Cuprins

In loc de concluzie: UNIX, un sistem de operare inca neprietenos, in care poti face aproape orice iti doresti!

UNIX - cel mai puternic si mai folosit sistem de operare

Sistemul de operare, oricare ar fi el, are ca sarcina principala gestiunea resurselor fizice, logice si informationale ale sistemului de calcul, in vederea folosirii cit mai eficiente a acestor resurse. Pentru a putea fi cit mai eficiente, sistemele de operare sint, in general, puternic dependente de arhitectura si structura calculatorului. Din acest punct de vedere, sistemul UNIX este o exceptie cu totul remarcabila. Scris in proportie de aproximativ 90% in limbajul de nivel inalt C, el realizeaza deziderate importante cum ar fi standardizarea, portabilitatea si generalitatea algoritmilor, alaturi de facilitatile multitasking, multiuser. Considerat ca o revolutie in lumea sistemelor de operare, UNIX s-a bucurat de un succes deosebit in lumea specialistilor, el devenind curind si un succes de piata, datorita in principal portabilitatii sale, atit la nivelul interfetei, cit si al aplicatiilor, care i-a permis sa se raspindeasca pe cele mai variate arhitecturi hardware. Alte facilitati oferite de sistem sint din sfera prelucrarii performante a textelor. Un mecanism interesant pus la indemina utilizatorilor este mecanismul “pipe”, care permite interconectarea diverselor procese, in vederea realizarii unor prelucrari complexe. In ultimii ani UNIX-ul s-a dezvoltat continuu, atit in ceea ce priveste nucleul, caruia i-au fost adaugate concepte evoluate cum ar fi prelucrarea in retea, lucrul in timp real, precum si programarea concurenta, cit si in ceea ce priveste aplicatiile, fiind implementate compilatoare performante pentru un numar mare de limbaje, editoare si procesoare de text evoluate, interfete grafice, sisteme de gestiune de baze de date performante, sisteme expert, etc. O calitate importanta a sistemului este disponibilitatea sa pentru interoperabilitate cu retele de calculatoare (de ex., Novell) - acest lucru este de dorit pentru ca permite atit exploatarea aplicatiilor existente sub UNIX, cit si a celor existente sub sistemul de operare de retea. Deasemenea, UNIX-ul ofera facilitati de lucru in retele de larga raspindire geografica (de ex., Internet).

Oportunitatea realizarii sistemului de operare UNIX este ilustrata de urmatoarele considerente:

- necesitatea standardizarii si unificarii mediilor de operare
- transportabilitatea fisierelor intre diverse sisteme de calcul, mentinind identica structura volumelor si fisierelor
- asigurarea unor niveluri superioare de portabilitate a produselor program (se extinde portabilitatea de la nivelul limbajelor de programare catre portabilitatea la nivelul executivului - asistenta fiind identica din partea sistemului de operare - si la nivelul filozofiei de implementare a programelor)
- folosirea unor game largi de arhitecturi de calcul, prin interconectarea calculatoarelor de tipuri si puteri diferite, functionind sub acelasi sistem de operare
- transparenta produselor software fata de evolutia hardware-ului

Un amanunt interesant legat de istoria aparitiei sistemului este ca o copie in format sursa a primei versiuni a sistemului a fost pusa la dispozitia utilizatorilor pentru numai 150 de dolari! In afara de asta, laboratoarele Bell (AT&T), “parintii” UNIX-ului, au distribuit sistemul catre universitati, astfel incit o intreaga generatie de absolventi in domeniul stiintei calculatoarelor a patruns in domeniul industrial ca adepta a UNIX-ului. Acest lucru a dus la diversificarea sistemului intr-o masura neatinsa de nici-un alt sistem de operare; bineinteles ca, in timp, s-au impus citeva versiuni majore (derivate din AT&T UNIX System V si 4.3BSD, care sint adevarate standarde de facto; o directie interesanta este SOLARIS-ul firmai SUN, care incearca imposibilul : un UNIX universal, compatibil BSD si Sytem V).

In prezent, UNIX este, de fapt, numele unei familii de sisteme de operare, care se dezvolta si se raspindeste continuu, fiind foarte potrivita caractreizarea de “unitate in

diversitate". S-a reusit deasemenea realizarea unui standard, acesta fiind versiunea ("release") UNIX System V Release 4.

Deci sistemul de operare UNIX este sistemul suport pentru o multime de aplicatii ingineresti si economice din domenii cum ar fi invatamintul superior, cercetarea, financiar-bancar, militar, proiectarea, asigurari, guvernamental, multimedia, etc. si constituie cea mai importanta modalitate de realizare a sistemelor deschise.

Sistemul de operare UNIX e alcatuit din patru clase de componente software:
- nucleu (kernel) : care realizeaza functiile de baza la sistemului de operare
- interfata (asigurata de limbajul de comanda Shell) : care asigura comunicarea utilizator - sistem
- programe utilitare : care ofera posibilitati suplimentare de exploatare a sistemului de calcul
- programe de aplicatie : care incearca sa rezolve probleme specifice utilizatorilor

I. Sesiune de lucru UNIX. Comenzi Shell :

1. Deschiderea unei sesiuni de lucru : se face la intrarea in sistemul UNIX sau dupa lansarea unui emulator de terminal pentru un sistem non UNIX.
Dupa ce ne-am hotarit pe ce poarta sa intram in lumea UNIX, daca am ajuns in fata ei, pe ecran ar trebui sa vedem:

Enter your login name: <--aici se va tasta numele de utilizator, urmat de apasarea tastei ENTER
Enter your password: <--aici se va tasta parola, urmata de apasarea tastei ENTER

(Un utilizator e caracterizat de :
- uid (identificatorul utilizatorului)
- gid (identificatorul grupului din care face parte uid)
- cont
- parola
- home directory
- Shell implicit

)

Daca totul e OK, atunci pe ecran vom vedea unele din informatiile urmatoare :
Welcome to UNIX SYSTEM... (asta e mesajul zilei)
You have mail (asta e mesaj de posta)
news : readme (asta e mesaj de noutati)
$ (asta e prompter-ul sistemului de operare)

OBSERVATIE Oricare din informatiile anterioare lui $ poate lipsi, $ insa nu (cind sistemul de operare e gata sa ne ajute el ne anunta acest lucru afisind un prompter de sistem)
Orice comanda se termina cu ENTER sau RETURN

IMPORTANT! ORICE NOTIUNE NEEXPLICATA IN MOMENTUL INTRODUCERII EI VA FI RELUATA SI EXPLICATA IN CONTEXTUL CORESPUNZATOR.

2. Ce e nou in sistem ?

Daca doriti sa aflati raspunsul la aceasta intrebare aveti mai multe posibilitati de actiune :

$news (citire a tuturor mesajelor)
$news nume_mesaj (citire a unui anumit mesaj)
$news -a (recitire a tuturor mesajelor)

3. Linie de comanda:

Format general:
$comanda +/-optiune(optiuni) argument(argumente)
↑ ↑
blank sau TAB blank sau TAB

O comanda este un program care spune sistemului sa faca ceva (ex. news)
Un argument spune comenzii respective cum sa-si execute actiunea (ex. nume_mesaj)
O optiune modifica comanda (ex. -a)

IMPORTANT! Pentru a afla tot ce va doriti despre o comanda se foloseste comanda "man", care permite citirea manualelor UNIX, existente in sistem. Comanda este similara comenzii "help" din alte sisteme de operare. Sintaxa este : $man nume_comanda

Aplicatie:

CE ESTE GRESIT in aceste linii de comanda ?
$news-a

$newsunu doi
$news unudoi

4. Listarea fisierelor (fisierul e similar unei/unui carti/caiet dintr-o biblioteca):

$ls (listeaza fisierele din directorul curent)

OBSERVATIE Cind intrati in sistemul UNIX va aflati in catalogul dumneavoastra initial (home directory). Fiecare utilizator are un catalog initial - Pentru a vedea care este acesta folositi comanda : $pwd (print work directory)

$ls -p
↑
se foloseste pentru a se lista in mod diferit directoarele(cataloagele) de fisiere
(un director e similar unui raft intr-o biblioteca)

5. Afisarea continutului fisierelor ASCII:

(fisiere ASCII sint acele fisiere care contin text)
(fisiere non ASCII sint :
- fisiere cu module obiect
- directoare
- fisiere de date folosite de unele programe)

$cat nume_fisier
Daca fisierul e "mai mare de un ecran" se va afisa in regim paginat folosind(altfel nu veti vedea decit finalul):
$pg nume_fisier

6. Data curenta : se poate afla/modifica cu comanda
$date

7. Comanda "banner" isi afiseaza in format marit argumentele (ex. $banner Hello world!)

8. Inchiderea sesiunii de lucru : se face
- tastind CTRL D
sau
- tastind exit <return>

OBSERVATIE Dupa deconectarea logica de la retea prin inchiderea sesiunii de lucru poate urma o deconectare fizica (administratorul de retea hotaraste in ce cazuri este necesara)

II. Posta electronica. Comunicatie interactiva

In UNIX exista urmatoarele posibilitati de comunicare intre utilizatori sau intre un utilizator si alt sistem :
- conversatia cu alt utilizator de pe sistemul UNIX
- transmiterea de posta la utilizatori de pe acelasi sistem sau de pe alt sistem UNIX
- transmiterea/receptia fisierelor de pe alte sisteme UNIX
- conectarea pe alt sistem (la distanta) prin sistemul local si posibilitatea de transfer de fisiere sau executie a unor comenzi de pe sistemul la distanta

9. Sa facem cunostinta cu comanda MAIL

Posta electronica (mail) este un mijloc prin care utilizatorii sistemului UNIX pot sa comunice intre ei.
Orice utilizator are o cutie postala (mailbox).
La fel ca si la posta obisnuita, mesajele (scrisorile) ramin in cutia postala pina cind e luata de destinatar

