Programare de sistem

Lucrare de laborator nr. 4

Permisiuni. Comenzi. Expresii regulate.

Acest laborator are drept scop familiarizarea cu setarea şi efectele permisiunilor, cu expresiile regulate, cu procedurile si tehnicile de filtrare de informaţii. Obiectivul de a putea avea acces rapid la informaţii despre fişierele şi directoarele de pe staţia dumneavoastră de calcul este pe deplin justificat data fiind abundenta de date rezidente pe orice calculator.

Realizaţi un director LAB4 în care veţi depune rezolvările tuturor exerciţiilor propuse.
(`mkdir LAB4`)

Conţinut:

- Permisiuni: comanda `chmod`
- Comanda shell UNIX `cut`
- Expresii regulate
- Comanda shell UNIX `wc`
- Comanda shell UNIX `find`
- Filtrarea conţinuturilor de fişiere şi directoare

1. Permisiuni UNIX

Permisiunile (drepturile de acces) fac parte din sistemul de securitate al SO UNIX şi necesita o atenţie sporita din partea utilizatorilor. Comanda de baza pentru setare/modificare a permisiunilor este `chmod`, vedeţi pagina ei de manual pentru a înţelege cum funcţionează ea. Realizaţi în continuare exerciţiul ce urmează şi lămuriţi ce se întâmplă

Exerciţiul 1 :

1. Deplasaţi-vă în dosarul `~/LAB4`, creaţi un director cu numele `tmp` şi poziţionaţi drepturile de acces în felul următor: `rwx r-x ---`.
2. Creaţi fişierul `test` cu comanda `touch` in dosarul `tmp`, introduceţi in el câteva linii de text, setaţi drepturile de acces `rw- r-- ---` şi listaţi conţinutul acestui fişier.
3. Retrageţi pentru proprietar dreptul de a citi fişierul `test` şi încercaţi să deschideţi fişierul.
4. Retrageţi pentru proprietar dreptul de scriere in dosarul `tmp` şi încercaţi sa ştergeţi fişierul `test`.
5. Retrageţi pentru proprietar dreptul de citire pentru `tmp` şi încercaţi sa afişaţi lista de fişiere din acest dosar.
6. Retrageţi pentru proprietar dreptul `x` pentru `tmp` şi încercaţi sa va poziţionaţi în acest dosar.
7. Setaţi pentru proprietar şi pentru grup drepturile de acces `rwx` pentru dosarul `tmp`. Listaţi drepturile setate pentru dosarul `tmp` şi pe cele ale fişierului `test`. Ce remarcaţi?
8. Creaţi dosarul `abc` în dosarul `tmp` şi creaţi fişierul `file1` în acest dosar. Listaţi permisiunile pentru elementele nou create şi explicaţi de ce au fost ele setate aşa. De ce depind aceste setări?
9. Setaţi recursiv printr-o singura comanda drepturile `r-x r-x r--` pentru dosarul tmp şi pentru toate elementele (fişierele şi subdirectoarele) pe care le conţine el. Listaţi setările şi încredinţaţi-vă ca ele au fost modificate după cum a fost cerut.

10. Încercaţi sa setaţi permisiunea de scriere in dosarul /home. Reuşiţi? De ce ? Concluzii.
11. Creaţi un nou utilizator `user_nou` şi atribuiţii cu comanda `chown` rolul de proprietar al fişierului `tmp/abc/file1`. Încercaţi apoi, fără a schimba userul, sa modificaţi pentru acest fişier drepturile de acces `rw- rw- r--` Ce se întâmplă? Ştergeţi acest fişier devenind root dacă e necesar.

2. Comanda shell UNIX `cut`

Comanda cut este utilizata pentru a selecta anumite unităţi dintr-unul sau mai multe fişiere. Aceste unităţi pot fi caractere, cuvinte sau unităţi definite de către utilizator.

