnat ca prim parametru.

50. Sa se scrie un program care va supraveghea toate conexiunile de la o masina (server) dat ca parametru in linia de comanda si va scrie intr-un fisier timpul (data) la care s-a realizat conexiunea. Daca in plus, portul la care se conecteaza masina pe serverul local este portul 23, programul va trimite un mail la o adresa (care vreti voi;) in care va scrie numele masinii (de la distanta) si timpul cand s-a realizat conectarea. (indicatie: se foloseste comanda `netstat`).

51. Sa se scrie un shell care primeste ca si parametru un nume de director. Programul va examina fiecare fisier din director (nu si subdirectoare) si va face urmatorul lucru: Fie fisierul 'exemplu.txt' care are N caractere. Programul va face in locul fisierului 'exemplu.txt' doua fisiere 'exemp' si 'lu.txt' (de fapt desparte numele original in doua), iar 'exemp' va contine primele N/2 caractere din 'exemplu.txt', iar 'lu.txt' va contine ultimene N/2 caractere din fisieul original 'exemplu'. Nota:se pot scrie mai multe fisiere, scenariu awk,..nu este obligatoriu sa se scrie un singur program shell.

54.Sa se scrie un program shell care numara toate fisierele cu drept de scriere pentru group si others dintr-un director care este dat in linia de comanda si subdirectoarele sale si numara si directoarele cu drept de executie pentru others din acest director si subdirectoarele sale.

55.Sa se scrie un script shell care tot citeste utilizatori din linia de comanda si pentru fiecare utilizator afiseaza numele real al acestuia, numarul de procese (daca are) si ce comenzi ruleaza.

56.Sa se scrie un program shell care primeste ca si parametru un nume de director si un fisier. Programul va afisa toate fisierele din director si subdirectoarele sale care au extensia .c si care sunt mai noi decat fisierul dat ca al doilea parametru.

57. Se cere un fisier de comenzi care are in linia de comanda un nume de fisier f si un numar n. Se cere crearea fisierului f prin concatenarea fisierelor f.1, f.2, ..., f.n. Daca unul din aceste fisere nu exista el va trebui creat si va trebui sa contina textul "Eroare: acest fisier nu a existat inainte de executia comenzii".

58. Pentru fiecare fisier ASCII dintr-un director dat ca parametru si din toate subdirectoarele lui, se vor afisa primele zece linii care contin un anumit text (dat ca parametru).