Un utilizator este anuntat ca are posta prin afisarea pe ecran a mesajului : "You have mail" (Sistemul UNIX nu intrerupe ceea ce lucrati in momentul primirii postei - abia cind terminati va afisa mesajul de anunt)

Pentru citirea postei folositi :
$mail
(Exemplu:

```
$mail
From Michael Fri May 3 11:27 EDT 1994    <- antet
Ne intilnim marti, cum am stabilit.      <- mesaj
Monica
?                                        <- prompter-ul comenzii mail
```

)

Se observa ca dupa ce tastati mail, urmat de ENTER, pe ecran se afiseaza primul mesaj din cutia respectiva (dupa un antet din care se poate afla expeditorul, data si ora transmiterii mesajului), dupa care se afiseaza prompter-ul "?" - la acest prompter se va indica comenzii mail ce sa faca cu mesajul respectiv :

---

s <RET> salveaza mesajul in fisierul mbox
w <RET> similar cu s <RET>, dar mesajul e salvat fara antet
s nume_fisier<RET> salveaza mesajul in fisierul precizat
d <RET> sterge mesajul din cutia postala
d numar <RET> sterge mesajul cu numarul precizat
<RET> trece la urmatorul mesaj, lasind mesajul curent in cutia postala
m <RET> utilizator expediaza mesajul curent unui anumit utilizator
q <RET> intrerupe citirea corespondentei(iesire din mail)
x <RET> iesire din mail, lasind nemodificat continutul cutiei postale
h <RET> afiseaza un scurt sumar al mesajelor
? <RET> la fel cu h <RET>
! comanda executa comanda UNIX
p <RET> determina refisarea mesajului curent
u [numar] <RET> reface mesajul sters cu numarul precizat (implicit mesajul curent)

OBSERVATIE Dupa ce indicati comenzii mail ce sa faca cu mesajul respectiv, ea va afisa mesajul urmator sau, daca nu mai sint mesaje, prompter-ul $

Pentru trimiterea unui mesaj prin posta folositi:

```
$mail destinatar1, destinatar2,.., destinatarn
Text mesaj(linia 1)<RET>
Text mesaj(linia 2)<RET>
        .
        .
        .
Text mesaj(linia n)<RET>
.<RET>
$
```

Sfirsitul mesajului se indica cu .<RET> sau cu CTRL D

Folosind comanda mail in acest mod puteti sa transmiteti chiar fisiere editate cu un editor de texte (de exemplu vi) : $mail destinatar <nume_fisier

OBSERVATIE Se pot transmite mesaje si la grupuri de utilizatori astfel: $mailx nume_grup

10. O comanda mai sofisticata decit mail : mailx

Comanda "mailx " este mai puternica si mai flexibila decit mail - folosirea ei este asemanatoare cu a lui mail :

$mailx destinatar1,destinatar2,...,destinatarn
Subject:mesaj ceva despre continutul mesajului
Acum urmeaza mesajul de transmis
. <RET>
$

Dupa introducerea subiectului si apasarea tastei <RET> sinteti in modul intrare al lui mailx - in acest mod puteti fie sa va tastati mesajul, fie sa executati comenzile mailx - iata citeva:
~p afiseaza mesajul pe ecran inainte de a fi trimis
~x se iese fara transmiterea mesajului
~c nume_noi permite adaugarea unei liste suplimentare de destinatii pentru mesajul respectiv
~r citeste continutul unui fisier in interiorul mesajului
~v permite editare mesaj cu vi

OBSERVATIE Comenzile trebuie tastate la inceputul liniei.

Pentru citirea postei electronice folosind mailx :
$mailx <RET>
...
?

La acest prompter se poate folosi una din optiunile:

---

n <RET> afiseaza mesajul n
d <RET> sterge mesajul curent din cutia postala
d [lista mesaje]<RET> sterge mesajele din lista din cutia postala
s <RET> salveaza mesajul curent in fisierul mbox
s [lista mesaje] nume_fisier<RET> salveaza mesajele din lista in fisierul respectiv
<RET> urmatorul mesaj este afisat
m utilizatori <RET> trimite un mesaj la alti utilizatori
q <RET> iesire pastrind mesajele necitite
h <RET> afiseaza sumarul mesajelor (la fel ca ?)
R <RET> trimite un raspuns la expeditorul mesajului curent
R [lista mesaje]<RET> trimite un raspuns la expeditorii mesajelor din lista
x <RET> se iese din mailx, dar in plus fata de "q <RET>", se anuleaza toate modificarile facute in cutia dumneavoastra postala (de ex., orice mesaj pe care l-ati sters va fi pus la loc in cutia postala). Deci comanda "x<RET>" paraseste mailx lasind cutia postala intacta

---

11. Trimiterea de mesaje la distanta :

Trimiterea de mesaje pentru un utilizator de pe un sistem aflat la distanta :

$mail destinatar\@nume_sistem_la_distanta
$mailx destinatar\@nume_sistem_la_distanta

OBSERVATIE In urma tastarii caracterelor \@ pe ecran va aparea caracterul @ (este un artificiu legat de caractere speciale in sistemul UNIX)

Aplicatie:

* Sa ne si amuzam putin :

Intr-un sistem UNIX se poate comunica direct cu un alt utilizator (ca si cum ati vorbi la telefon) folosind comanda "write" .
Exemplu:

$write stud1 <RET>
Vrei sa vorbim ?
-o- <RET>
Dupa ce ati initiat cu succes convorbirea, nu veti mai vedea prompter-ul interpretorului de comenzi si fiecare linie pe care o tastati va fi afisata pe ecranul celuilalt de indata ce apasati <RET>. Este bine sa comunicati celuilalt ca "vorbiti" în acel moment asfel incit cealalta persoana sa poata sa va raspunda fara sa "va intrerupa". O conventie, preluata de la "walkie-talkie", este sa tastati -o- la sfirsit (over). Daca partenerul, in acest caz stud1, doreste sa va raspunda, el poate apela write cu numele de cont al dumneavoastra (numele utilizatorului) si incepind din acest moment tot ce tasteaza oricare dintre dumneavoastra va fi afisat si pe ecranul celuilalt.Cind ati terminat convorbirea tastati -oo- ("over and out"). Pentru a iesi din write si a recapata prompter-ul interpretorului de comenzi, tastati un singur <CTRL D>. Un mesaj de sfirsit al transmisiei va fi afisat pe ecranul celuilalt utilizator, aratind ca ati iesit din write. Celalalt utilizator ar trebui sa iasa si el in acelasi mod.

Atentie!- daca mai multi utilizatori initiaza convorbiri (write) simultan cu acelasi terminal, ecranul acelui terminal poate deveni de neinteles.

.Pentru a permite utilizatorilor sa scrie la terminalul dumneavoastra, folositi "chmod" (cu numele fisierului asociat terminalului la care lucrati - pentru a afla acest nume folositi comanda "tty"). Exemplu : $chmod 666 /dev/term/40

.Pentru a nu permite scrierea pe terminalul dumneavoastra folositi comanda "mesg n", care inhiba dreptul de scriere (implicit este YES)

Aplicatie (folosirea comenzii “write”):

STUD1		STUD2
$ write stud2 ❶	→	message from stud1❷
Vrei sa vorbim ?		tty10
-o- <RET>		Vrei sa vorbim ?
		-o-
		↓
message from stud2❹←		write stud1 ❸
Vreau		Vreau <RET>
-o-		-o- <RET>
↓❺		
Imi pare bine <RET>→		Imi pare bine❻
.....		....
-oo-		-oo-
CTRL D	→	EOT ❼
$	←	CTRL D
EOT❽		

Asta a fost !

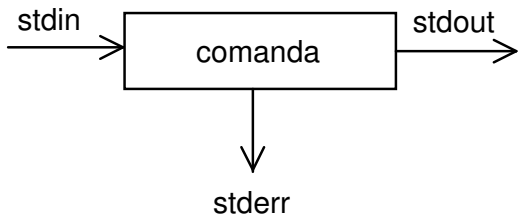
Aplicatie:

Intr-un sistem UNIX se mai poate comunica direct cu un alt utilizator (ca si cum ati vorbi la un "videotelefon", fara imaginea interlocutorilor) folosind comanda "talk" - pentru a afla cum lucreaza comanda “talk” folositi comanda de citire a manualelor UNIX, “man” ($man talk)

III. Comenzi Shell (continuare)

12. Redirectari fisiere standard

Cele mai multe comenzi UNIX isi preiau datele de la "intrarea" standard (stdin), care, in mod normal, este tastatura terminalului la care se desfasoara sesiunea de lucru. Dupa ce interpretorul de comenzi citeste si prelucreaza sirul de intrare, se executa comanda corespunzatoare (daca sirul de intrare este corect). Frecvent comanda produce o "iesire" (stdout) de exemplu "date" produce o linie de iesire continind informatii despre data si ora) - aceasta iesire este scrisa la "iesirea" standard, care, in mod normal, este ecranul terminalului. Daca apar probleme legate de executia comenzii, sistemul genereaza un mesaj de eroare, numit "iesire diagnostic " - acesta este afisat la asa numitul fisier standard de erori (stderr), care, de obicei, este tot ecranul terminalului.