Sintaxa generala a comenzii este :

```
cut  opţiuni [fişier(e)]
```

Principalele opţiuni ale acestei comenzi sunt următoarele :

-c lista – extragere in mod caracter
-f lista – extragerea câmpurilor specificate în lista
-d c – permite specificarea caracterului utilizat ca si separator în fişierul text intre diferitele entităţi (în mod implicit este considerat separator caracterul tab)

Formatul listei poate fi de forma :

n	a n-a unitate
n, m	doar a n-a unitate si a m-a unitate
n-m	de la a n-a unitate la a m-a unitate
n-	de la a n-a unitate pana la sfarsit
-n	de la prima unitate la a n-a unitate

Următoarele exemple vor clarifica detaliile prezentate ale acestei comenzi.

Exemple de utilizare a comenzii cut

Creaţi un subdirector Cut, in directorul LAB4. Sa se creeze fişierul **Cumparaturi.txt** asa cum este definit mai jos :

```
2kg  Lapte      4   lei
1    Ciocolata  3   lei
1kg  Faina      1.5 lei
1    Unt        2.5 lei
1    Biscuiti   3.5 lei
```

Acest fişier are datele grupate pe patru coloane după cum urmează : prima coloana indică cantitatea, a doua denumirea produsului, a treia valoarea iar a patra moneda de plata.

Pentru a selecta primul caracter de pe fiecare linie se va folosi comanda :

```
$ cut -c 1 Cumparaturi.txt
2
1
1
1
1
```

Din aceasta captura de terminal se poate vedea şi răspunsul pe care l-a dat shell-ul. Comanda de mai sus realizează o extragere de caractere (opţiunea -c), mai exact a primului caracter (aceasta este semnificaţie lui 1).
Din calcule asupra ordonării datelor în fişierul de mai sus se poate remarca faptul că denumirea produselor se afla intre caracterele 5 şi 15. O selecţie a acestor caractere se poate realiza cu ajutorul următoarei comenzi :

```
$ cut -c 5-15  Cumparaturi.txt
Lapte
Ciocolata
Faina
Unt
Biscuiti
```

Comanda de mai sus realizează o extragere de caractere (opţiunea -c), mai exact a caracterelor aflate intre poziţiile 5 şi 15 inclusiv (aceasta sunt semnificaţiile lui 5-15).

Pentru a realiza o imprimare a produselor şi cantităţilor ce urmează a fi cumpărate se poate folosi comanda :

```
$ cut -c -15  Cumparaturi.txt
2kg Lapte
1   Ciocolata
1kg Faina
1   Unt
1   Biscuiti
```

Comanda de mai sus realizează o extragere de caractere (opţiunea -c), mai exact a tuturor caracterelor care se afla pana la poziţia 15 inclusiv (aceasta este semnificaţie lui -15).

Pentru a realiza o imprimare a elementelor de pe prima coloana din fişierul text se poate folosi comanda :

```
$ cut -d ' ' -f1 Cumparaturi.txt
2kg
1
1kg
1
1
```

Comanda de mai sus realizează o extragere de coloane, mai exact al primei coloane – aceasta este semnificaţia lui f1. Dacă de exemplu se va cere extragerea coloanelor 2 şi 4 se folosi -f2-4(testaţi). Separarea intre coloane se realizează prin spatii – aceasta este semnificaţia lui -d ' '. Dacă de exemplu ca şi caracter de delimitare intre cuvinte se foloseşte semnul : (doua puncte) atunci se va specifica -d ':'.

Exercițiul 2 :

Se considera fişierul :

```
Ion       Ionescu   5 6 8 8 9 10
Andrei  Popescu   7 5 4 9 9 9
Adelina Verde      8 8 7 9 9 10
```

Folosind comanda cut sa se afişeze :

a) numele de familie;
b) prenumele;
c) nota obţinuta la a treia disciplina de studiu;
d) iniţiala prenumelui;
e) sa se creeze un fişier numit Studenti.txt, care sa conţină numele si prenumele studenţilor din fişierul de mai sus.

3. Expresii regulate

Expresiile regulate sunt nişte şiruri de caractere ce reprezinta şabloane sau tipare (*pattern* in limba engleza). Ele se construiesc pe baza unei gramatici, i mod similar unui limbaj de programare. Aceste şabloane sunt folosite pentru "recunoaşterea" si manipularea unor şiruri de caractere.