OBSERVATIE Cele trei fisiere standard sint codificate cu numerele, respectiv, 0, 1, 2 - 0 pentru intrare, 1 pentru iesire si 2 pentru eroare.

a) Redirectare fisier standard de iesire :

Sa dam mai intii un exemplu : $banner welcome >mesaj <RET>

Executia acestei comenzi va avea ca rezultat depunerea iesirii comenzii "banner" in fisierul "mesaj" - acest fisier va fi creat, daca nu exista deja sau va fi rescris (continutul original al fisierului va fi inlocuit cu iesirea comenzii "banner")

Deci, implicit iesirea standard este ecranul terminalului - ea poate fi schimbata cu ajutorul a doua simboluri :
> nume_fisier :redirecteaza iesirea standard catre un fisier
>> nume_fisier : concateneaza iesirea standard la un fisier

Exemplu pentru concatenare : $date >>mesaj

Executia acestei comenzi va concatena iesirea standard a comenzii "date" la sfirsitul fisierului "mesaj" - daca fisierul "mesaj" nu exista, el va fi creat

b) Redirectare fisier standard de intrare :

Sa incepem tot cu un exemplu : $mail user1 user2 < mesaj

Executia acestei comenzi are ca efect transmiterea ca mesaj prin posta electronica a continutului fisierului "mesaj" catre cele doua conturi utilizator "user1", "user2"

Intrarea standard poate fi schimbata cu ajutorul simbolului "<", urmat de numele fisierului de intrare.

OBSERVATIE Simbolurile de redirectare a intrarii, respectiv iesirii, pot fi folosite simultan in aceeasi linie de comanda.

c) Redirectare fisier standard de eroare

Exemplu : $cat fi 2>err

Efectul executiei comenzii "cat" este afisarea continutului fisierului "fi", daca acesta exista - daca nu exista se va genera un mesaj de eroare, care, in mod normal, va fi afisat pe ecran - daca s-a folosit indirectarea"2>", mesajul de eroare va f scris in fisieul "err"(daca vom incerca sa vizualizam continutul lui "err", vom vedea ca acesta este : " cat : cannot open fi ")

OBSERVATII
. Redirectarea permite memorarea iesirii sau folosirea informatiilor memorate ca intrare
. Simbolul de redirectare apare intotdeauna intre comanda si fisier

13. Mecanismul "pipe"

Uneori, pentru obtinerea unui anumit rezultat, este nevoie de mai mult de o comanda - bineinteles ca ele pot fi date pe rind, insa sistemul ofera posibilitatea "concatenarii" a doua sau mai multe comenzi prin mecanismul "pipe" - de exemplu, este nevoie de comenzile "banner" si "mail" pentru a trimite prin "mail" o comanda prelucrata in prealabil cu "banner" :

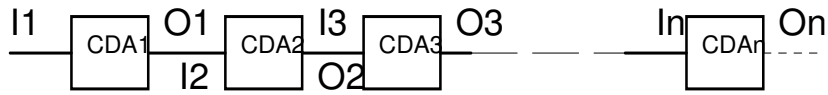
```
$banner Donald | mail user1
                 ↑
           simbol "pipe"
```

Pentru rezolvarea acestei probleme se mai putea folosi redirectarea intrarii/iesirii standard, astfel :

```
$banner Donald > mesaj
$mail user1 < mesaj
```

Avantajul primei solutii este cistigul de timp (nu se mai produc intirzieri datorita lucrului cu fisiere intermediare)
Avantajul celei de a doua solutii este ca informatiile din fisierul intermediar ("mesaj") se pot folosi ulterior

Cea mai mare parte a comenzilor UNIX sint proiectate astfel incit sa poata fi executate "pipeline" - un "pipeline" este o serie de comenzi conectate cap-la-cap, astfel :



unde

- Ii este intrarea pentru blocul "i"
- Oi este iesirea pentru blocul "i"
- CDAi este a "i" a comanda

14. Alte comenzi

a) Comanda "who" afiseaza lista utilizatorilor conectati in sistem, cite o linie cu informatii pentru fiecare utilizator. Exemplu :

```
$who
user1  term/04 Mar   11  14:20
user2  term/51 Mar   11   09:50
$
```

b) Comanda " wc" cu optiunea -l numara liniile de la intrare.
Daca se realizeaza un pipe intre cele doua comenzi, pe ecran se va afisa numarul utilizatorilor aflati in acest moment in sistem. Exemplu :

```
$who |wc -l
       2
```

$

OBSERVATIE Acest numar poate fi memorat (in vederea unei utilizari ultrerioare) intr-un fisier folosind o indirectare, astfel : $who | wc -l > users

c) Comanda "cal" afiseaza un calendar pentru un anumit an (exemplu cal 1890); ca si alte comenzi intilnite produce mai mult de un ecran la iesire - afisarea poate fi controlata folosind combinatia de taste corespunzatoare, astfel :
CTRL S - opreste temporar afisarea
CTRL Q - reporneste afisarea

d) Uneori dorim oprirea unei comenzi inainte de terminarea executiei ei (de exemplu pentru "who") - aceasta se poatee face apasin una din tastele (aceea de care dispuneti la terminalul dumneavoastra) :

<DEL>
<RUBOUT>
<BREAK>
<INTERRUPT>

Dupa apasarea uneia dintre aceste taste, de obicei, reapare prompter-ul $.

15. Parole

Cind intrati in sistemul UNIX trebuie sa aveti un cont si o parola valide; oricine poate sa va cunoasca numele de cont, dar numai dumneavoastra trebuie sa va cunoasteti parola. Unele sisteme UNIX sint configurate astfel incit sa ceara schimbarea periodica a parolei. O parola corecta trebuie sa

- contina cel putin 6 caractere
- foloseasca cel putin 2 litere (mari sau mici)
- foloseasca cel putin o cifra sau un semn de punctuatie

OBSERVATIE Numai primele 8 caractere sint recunoscute

OBSERVATIE Aceste reguli sint valabile pentru UNIX System V Release 4 - pe un sistem mai vechi regulile pot sa fie putin diferite.

Schimbarea parolei se face folosind comanda “passwd”, astfel :

$ passwd
Old password : ----
New password :----
Re-enter new password :----
$

Aceasta e secventa actiunilor care au loc daca procesul de schimbare a parolei decurge fara erori - pot apare erori daca :

- introduceti gresit vechea parola
  Mesajul de eroare : “Sorry”
- parola nu e construita corect
  Un mesaj de eroare :
  “Passwords must differ by at least 3 positions”
  Alte mesaje pot indica anumite caracteristici ale parolei, dependente de sistem
- parola nu e retastata corect :
  Mesajul de eroare :
  “They don’t match; try again”

IV. Sistemul de fisiere UNIX System V

Un fisier este un loc cu nume unde se stocheaza informatie.

16. Tipuri de fisiere

a) obisnuite (ordinare)
In ele se memoreaza texte, date, programe
OBSERVATIE Si fisierele executabile sint fisiere obisnuite (de exemplu un program compilat sau un program in limbaj de comanda; cele mai multe din comenzile UNIX sint programe compilate

b) cataloage (directoare)
Ele sint folosite pentru a grupa fisiere, de obicei, "inrudite". Fiecare catalog contine pentru fiecare fisier component numele sau, impreuna cu un index (i_number) intr-o tabela cu informatii despre fisiere - fiecare intrare in tabela (i-nod) contine informatii despre fisier, cum ar fi proprietarul, tipul, dimensiunea, drepturile de acces.

c) speciale
Orice obiect din sistemul UNIX este un fisier.
Echipamentele periferice (terminale, unitati de disc, imprimante, unitati de banda magnetica) sint tratate ca fisiere speciale (avantajul acestei posibilitati este ca se pot asocia drepturi de acces pentru ele). Comunicatia cu dispozitivul se face prin intermediul fisierului special. Administratorul de sistem creeaza un fisier special de fiecare data cind se adauga o noua componenta hardware in sistem.

d) legaturi simbolice
Un fisier de legare simbolica este un fisier care reprezinta simbolic alt fisier. Legaturile simbolice pot traversa sisteme de fisiere diferite.

17. Reguli de construire a numelor de fisiere

- se recomanda sa descrie continutul
- trebuie sa aiba maxim 14 caractere in USV (altfel se face o trunchiere)
- se recomanda sa fie constituite cu ajutorul literelor mari/mici, cifrelor, punctului (.) precum si a liniutei de unire _ .

OBSERVATIE Sistemul de operare UNIX face distinctie intre literele mari si mici.

- nu se recomanda folosirea caracterelor speciale (cu semnificatie speciala pentru limbajul de comanda) : + - \ * / [] () ; & ? $ ^ | < > ‘ “ {}
- blank-urile nu sint acceptate

OBSERVATII
- Daca numele fisierului incepe cu ".", acesta nu va apare cind se incearca listarea cu "ls", decit daca se foloseste optiunea "-a"; numele fisierelor administrative incepe cu "."
- Pentru a afla dimensiunea maxima a unui fisier folositi comanda "ulimit" - exemplu :

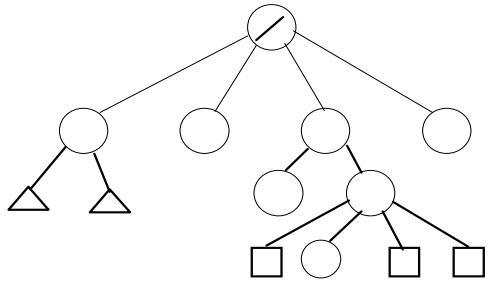
```
$ulimit
    2048
$
```

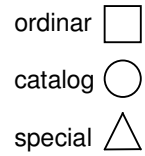
*unde, 2048 este numarul de blocuri fizice (dimensiunea unui bloc depinde de masina - de exemplu, daca dimensiunea blocului este de 512 octeti -> 2048*512= 1Moctet)*

18. Sistem de fisiere

Un sistem de fisiere este o organizare a fisierelor.

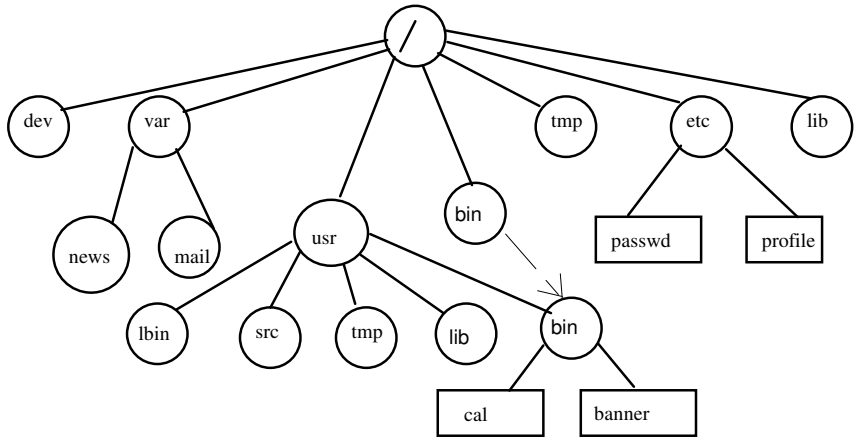
Sistemul de fisiere UNIX este o structura ierarhica, arborescenta, de fisiere - el se constituie intr-un arbore cu radacina in sus, in care directoarele sint noduri interne, iar fisierele sint frunze :





Exista mai multe standarde de sisteme de fisiere : vxfs, ufs, s5, s.a. Uzual se foloseste unul dintre ele

19. Cataloage standard



Un sistem de fisere contine, in mod obisnuit urmatoarele cataloage :

- */dev* (“device”) : contine fisierele speciale pentru dispozitivele periferice

- */bin* (“binar”): inainte de USV acest catalog continea comenzile UNIX standard, ca “ date” si “who” - acum “bin” este legat simbolic la catalogul /usr/ bin care contine comenzile UNIX standard (a fost numit “bin” pentru ca cele mai multe comenzi erau programe binare)

- */var* (“variabil”): contine urmatoarele cataloage :
  news - contine cite un fisier pentru fiecare noutate din sistem
  mail - orice utilizator care are posta necitita va avea un fisier in acest catalog cu acelasi nume ca al contului utilizatorului respectiv

- */usr* (“users”): include urmatoarele cataloage:
  lbin: poate contine programe scrise de utilizatori
  src: contine fisierele cu codul sursa al sistemului UNIX, daca sistemul dumneavoastra contine codul sursa
  tmp: contine fisiere temporare create de utilizator
  lib: contine biblioteci
  bin: contine comenzi sistem ca “banner” si “cal”

- */tmp* (“temporar”): contine fisiere temporare, precum si spatiu de lucru pentru comenzile UNIX

- */etc*: contine programe administrative si informatii despre utilizatori

- */lib*: contine biblioteci folosite pentru programare

20. Comenzi de lucru cu fisiere

Catalogul initial este locul in cadrul sistemului de fisiere unde va aflati in momentul cind incepeti sesiunea de lucru

Catalogul curent este locul in cadrul sistemului de fisiere unde va aflati la un moment dat (comanda "pwd" afiseaza numele complet al directorului curent)

OBSERVATII

- catalogul initial este decis de administratorul de sistem; de obicei, cataloagele initiale ale utilizatorilor care fac aceeasi munca sint grupate in acelasi catalog parinte

- catalogul dumneavoastra initial este propriul dumneavoastra subsistem (subarbore) : aici puteti crea/sterge directoare, crea/sterge fisiere, etc. (in functie de drepturile de acces la fisiere ei au sau nu dreptul sa faca acelasi lucru in alte cataloage)

a) Listarea fisierelor se face cu comanda "ls"

Exemplu :

```
$ls -l
total 12   ← numarul total de blocuri de disc ocupate de fisierele listate

legaturile la fisiere   numarul de octeti din fisier   numele fisierului
                  ↓                    ↓                    ↓
drwx------ 2 nume_cont nume_grup 128 Jun 9 12:58 bin
↑   ⇑                                          ⇑
tipul fisierului   masca drepturilor de acces          data si ora ultimei modificari
```

d : director
- : fisier ordinar
l : legatura simbolica

OBSERVATII

- Prin "legaturi la fisiere" se intelege numarul de directoare in care apare numele fisierului respectiv (pentru cataloage, numarul de astfel de legaturi este de cel putin doua, pentru ca fiecare director e continut in el insusi si in parintele sau, iar pentru fisiere ordinare este unu, pina cind se foloseste comanda "ln")

- Nume_cont, respectiv nume grup sint identificatori de cont, respectiv de grup pentru proprietarul fisierului

b) Un nume de cale ("path name") este drumul pe care sistemul trebuie sa-l urmeze in arbore pentru a localiza un fisier. El este constituit dintr-o lista de cataloage separate prin "/", lista care se termina cu numele fisierului tinta. Doua notiuni importante sint : nume complet de cale si nume relativ de cale -

Nume complet de cale (cale absoluta)	Nume partial de cale (cale relativa)
Cautarea incepe dinradacina	Cautarea incepe din catalogul curent
Intotdeauna incepe cu /	Nu incepe niciodata cu /
Nu depinde de catalogul curent	Depinde de catalogul curent

OBSERVATIE Se foloseste numele de cale care este de lungime mai mica.

c) Notatii speciale pentru

catalogul curent : .
catalogul parinte : ..

OBSERVATIE Aceste notatii pot fi folosite ca atare intr-un nume de cale - de exemplu in comanda : $ls .. , care va avea ca efect listarea fisierelor din catalogul parinte al catalogului curent

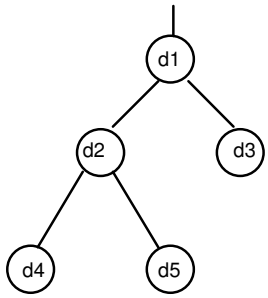
d) Schimbarea catalogului se face cu comanda "cd"

Exemple :

$cd /dir1/dir2

Comanda "cd" fara parametri ne duce de oriunde ne-am afla in directorul nostru initial $ cd

Aplicatie (folosirea comenzii "cd" cu cale absoluta, respectiv relativa):



Presupunem ca sintem in directorul radacina, d1 :

-Comanda $cd /d2/d4 are ca efect schimbarea directorului din radacina (/) in directorul d4 (cale absoluta)
-Acum sintem in directorul d4 - pentru ajunge in d5, avem doua posibilitati:

$cd /d2/d5 (cale absoluta)
$cd ../d5 (cale relativa)

e) Determinarea tipului fisierului

Atunci cind dorim sa vizualizam continutul unui fisier putem folosi comanda "cat", insa, daca fisierul nu este ASCII, comanda "cat" avind ca argument fisierul respectiv, va poate bloca terminalul - pentru a evita acest lucru se poate preceda comanda "cat" cu comanda "file" pentru a afla tipul fisierului (ea verifica primele 512 caractere ale fisierelor primite ca argumente si face o scurta descriere a fiecaruia dintre ele)

Exemplu :

```
$file test text d
test :    ascii text
text :    English text
 d :       directory
 $
```

OBSERVATII
- Daca tipul fisierului include cuvintul text, atunci acesta poate fi afisat cu comanda "cat"

- Comanda "file" desemneaza ca English text orice fisier ASCII care contine semne de punctuatie
- Alte tipuri de fisiere identificate de aceasta comanda sint :
programe sursa C, Fortran
fisiere de comenzi
fisiere executabile, s.a.

f) Cautarea fisierelor se face cu comanda "find", care realizeaza o cautare recursiva, in tot subarborele care are ca radacina catalogul de start

Sintaxa generala a comenzii este :

$find catalog_start expresie_logica actiune, unde

- catalog_start este catalogul din care incepe cautarea (numele de cale catre el)

- expresie_logica este o expresie care descrie conditiile in care se face cautarea (conditiile pot fi "inlantuite" folosind operatori de negare (!), "sau" logic (-o), "si" logic (conditie1 conditie2))

Exemple :

$find . -type d -name biblioteca - print, care cauta directoarele al caror nume este "biblioteca", tiparind de fiecare data cind gaseste unul, numele lui

$find . ! \(-type d \) -print, care cauta toate fisierele care nu sint directoare; \, urmat de un caracter, inhiba orice alte semnificatii ale caracterului urmator

- actiune descrie actiunea ce va fi executata asupra fisierelor gasite de comanda "find"

Exemplu :

$find . -name carte -print

Aceasta comanda gaseste, pornind cautarea din directorul curent, toate fisierele cu numele "carte" si tipareste numele lor de cale - deci catalogul de start este directorul curent (.), expresia este "-name carte", iar actiunea este "-print". Fisierele pot fi cautate si dupa alte criterii decit numele si pot fi tratate si in alte moduri, in afara de "print"

Exemple de optiuni :
-type x : cautare dupa tipul, x, al fisierului
x va fi
d pentru director
b pentru fisiere orientate pe blocuri
c pentru fisiere orientate pe caractere
p pentru fisiere "pipe"
f pentru fisiere obisnuite
-links n : cautare dupa numarul de legaturi, n, al fisierului
-user nume_utilizator : cautare a fisierului, daca apartine utilizatorului specificat

Exemplu de argument :
-exec comanda

Folosirea acestui argument duce la executia comenzii specificate, avind ca argument fisierul gasit de "find"

OBSERVATII:
- Nu se recomanda folosirea frecventa a comenzii "find", deoarece aceasta solicita destul de mult UCP
- "Find" va afisa un mesaj de eroare pentru fiecare catalog din subsistem in care nu aveti drept de cautare
- Comanda cauta in tot sistemul de fisiere, indiferent cite volume exista

g) Listarea recursiva a fisierelor se face folosind ca optiune a comenzii "ls", "-R" - exemplu : $ ls -pR, care va lista fisierele din directorul curent, precum si din intregul subarbore care are ca radacina directorul curent

IMPORTANT! Chiar daca un director este subdirector al lui "home directory" asta nu inseamna ca aveti toate drepturile in el !

h) Tiparirea continutului fisierelor ASCII se poate face cu una din comenzile folosite in exemplele urmatoare:

$cat nume_fisier
$pg nume_fisier afiseaza in regim paginat
$pr -d nume_fisier afiseaza "la doua rinduri"
$pr -n nume_fisier afiseaza fisierul, numerotind rindurile

OBSERVATII
- aceste comenzi afiseaza continutul fisierului pe ecranul terminalului
- "pr" permite si formatarea textului

l) Servicii de imprimare

Tiparirea unui fisier se poate face :

- referind imprimanta cu numele fisierului special corespunzator :
$cat carte >/dev/lp

Varianta aceasta este rapida, insa poate conduce la afisari incorecte datorita posibilitatii aparitiei unor cereri de imprimare concurente; se recomanda folosirea ei cind imprimanta este alocata exclusiv.