Analog cu expresiile aritmetice, o expresie regulata este construita prin combinarea unor expresii mai mici cu ajutorul unor operatori.

Sa începem cu un exemplu simplu:

```
a.z
```

Aceasta expresie regulata se potriveste cu orice sir de caractere ce contine literele 'a' si 'z' intre care se gaseste orice caracter -- dar unul singur (cu exceptia caracterului newline, deobicei), cum ar fi: "axz", "aaz", "barza", dar nu "abcz".

Cele mai simple expresii regulate sunt cele care "se potrivesc" cu un singur caracter: majoritatea caracterelor (toate literele si cifrele) se potrivesc cu ele insele. Alte caractere insa au semnificatie speciala , si daca dorim ca expresia regulata sa se potriveasca cu acel caracter, trebuie sa il citam (*quote* in limba engleza). Aceasta se poate realiza prin plasarea unui *backslash* ('\') in fata caracterului respectiv. Expresiile regulate mai complexe se vor forma fie prin concatenare (scriere una dupa alta) fie cu ajutorul operatorilor ce vor fi descrisi mai jos.

Prin concatenarea a doua expresii regulate rezulta o expresie regulata ce se va potrivi cu şiruri de caractere ce conţin doua subşiruri adiacente ce se vor potrivi cu prima respectiv a doua expresie regulata.

O alta construcţie care potriveste un singur caracter este o lista de caractere inchise intre paranteze drepte. Aceasta expresie se va potrivi cu oricare din caracterele din lista. Astfel, expresia regulata

```
compl[ei]ment
```

se va potrivi cu oricare din şirurile "complement", "compliment" sau "mulţumesc pentru complimentul dumneavoastră". Dacă o construcţie cu paranteze drepte începe cu un ^ , atunci ea se va potrivi cu orice caracter ce **nu** este intre paranteze:

```
3[^6890]
```

reprezinta o expresie regulata ce se potriveşte cu orice sir ce conţine cifra 3 şi nu conţine pe poziţia următoare una din cifrele 6, 8, 9 sau 0 (atenţie: dacă în şirul căruia i se aplica expresia regulata nu conţine după 3 nici un alt caracter, expresia nu se va potrivi!). De asemenea se pot specifica intervale întregi (considerând ordinea ASCII a caracterelor) cu ajutorul cratimei:

```
[A-Za-z]
```

reprezinta orice litera, mica sau mare.

Caracterele care nu se potrivesc cu ele înşele şi de care aminteam mai sus sunt metacaractere si operatori. O parte dintre ele, printre cele mai des utilizate si implementate in diversele variante de expresii regulate le vom descrie mai jos, alături de alte construcţii.

Punctul (**.**) se potriveşte cu orice caracter, unul singur (mai puţin caracterul *newline* , de obicei).

Expresiile regulate formate cu caracterele ^ şi **$** sunt puţin mai speciale, ele nu se potrivesc cu un sir de caractere ci cu şirul vid de la începutul si respectiv sfârşitul unui rind (sir). De exemplu

```
^turing$
```

se va potrivi doar cu şirul "turing" (nu si cu "featuring" sau "turing ").

Sunt definiţi o serie de operatori pentru a specifica repetiţiile. Ei se aplica în dreapta unei expresii regulate , făcând-o sa se potrivească repetitiv:

Operator	**Modificare adusa**
*****	Potriveşte de 0 sau mai multe ori
+	Potriveşte de 1 sau mai multe ori
?	Potriveşte de 0 sau 1 ori

Parantezele rotunde se folosesc pentru a grupa expresiile regulate. Astfel, cada scriem

```
ab*
```

aceasta semnifica un 'a' urmat de oricâte 'b'-uri (inclusiv nici unul) ; dar dacă scriem

```
(ab)*
```

operatorul * se aplica grupului, semnificaţia fiind 0 sau mai multe repetiţii ale şirului de caractere "ab".