- referind un serviciu de tiparire :
Aceasta varianta presupune instalarea la nivelul sistemului a unor servicii de tiparire, locale sau de retea - fiecare serviciu se identifica prin nume

Comanda "lp" se foloseste pentru a obtine o copie a unui fisier la imprimanta - exemplu :
$lp carte
Efectul executiei acestei comenzi va fi tiparirea la imprimanta a continutului fisierului "carte"; fisierul nu va fi formatat in nici-un fel, el aparind ca si cum s-ar fi folosit "cat".

OBSERVATIE
-Anumite fisiere nu pot fi tiparite in acest fel - e vorba de fisierele la care drepturile de acces sint restrinse, adica exista putine posibilitati de prelucrare a fisierului respectiv - de aceea, de obicei, fisierele sint formatate cu comanda "pr" si iesirea acestuia este legata prin "pipe" la comanda "lp" - exemplu : $pr carte | lp
- Daca sistemul pe care lucrati are mai mult de o imprimanta, una dintre ele va fi cea implicita - pentru a folosi o alta imprimanta decit cea implicita se poate proceda in doua feluri
. se poate folosi in comanda "lp" optiunea "-d", urmata de numele imprimantei; exemplu:
$pr carte | lp -d lp1
. se poate modifica fisierul ".profile" astfel ca, de fiecare data cind intrati in sistem, variabila "LPDEST" sa fie setata la numele imprimantei dorite, aceasta devenind noua imprimanta implicita- fiecare utilizator dobindeste astfel un serviciu implicit:
LPDEST=lp1
export LPDEST

- Exportarea unei variabile o plaseaza in mediul dumneavoastra de lucru, ea fiind accesibila in continuare comenzilor ca "lp".

Sistemul de operare mentine o coada cu cererile de imprimare existente -la fiecare cerere de imprimare sistemul va raspunde cu un numar de lucrare, care este identificatorul intern al cererii - exemple :

```
$pr carte | lp
request id is lp-4327 (standard input)

$ls -R | lp -d lp1
request id is lp1-4473 (standard input)
```

Starea cererilor de imprimare se poate afla folosind comanda "lpstat" - exemplu :

```
$lpstat
lp1-4473 jmg 743 nov 30 12:45 on lp1
```

O cerere de imprimare poate fi abandonata folosind comanda "cancel", care va folosi ca argument numarul de lucrare intors de comanda "lp" - exemplu:

```
$cancel lp1-4473
request "lp1-4473" cancelled
```

Daca nu se precizeaza numarul cererii se abandoneaza cererea curenta de imprimare.

Fisierul de tiparit primeste in fata o pagina antet, numita pagina "banner" din care aflam de unde provine; tiparirea acestei pagini se poate inhiba cu -onobanner - exemplu :

```
$lp -onobanner carte |pg
request id is dl1200  -81 (1 file)
$
```

m) Si in UNIX se poate opera asupra unui grup de fisiere, numele grupului putind fi reprezentat de un sablon care contine caractere obisnuite si caractere de generare a numelor de fisiere.

OBSERVATIE O alta varianta, mai greoaie, de operare asupra unui grup de fisiere este sa insiruim numele fisierelor respective in linia de comanda respectiva, separate de blank

Atunci cind vrem sa operam direct asupra unui grup de fisiere se presupune ca ele sint inrudite ca inteles, deci si ca nume - asta inseamna ca numele lor vor avea o parte comuna si una care difera de la un fisier la altul - aceasta parte diferita poate fi reprezentata cu ajutorul caracterelor de generare, ele tinind locul caracterelor care lipsesc.

Exemple de caractere de generare:

1) * (asterisc) poate inlocui 0 sau mai multe caractere consecutive dintr-un nume de fisier (exceptie : punctul la inceput de fisier)

Exemplu :

- $ls fisier*
  fisier11 fisier22 fisier1 fisier2 fisier3 fisier

OBSERVATIE
- Un singur asterisc tine loc pentru toate numele de fisier din directorul respectiv
- Daca sablonul se potriveste cu un nume de director, atunci "ls" va lista continutul acestuia in intregime

2) ? (semnul intrebarii) poate inlocui un singur caracter dintr-un nume de fisier; se pot folosi mai multe caractere "?", fiecare reprezentind un caracter oarecare (exceptie : punct la inceputul unui nume de fisier) .

Exemplu :

```
$ls fisier??
fisier11 fisier22
```

3) [] (multime de caractere) inlocuiesc numai un caracter dintre cele aflate intre paranteze (aceeasi exceptie ca la 1,2).

Exemplu :

```
$ls fisier [1 2 3]
fisier1 fisier2 fisier3
```

In acest exemplu multimea elementelor este data prin enumerarea elementelor ei - ea poate fi data si ca domeniu, prin specificarea limitei inferioare, respectiv a limitei superioare, separate de cratima.

Exemplu :

```
$ls fisier[1-3]
```

OBSERVATII
- Exista posibilitatea negarii unui set de caractere, prin adaugarea simbolului "!" ca prim carater al multimii - exemplu:

```
$ls fisier [!1]
fisier2 fisier3
```

- Caracterele de generare pot apare oriunde intr-un nume de cale
- Se pot folosi combinatii de caractere de generare pentru a genera nume de fisiere ca argumente in comenzi

n) Manipularea fisierelor si a cataloagelor

OBSERVATIE Se recomanda gruparea fisierelor intre care exista relatii in acelasi director

- creare director :

Pentru a crea un director se foloseste comanda "mkdir", astfel :

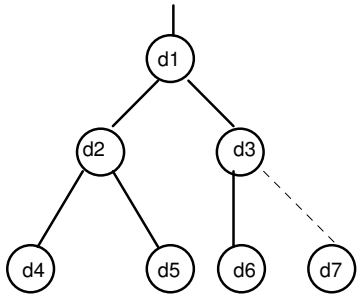
```
$mkdir nume_de_cale_director
```

Evident, cel mai simplu este sa ne mutam in directorul care va avea ca fiu noul director si sa folosim comanda astfel : $mkdir director

OBSERVATIE
-"nume_de_cale_director" poate fi dat in forma cale absoluta sau relativa
- absenta oricarui mesaj de eroare indica incheierea cu succes a executiei comenzii

Aplicatie :

Daca ne aflam in directorul d3, pentru a crea un fiu al acestuia, sa spunem d6, putem folosi comanda "mkdir" astfel : $mkdir d7
Ca urmare, directorul d7 va aparea ca fiu al lui d3



Comanda mai putea fi folosita si cu precizarea caii absolute, indiferent unde ne-am fi aflat in arborele de informatie : $mkdir /d3/d7

Daca ne aflam in d6 se putea preciza calea relativa pentru d7 : $mkdir ../d7

- stergere fisiere ordinare :

Fisierele de care nu mai avem nevoie pot fi sterse cu ajutorul comenzii “rm” :

$rm nume_de_cale_fisier

Si in acest caz calea poate fi precizata absolut sau relativ. Absenta unui mesaj de eroare indica succes in actiunea comenzii.

OBSERVATIE
- In numele de cale pot aparea caractere de generare
- Comanda “rm” folosita ca mai sus nu intreaba utilizatorul daca este sigur in ceea ce priveste stergerea fisierului sau fisierelor respective - daca utilizatorul are drepturile de acces necesare asupra fisierului respectiv, acesta va fi sters. Pentru un mod de lucru interactiv se foloseste optiunea “-i” - exemplu :
$rm -i fisier?
fisier1: ?n - tastind n fisierul nu va fi sters
fisier2: ?y - tastind y fiserul va fi sters
fisier3: ?y
$

- stergere directoare :

Stergerea cataloagelor se face folosind tot comanda “rm” cu optiunea “-r” - folosirea acestei optiuni permite stergerea recursiva a catalogului si a continutului sau, fara a mai pune vreo intrebare; absenta unui mesaj de eroare indica succes.
Exemplu (presupunem ca ne aflam in d3):

$ rm -r d7

Daca directorul este vid atunci el poate fi sters si cu comanda “rmdir” ($rmdir d7).

OBSERVATIE
- Pentru a sterge interactiv continutul unui director se pot folosi ambele optiuni astfel :

$rm -ri nume_de_cale_director
- Daca ati sters din greseala un director, singura posibilitate de a-l reface este de pe un “backup” anterior facut de administratorul de sistem, daca directorul exista in momentul realizarii backup-ului.

- copiere fisiere ordinare :

Copierea fisierelor se face cu ajutorul comenzii “cp”, astfel :

$cp nume_cale_fisier_sursa nume_cale_fisier_dest

NOTA

In continuare se va folosi o "prescurtare" : "sursa" entru numele de cale al sursei, respectiv "destinatie" pentru numele de cale al destinatiei.

OBSERVATII referitoare la "cp"
- Daca "destinatie" este un nume de fisier, atunci se va face o copie a fisierului initial cu acest nou nume, in directorul curent. Daca exista deja un fisier cu acest nume nou, acesta va fi rescris fara nici-un avertisment - pentru a evita aceasta situatie se foloseste opriunea "-i", care permite folosirea interactiva a comenzi
- Daca "destinatie" este un nume de director, atunci fisierul (fisierele) sursa vor fi copiate, cu numele lor initial, in acest director; daca se doreste copierea unor fisiere in directorul curent, acesta poate fi desemnat cu punct
- Utilizatorul care executa "cp" devine proprietarul fisierului copiat

- mutare/redenumire fisiere ordinare :

Aceste operatii pot fi efectuate cu comanda "mv" :

$ mv nume_cale_fisier_sursa nume_cale_fisier_dest

OBSERVATIE
- Diferenta intre copiere si mutare este ca in urma mutarii fisierul sursa se va gasi numai la destinatie
- Observatiile valabile pentru "cp" sint valabile si pentru "mv".
- Daca si "sursa" si "destinatie" sint nume de fisiere ordinare, atunci se va produce o redenumire a fisierului "sursa"

- copiere cataloage :

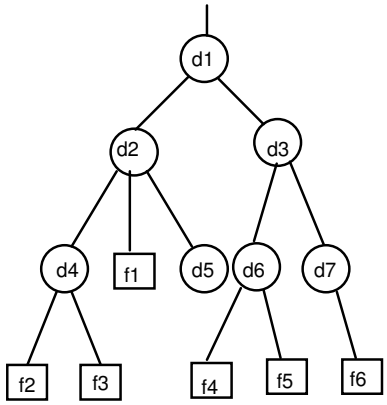
Pentru a copia un director, impreuna cu toate fisierele sale se foloseste comanda "cp", cu optiunea "-r" - daca directorul destinatie(aici, obligatoriu destinatia este un director) nu exista, el va fi creat mai intii cu ajutorul comenzii "mkdir".
Sintaxa este analoaga celei folosite pentru fisiere ordinare.

- mutarea/redenumirea directoarelor :

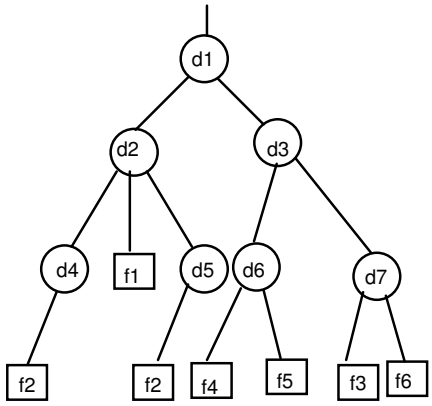
Folosind comanda "mv" un catalog pot fi numai redenumit - pentru a-l muta in alt catalog din arbore se pot parcurge urmatoarele etape :

- se face copierea "sursei" la "destinatie"
- se sterge "sursa"

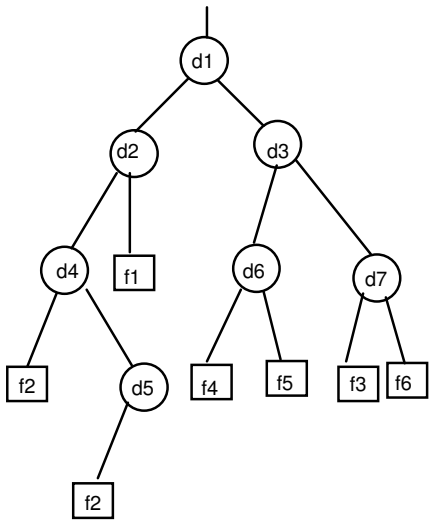
Aplicatie :



Sa presupunem ca sintem in radacina :
$cd /d2/d4 → ne va deplasa in d4
$cp f2 ../d5 → va copia fisierul f2 in directorul d5
($cp f2 f3 ../d5 → ar fi copiat ambele fisiere in d5)
$mv f3 /d2/d7 → muta fisierul f3 in directorul d7



Sintem in continuare in d4
$cp -r ../d5 . → va copia d5, recursiv, in d4



$mv d6 d8 → redenumeste d6 cu numele d8

o) Drepturi de acces

Drepturile de acces se pot atasa oricaror obiecte din sistem - reamintim ca orice obiect UNIX este fisier.
Se poate efectua o prelucrare asupra unui obiect UNIX, numai daca drepturile de acces ale procesului care efectueaza aceasta prelucrare sint in concordanta cu natura prelucrarii (de exemplu, nu se poate scrie intr-un fisier in care avem numai drept de citire)
La creare, un fisier va fi asociat cu doua informatii de identificare a proprietarului sau (initial este cel ce l-a creat) : uid-ul, respectiv gid-ul acestuia, pe care le vom desemna in continuare cu uidf, respectiv gidf.
In continuare exista 3 categorii de utilizatori care pot cere accesul la fisier :

- proprietarul sau
- grupul care poseda fisierul

(comanda “id” permite aflarea uid-ului, respectiv gid-ului utilizatorului curent)

- alti utilizatori

In momentul in care un utilizator doreste sa execute o anumita prelucrare asupra unui fisier, uid-ul utilizatorului, impreuna cu gid-ul sau, sint transmise procesului care urmeaza sa execute prelucrarea - acesta va cere acces la fisierul respectiv prin executia unui algoritm de acces simplu :

```
daca uid=uidf atunci

        procesul este proprietar  al fisierului

altfel

        daca gid=gidf atunci

          procesul e membru al grupului proprietar al fisierului

        altfel
                procesul intra in categoria altii
```

Fiecare clasa de utilizatori are un anumit set de drepturi de acces la un fisier - comanda “ls” cu optiunea “-l” afiseaza drepturile de acces la fisiere. Cele mai folosite drepturi de acces sint "read", "write", "execute".

Drept de acces	Semnificatii pentru : Fisier ordinar	Catalog
Read	permite citirea din fisier (de ex. cu “cat”, “pr”, “cp”)	permite listarea fisierelor din catalog
Write	permite modificarea continutului fisierului (de ex. prin editare)	permite adaugarea, stergerea, rede-numirea fisierelor, daca este prezent si dreptul de “execute”
Execute	permite executarea fisierului ca o comanda, daca fisierul este un program; este necesar si dreptul “read”	permite accesul la fisierele incluse in catalog, schimbarea catalogului cu “cd”

OBSERVATIE
- Se recomanda inhibarea dreptului de scriere in cataloagele proprii pentru utilizatorii din clasele “group” si “others”, deoarece un fisier poate fi sters, chiar daca dreptul de scriere nu e trecut printre drepturile asociate lui, insa catalogul parinte are asociate drepturile “write” si “execute”

Sa ne reamintim ca la folosirea comenzii “ls” cu optinea “-l”, se afiseaza o masca a drepturilor de acces de forma : XXXXXXXXXX, care poate fi vazuta ca fiind “partitionata” in 4 parti, astfel :

```
X       XXX     XXX     XXX
↑        ↑       ↑       ↑
tip     user    group   others
fisier
```

p) Schimbarea drepturilor de acces

IMPORTANT! Schimbarea drepturilor de acces poate fi facuta numai de catre proprietarul fisierului sau de catre super-utilizator (super-user)

Schimbarea drepturilor de acces poate fi facuta in doua moduri :

-absolut (numeric)
-simbolic

In ambele cazuri se foloseste comanda “chmod”.

Modul absolut :

In aceasta varianta de abordare a problemei, fiecarui drept de acces i se asociaza o valoare, astfel :

- read (r) : 4
- write (w) : 2
- execute (x): 1

si, deasemenea, fiecarei clase de utilizatori i se asociaza o cifra octala, obtinuta prin insumarea drepturilor de acces corespunzatoare clasei respective, astfel :

Exemplu :

```
d   rwx     r-x   r-x
    4+2+1   4+1 4+1, adica 755
```

OBSERVATIE Liniuta de unire arata ca dreptul respectiv este inhibat (insiruirea drepturilor se face in ordinea : read write execute)

Pentru schimbarea drepturilor de acces se foloseste comanda “chmod” astfel :

$chmod mod_nou nume_fisier(e)

unde mod_nou reprezinta noile drepturi de acces la fisierul nume_fisier, descrise in forma numerica (o insiruire de 3 cifre octale). Nume_fisier(e) poate fi dat si ca sablon.

Exemplu :

```
$ ls -l test
-rwxr-xr-x 1 test txt 165 Nov 5 11:11 test
$chmod 744 test
$ls -l test
-rwxr--r-- 1 test txt 165 Nov 5 11:11 test
$chmod 400 test
$ls -l test
$
```

Modul simbolic :

Comanda “chmod” se foloseste cu urmatoarea sintaxa :

$chmod [cine] operator [dr_acces] nume_fisier(e)
(informatiile aflate intre paranteze drepte sint optionale), unde

- “cine” este alcatuit din 0 sau mai multe caractere u, g, o, a, specificind clasa de utilizatori pentru care se asigneaza/schimba drepturile de acces; daca “cine” lipseste este considerat implicit “a”, luind insa in considerare si setarile din masca de creare a fisierului.