Un alt operator util este |, operatorul de alternare. Rezultatul lui este ca se va potrivi fie expresia regulata din stânga, fie cea din dreapta:

```
ion (pozitiv)|(negativ)
```

se va potrivi fie cu "ion pozitiv" fie cu "ion negativ".

Pentru mai multe detalii, consultaţi paginile de manual **grep(1)**. Expresiile regulate pot fi folosite si din limbaje de programare precum C, vedeti **regex(3)** si **regex(7)**.

4. Comanda shell UNIX `wc`

Sa se recupereze de pe moodle în maşina gazda arhiva `Lab4.tar.gz`. Copiaţi acest fişier din maşina gazda in dosarul LAB4 de pe maşina virtuala.

Sa se decompreseze si sa se dezarhiveze arhiva.

```
tar –xvf Lab4.tar.gz
```

Sa se creeze subdirectorul OScrisoarePierduta în directorul LAB4. Sa se deplaseze fişierul text “OscrisoarePierduta.txt” din directorul LAB4 in subdirectorul creat anterior. Pornind de la conţinutul acestui fişier text şi utilizând tehnica expresiilor regulate şi comenzile UNIX învăţate pana acum sa se realizeze următoarele task-uri:

Exerciţiul 3 :

a) Citiţi pagina de manual a comenzii wc.
b) Sa se numere cate linii are textul de mai sus, utilizând comanda wc.
c) Sa se numere cate cuvinte are textul de mai sus, utilizând comanda wc.
d) Sa se numere cate caractere are textul de mai sus, utilizând comanda wc.
e) Sa se numere liniile care încep cu litera minuscula.
f) Sa se numere apariţiile cuvântului urale.
g) Sa se afişeze numărul de replici ale lui Pristanda. Care este persoana care are cele mai multe replici in aceasta piesa de teatru ?
h) Cate scene are aceasta piesa de teatru ?
i) Cate apariţii are cuvântul curat ? Dar expresia "curat murdar" ? Cărui personaj aparţine ticul verbal "curat" ?
j) Afişaţi liniile care conţin cuvântul "curat" .
k) Afişaţi replicile cetăţeanului turmentat.

5. Filtrare. Comanda shell UNIX `find`

Comanda shell find este utilizata pentru regăsirea de fişiere într-o arborescenta de fişiere şi executarea de diferite acţiuni asupra fişierelor identificate în urma procesului de căutare.

Sintaxa generala a acestei comenzi este :

```
find   cale(cai)   criteriu(ii)  acţiune(i)
```

unde :

- cale, este numele directorului de unde se va începe procesul de căutare. Este obligatorie prezenta a cel puţin un nume valid de director, iar acesta trebuie sa se afle pe prima poziţie.

- criteriul se refera la modul de căutare. Pot exista mai multe criterii de căutare în ierarhia de fişiere după cum urmează :

- name regexf : numele fişierelor respecta o anumita expresie regulata.
Observaţie : expresia regulata trebuie separata de caracterul ' la ambele capete.
-path regexf : numele cailor ce desemnează locurile unde se cauta fişierele.
-amin n : fişierul a fost accesat acum n minute.
-mmin n : fişierul a fost modificat acum n minute.
-cmin n : statutul fişierului a fost modificat acum n minute.
-empty : fişierul este gol.
-size n : mărimea fişierului în blocuri este de n (-nc pentru biti, -nk pentru kilobiţi).
-perm n : permisiunea asupra fişierului exprimata prin cifra octala este n.
-type c : tipul fişierului este c(*d* pentru directoare, *l* pentru linkuri, *f* fişiere obişnuite)
-maxdepth n : adâncimea maxima la care se coboară în sistemul de fişiere.
-mindepth n : adâncimea minima la care se coboară în sistemul de fişiere.