- “operator” poate fi +, -, sau = si are ca semnificatie felul in care permisiunile se schimba :

+ se adauga permisiuni
- se elimina permisiuni
= se asigneaza permisiuni in mod absolut
(spre desosebire de ceilalti operatori, “=“ are un efect absolut resetind toti bitii - omiterea permisiunilor se foloseste numai impreuna cu acest operator)

- “dr_acces” este orice combinatie valida alcatuita din urmatoarele litere :
r drept de citire
w drept de scriere
x drept de executie
l blocare acces pentru citire/scriere in timp ce un program acceseaza acest fisier
u, g, o indica faptul ca permisiunea este luata de la utilizatorul curent, grup sau altii

OBSERVATIE
- Comanda “chmod” se poate folosi pentru directoare cu optiunea “-R”, care permite coborirea recursiva in director, setind modul pentru fiecare fisier intilnit

Exemple :

$chmod a-x test → interzice tuturor dreptul executie
$chmod go+rw test → asigneaza drepturile “read” si “write” pentru “group” si “others”
$chmod 066 test → acelasi efect ca mai sus

Exista doua moduri de acces implicite, unul pentru cataloage si unul pentru fisiere, care sint atasate acestora la creare :

- cataloage : acestea, implicit, pot fi citite, scrise, sau consultate de oricine, deci in absenta oricarei restrictii, un catalog nou creat va avea modul 777
- fisiere ordinare : acestea, in mod normal, nu sint executabile, astfel modul lor implicit este 666

Restrictii in privinta modului de acces initial pot fi introduse de catre administratorul de sistem, prin definirea unei “masti” care inhiba anumite drepturi de acces - aceasta masca va fi introdusa in fisierul /etc/profile, astfel ca la crearea unui nou fisier, acestuia i se va atribui automat “masca” respectiva. Pentru a afla care este aceasta se foloseste comanda “umask” :

```
$umask
0022
$
```

Prima cifra este 0 si semnifica faptul ca este un numar octal, iar urmatoarele 3 corespund celor 3 clase de utilizatori - fiecare cifra indica drepturile care sint inhibate clasa de utilizatori respectiva (de exemplu, masca 022 va inhiba dreptul de scriere pentru “group” si “others”, deci modul de acces al directoarelor nou create va fi 755, iar al fisierelor ordinare 644); aceasta masca afecteaza numai drepturile de acces ale fisierelor nou create, pentru cele deja existente, ea raminind neschimbata.
Pentru a modifica masca existenta se foloseste aceeasi comanda avind ca argument noua masca - de exemplu, daca vrem ca noua masca sa interzica scrierea doar pentru altii, vom folosi “umask”, astfel : $umask 002. Pentru a stabili o masca pentru toate sesiunile de lucru ulterioare se include aceasta masca in fisierul “.profile”

r) Grupuri de utilizatori

Necesitatea grupurilor se impune atunci cind mai multi utilizatori, lucrind la un proiect comun, au nevoie sa partajeze fisiere sau atunci cind se permite numai folosirea anumitor comenzi

pentru anumiti utilizatori. Comanda"groups" afiseaza toate grupurile la care apartineti - exemplu :

```
$groups
users other
$
```

La fiecare intrare in sistem sinteti membru intr-un grup implicit, pe care-l puteti afla folosind comanda "id" - exemplu :

```
$id
uid=103 (guser1) gid=100 (users)
$
```

Daca sinteti membru al mai multor grupuri, pentru a crea fisiere care sa apartina unui anumit grup, trebuie sa "comutati" pe grupul respectiv folosind comanda "newgrp" - de exemplu : $newgrp alt, va schimba grupul curent cu "alt", presupunind ca acest grup exista si ca sinteti membru in el; absenta oricarui mesaj de eroare indica executia cu succes a comenzii. O noua folosire a comenzii "id" va semnala comutarea pe acest grup.

V. Editorul ecran al sistemului UNIX (vi)

OBSERVATIE "locala"
Daca lucrurile nu se desfasoara asa cum afirm eu
in cele spuse in continuare, s-ar putea sa fie vina terminalului pe care lucrati - folositi urmatoarea linie de comanda :

```
$TERM=vt100;export TERM
```

21. Generalitati

a) Un editor este un pachet de programe folosit pentru a crea si modifica fisiere (text); aceasta operatie se numeste editare.
Un editor poate fi orientat linie sau ecran, facilitatile de deplasare a cursorului fiind corespunzatoare; un editor orientat linie e folositor cind iesirea este pe hirtie si nu pe un display.
Editorul "vi" este editorul ecran standard al sistemului UNIX. El poate fi folosit pe o gama larga de terminale video - editorul trebuie sa cunoasca tipul terminalului (variabila TERM din mediul utilizatorului contine tipul terminalului - pentru a vizualiza mediul utilizatorului se pot folosi comenzile "set" sau "env"; pentru a schimba continutul ei se foloseste o atribuire de forma "$TERM=noua_val", urmata de un apel al comenzii "$export TERM" pentru a face vizibil continutul valorii acestei variabile comenzilor lansate de utilizator).

Modul de lucru al editorului "vi", ca si al altor editoare, se bazeaza pe existenta unui buffer intermediar intre utilizator si fisierul de pe discul fizic - in acest tampon se afla copia curenta a fisierului, curenta in sensul ca ea este aceea care va fi afectata de comenzile editorului; continutul acestui buffer poate fi salvat (scris) pe discul fizic, folosind o comanda corespunzatoare. Din acest motiv exista o limita maxima a dimensiunii unui fisier; pentru a putea manipula fisiere mai mari se foloseste comanda "split".

b) Apelul editorului se face astfel :

$vi [nume_fisier]

OBSERVATIE Numele fisierului de editat este optional in aceasta linie de comanda, deoarece el poate fi precizat si mai tirziu, pe parcusul sesiunii de editare, in momentul in care se salveaza continutul fisierului.

In urma tastarii acestei linii de comanda pe ecran se afiseaza o fereastra de editare in care, daca fisierul exista deja vom vedea primele sale linii, iar daca fisierul nu exista pe ecran va apare "prima sa pagina goala" in care putem incepe sa introducem informatie. Prin aceasta fereastra de editare vedem pozitia curenta in cadrul tamponului.

Initial, cursorul va fi pozitionat in coltul stinga sus al ferestrei. Liniile aflate dupa sfirsitul fisierului, care nu contin nici-un fel de informatie si deci sint diferite de liniile cu blank-uri, sint indicate cu ajutorul caracterului tilda ("~") aflat in prima pozitie din fiecare astfel de linie. Pe

ultima linie a ecranului apar informatii legate de numele fisierului, numarul sau de linii, precum si numarul de caratere.

c) Moduri de lucru

Editorul ecran are 2 moduri de lucru :

- mod comanda
- mod introducere text

Comutarea din mod introducere text in mod comanda se face tastind <ESC>.

Modul comanda este folosit pentru introducerea de comenzi; la inceputul sesiunii de lucru va aflati in mod comanda. Comenzile care sint folosite pe parcursul unei sesiuni de editare sint de doua feluri:

- comenzi fara ecou si care nu se termina cu <RET> (de ex. a,i,o); ele pot fi folosite imediat ce am tastat <ESC>
- comenzi cu ecou si care se termina cu <RET> (asemanatoare comenzilor date in sistem) (de ex. w,wq); ele se tasteaza la prompter-ul ":", care se obtine tastind ":" in mod comanda (din mod introducere text : se tasteaza <ESC> :)

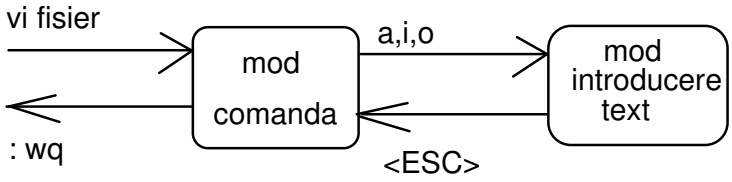
OBSERVATIE

- Daca introduceti incorect o comanda veti fi avertizati (sonor sau altfel, functie de tipul terminalului)

- Pentru a anula o comanda partial tastata apasati <ESC>

- Pentru afla permanent in ce mod de lucru sinteti tastati la prompter-ul ":" comanda "set showmode"

Modul introducere text este folosit pentru a introduce text in buffer; se folosesc comenzi de inserare cum ar fi a (append),i (insert),s.a. Initial va aflati in mod comanda, iar daca doriti sa introduceti text trebuie sa tastati "a".



22. Comenzi de editare :

OBSERVATIE

Se recomanda editarea textului in mod inserare text si modificarea lui, ulterior, in mod comanda.

a) Deplasare cursor

Atunci cind scriem ceva, fie in lumea reala, fie cu un editor, avem nevoie de un mecanism care sa permita punctarea pozitiei curente in care scriem la un moment dat - pentru lumea reala acesta este asigurat de virful obiectului cu care scriem, iar la editarea computerizata avem la dispozitie un cursor.
Cursorul este un caracter, de obicei "_", care indica pozitia unde se va face urmatoarea modificare in text. Prin caracter curent se intelege caracterul din pozitia cursorului.

DEPLASARE CURSOR	COMANDA
o pozitie la dreapta	→ l <SP>
o pozitie la stinga	← h <BS>
o pozitie in sus	↑ k

o pozitie in jos	↓ j
sfirsitul liniei curente	$
inceputul liniei curente	^
inceputul cuvintului urmator	w nw W nW (word)
inceputul cuvintului anterior	b nb B nB (back)
sfirsitul cuvintului curent sau urmator	e ne E nE (end)
inceputul liniei urmatoare	<ENTER>
inceputul liniei anterioare	-
ultima linie	G (go)
linia n	nG

OBSERVATII referitoare la tabel :

- <SP> este prescurtare pentru tasta <SPACE>
- <BS> este prescurtare pentru tasta <BACKSPACE>
- "n" este codificarea pentru "n" cuvinte

b) Inserare de text

Pentru a insera text se folosesc unele din comenzile fara ecou urmatoare :

COMANDA	EFECT
a (append)	adauga text dupa cursor
i (insert)	insereaza text inainte de cursor
o	insereaza o linie noua sub cursor
O	insereaza o linie noua deasupra cursorului

OBSERVATIE In mod introducere text se pot folosi pentru stergere :
- <BS> sterge un caracter (ultimul introdus)in modul introducere text
- @ sterge o linie in modul introducere text

Nu se recomanda folosirea acestora, deoarece este greoaie din cel putin doua motive:
- grupul de caractere sters din tampon nu dispare de pe ecran pina nu se revine in mod comanda (cu <ESC>)
- stergerile pot fi realizate numai daca nu se deplaseaza cursorul de pe linia respectiva

c) Stergere de text

COMANDA	EFECT
x	sterge caracterul din pozitia cursorului
nx	sterge "n" caractere
rc (replace)	inlocuieste caraterul curent cu "c"
dd (delete)	sterge linia curenta
ndd	sterge "n" linii
dw	sterge cuvintul urmator (de la stinga cursorului)
dnw	sterge "n" cuvinte la stinga cursorului
d$	sterge caracterele pina la sfirsitul liniei curente
d^	sterge caracterele pina la inceputul liniei curente
dL	sterge pina la ultima linie de pe ecran
dG	sterge pina la ultima linie din buffer
u (undo)	anuleaza efectul ultimei comenzi de modificare
U	reface linia curenta, numai daca nu ati mutat cursorul - indiferent de numarul de modificari
m,n dd	sterge liniile de la "m" la "n"

OBSERVATIE
- Daca legatura cu sistemul se intrerupe brusc, exista posibilitatea refacerii fisierului in curs de editare astfel:

$vi -r nume_fisier
(recovery)

Aceasta facilitate este posibila deoarece sistemul mentine un fisier temporar, copie al fisierului curent, in directorul "\tmp" - acesta este sters la o iesire normala din editor.
- Facilitatile de recuperare nu exista pe toate sistemele

d) Comenzi de manipulare buffer

COMANDA	EFECT
:wq<ENTER>	salveaza continutul buffer-ului si iese in sistem
:w<ENTER>	salveaza continutul buffer-ului si ramine in editare
:w nume_fisier<ENTER>	salveaza continutul buffer-ului in alt fisier
:q<ENTER>	iesire din editor fara salvare a buffer-ului, daca nu s-au facut modificari ale continutului buffer-ului
:q!<ENTER>	iesire din editor fara salvare a buffer-ului, daca au fost facute modificari ale continutului buffer-ului
<CTRL/d> (down)	defilare "in jos" a continutului buffer-ului
<CTRL/u> (up)	defilare "in sus" a continutului buffer-ului
<CTRL/f> (forward)	afisare a paginii urmatoare
<CTRL/b> (back)	afisare a paginii anterioare

:r nume_fisier<ENTER> citeste continutul fisierului "nume_fisier" dupa linia curenta
:nr nume_fisier<ENTER>citeste continutul fisierului "nume_fisier" dupa linia "n"

e) O sesiune de lucru cu editorul "vi" corespunde urmatorului "sablon" :

```
$vi text
Aceasta este o linie de text.
Aceasta este o linie de text.
Aceasta este o linie de text.
Aceasta este o linie de text.
~
~
~
"text" [New file]
:wq
$
```

23 . Comenzi de manipulare text

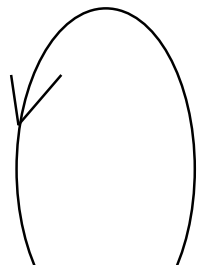
a) Cautare siruri

Facilitatea de cautare siruri permite regasirea rapida a sirurilor de caractere care corespund unui anumit sablon.

. cautare inainte, spre sfirsitul fisierului :

/sablon_sir<ENTER>

OBSERVATIE
- Cautarea este "circulara", in sensul ca ea incepe din dreapta pozitiei curente a cursorului, continua pina la sfirsitul buffer-ului, apoi revine la inceputul buffer-ului si continua pina la pozitia initiala a cursorului, astfel :

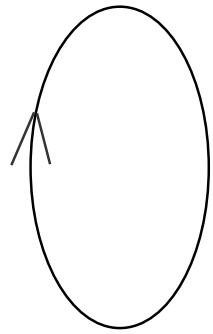


- Comanda are ecou in partea de jos a ecranului

. cautare inapoi, spre inceputul fisierului

?sablon_sir

OBSERVATIE
- Cautarea este "circulara", in sensul ca ea incepe din stinga pozitiei curente a cursorului, continua pina la inceputul buffer-ului, apoi revine la sfirsitul buffer-ului si continua pina la pozitia initiala a cursorului, astfel :



- Comanda are ecou in partea de jos a ecranului

. cautare repetata

Uneori primul sir gasit (care se potriveste cu sablonul) nu este cel dorit - cautarea poate continua :

- se poate cauta urmatoarea potrivire, in sensul initial, tastind "n" (next)
- se poate cauta urmatoarea potrivire, in sens invers, tastind "N"

b) Modificare text

COMANDA	EFECT
rc (replace)	inlocuieste caracterul curent cu caracterul "c"
xp (exchange position)	inverseaza caracterul curent cu urmatorul
stext<ESC> (substitute)	substituie caracterul curent cu sirul "text" (caracterul de inlocuit e marcat cu "$")
nstext<ESC>	substituie "n" caractere cu sirul "text"
cctext (change)<ESC>	modifica o linie, noul continut fiind "text"
ncctext<ESC>	modificare a "n" linii
cwtext<ESC>	modifica un cuvint
r<ENTER>	"sparge" o linie de text in doua, pe un blank
J	"alipire" a doua linii
nJ	"alipire" a "n" linii
cobiect text	modifica un obiect text Exemplu de obiect text: o pozitie in linia curenta

n,ms/sir/sir_nou/g	substituie un sir cu alt sir

OBSERVATIE Comanda "s", in aceasta sintaxa, permite substituirea tuturor ("g" - global) aparitiilor lui "sir" cu "sir_nou" , aflate pe liniile dintre "n" si "m" - exemple :

:s/abc/abcd/<ENTER> substituie prima aparitie a lui "abc" cu "abcd" in linia curenta

:s/abc/abcd/g<ENTER> substituie toate aparitiile lui "abc" cu "abcd" in linia curenta

:2,10s/abc/abcd/g<ENTER> substituie toate aparitiile lui "abc" cu "abcd" pe liniile de la 2 la 10

:3,3s/abc/abcd/<ENTER> substituie prima aparitie a lui "abc" cu "abcd" in linia 3

:1,$s/abc/abcd/g<ENTER> substituie toate aparitiile lui "abc" cu "abcd" in toate liniile din buffer (de la prima la ultima ($))

OBSERVATIE
Comenzile "s" si "c" comuta in modul introducere text - un <ESC> dupa folosirea comenzii ne aduce in mod comanda

c) copiere/mutare texte

Fiecare modificare sau stergere a unui text se face dupa ce acesta este salvat intr-un buffer temporar (bineinteles diferit de buffer-ul in care se afla fisierul de editat) - acest buffer contine intotdeauna ultima informatie stearsa sau modificata; aceasta facilitate este exploatata de unele comenzi UNIX, cum ar fi :

. Comanda "p" (sau "P" - put) permite aducerea in buffer-ul principal a unui text sters anterior - textul poate fi readus in orice pozitie dorim (comanda se poate folosi repetat, pentru acelasi text). Comanda "p" aduce textul din buffer-ul temporar in pozitia din dreapta cursorului; daca textul consta dintr-o linie sau mai multe, acesta va fi adus sub linia curenta.

.Comanda "P" aduce textul din buffer-ul temporar in pozitia din stinga cursorului, iar daca textul consta din una sau mai multe linii, el va fi plasat deasupra liniei curente.

Mutarea unui text dintr-un loc din buffer in alt loc se face conform secventei :

- se sterge textul cu ajutorul unei singure comenzi
- se muta cursorul in pozitia unde se doreste mutarea textului sters
- se foloseste comanda "put" ("p" sau "P") pentru a plasa o copie a buffer-ului temporar la locul dorit in buffer-ul principal

. Comanda "y" (yank) se foloseste pentru o copiere de text - ea salveaza textul din buffer-ul temporar, fara a-l sterge insa din buffer-ul fisierului; exemple:

- yw : salveaza o copie a cuvintului curent in buffer-ul temporar
- ynw : salveaza urmatoarele "n" cuvinte in buffer-ul temporar
- nyy : salveaza "n" linii in buffer-ul temporar
- yobiect_text : salveaza un obiect text

Exemplu :
yG salveaza din linia curenta pina la ultima
y$ salveaza pina la sfirsitul liniei curente
y^ salveaza pina la inceputul liniei curente

OBSERVATII
- Efectul comenzii "yank" nu este vizibil pe ecran

- Exista posibilitatea folosirii unor buffer-e cu nume pentru a salva mai multe segmente de text si a le transfera intre fisiere diferite, pe parcursul unei sesiuni de editare
- Exista posibilitatea folosirii unui set de 9 buffer-e cu numere care pot fi folosite pentru a restaura pina la 9 stergeri.

Copierea unui text dintr-o pozitie in buffer in alta pozitie se face conform secventei :

- se salveaza textul cu o singura comanda "'yank"
- se muta cursorul in pozitia din buffer unde se doreste realizarea copiei
- se foloseste comanda "put" pentru a plasa o copie a buffer-ului temporar in pozitia dorita din buffer-ul principal

OBSERVATII
- Exista posibilitatea folosirii unor buffer-e cu nume pentru a salva mai multe segmente de text si a le transfera intre fisiere diferite, pe parcursul unei sesiuni de editare
- Exista posibilitatea folosirii unui set de 9 buffer-e cu numere care pot fi folosite pentru a restaura pina la 9 stergeri

24. Alte comenzi

a) Executia unor comenzi Shell pe parcursul sesiunii de editare

:!comanda<ENTER> executa "comanda"

Exemplu:
:!ls va lista fisierele din directorul curent

Iesirea comenzii este urmata de un mesaj care ne indica ce trebuie sa facem pentru a reveni in editare (de obicei se tasteaza <ENTER>)

:!comanda %<ENTER> executa "comanda" avind ca argument fisierul curent

Exemplu:
:spell %<ENTER>

b) Stergerea si reafisarea ecranului

Daca pe ecran apar caractere nedorite (de exemplu un mesaj transmis de altcineva) se poate folosi comanda "CTRL l" care sterge ecranul si-l reafiseaza

Bibliografie

1.Tanenbaum A.,Operating Systems : Design and Implementation, Prentice Hall, N.J., 1987
2. Cristea V., Panoiu A., Kalisz E., Athanasiu I., UNIX, Editura Teora, Bucuresti, 1993
3. Pilat F., UNIX, Editura Teora, Bucuresti, 1992
4. Pilat F. V., Deaconu S., Popa S., Radu F., Introducere in Internet, Editura Teora, 1995