- acţiunile care se pot realiza asupra fişierelor regăsite sunt :

-delete : ştergerea fişierelor care corespund criteriilor.
-exec : executa o anumita comanda asupra acelui fişier.
-ok : executa o anumita comanda asupra unui anumit fişier dar numai cu acordul utilizatorului.
-print : afişaj simplu
-fprint file : analog cu file, doar ca direcţionează scrierea în fişierul specificat.
-ls : afişează diverse informaţii relativ la fişierele găsite.

Pentru argumentele numerice se poate folosi :
n : pentru a desemna exact numărul n.
+n : pentru a desemna numere mai mari sau egale cu n.
-n : pentru a desemna numere mai mici sau egale cu n.

De asemeni testele supuse spre verificare de către criterii pot fi grupate în structuri complexe după cum urmează :

- conjuncţie : test1 -a test2 (test1 si test2)
- disjuncţie : test1 -o test2 (test1 sau test2)
- negaţie : ! test1 (test invers lui test1)
- expresii : \(expresie\) (ansamblu de teste)

Exerciţiul 4

Creaţi un subdirector "Filtrare" în directorul LAB4. Copiaţi arhiva Cod_Sursa.tar.gz şi dezarhivaţi-o în acest folder. (Dezarhivarea unei arhive tar.gz se realizează cu ajutorul comenzii tar – vedeţi exemplu de utilizare mai sus în textul lucrării).

a) Găsiţi toate fişierele cu extensia .java din directorul Biblioteca.
b) Găsiţi toate subdirectoarele din directorul Biblioteca.
c) Găsiţi toate fişierele care pot fi accesate în scriere şi citire de către proprietar, şi doar în citire de către alţi utilizatori şi utilizatorii din grup.
d) Ştergeţi toate fişierele cu extensia mdb.
e) Găsiţi toate directoarele care încep cu litera I.
f) Găsiţi toate clasele care încep cu litera F şi se termina cu litera r si afişaţi date relative la ele.
g) Găsiţi toate fişierele care nu sunt vide şi au permisiunea rw-r--r-- .
h) Găsiţi toate directoarele care sunt goale sau au mărimea mai mica de 100 kiloocteţi
i) Găsiţi toate fişierele care au fost modificate în ultimele 5 minute.
j) Găsiţi toate fişierele mai mari de 1 kilooctet şi mai mici de 3 kiloocteti.

6. Căutarea într-un ansamblu de fişiere

Creaţi un subdirector "CautareInFişiere" in directorul LAB4. Copiaţi arhiva Cod_Sursa.tar.gz si dezarhivaţi-o în acest folder.

Exerciţiul 5

Scopul acestui exerciţiu este de familiarizare cu operaţiile care se pot realiza asupra unui ansamblu de fişiere : identificare anumitor tipuri de fişiere, identificare anumitor informaţii într-un ansamblu de fişiere.

a) Sa se numere fişierele java care se afla în directorul Biblioteca/src .
b) Sa se numere toate liniile de cod scrise în cadrul acestui proiect.
c) Sa se numere toate liniile de comentarii scrise in cadrul acestui proiect.
d) Sa se calculeze numărul de clase din acest proiect.
e) În care fişier este importata clasa StringTokenizer?
f) Afişaţi toate metodele care returnează date de tip int, dar nu date de tipul int[] .
g) Afişaţi lista tuturor variabilelor de tip vector definite în acest proiect.
h) Care sunt metodele publice de clasa pentru clasei Data. java ? În ce clase sunt ele de asemeni utilizate ? Cu ce argumente ?

Conţinutul raportului

- Faceţi un raport prezentând mersul exerciţiilor şi prezentaţi-l pentru validare la sfârşitul lucrării de laborator sau în decurs de maximum doua săptămâni după efectuarea lucrării. Exerciţiile pot de asemenea fi validate dacă reuşiţi să le prezentaţi profesorului pe parcursul lucrării de laborator.
- Răspundeţi cu grijă la întrebările puse în fiecare punct ale exerciţiilor şi indicaţi comanda sau şirul de comenzi utilizate precum şi opţiunile şi parametrii respectivi.

Referinţe

- Paginile de manual regex(3) si regex(7)
- http://www.regular-expressions.info/
- Diverse surse pe Internet