

## Curs 1

- prima gen de calc '40, fără SO
- a doua generație '50, primul SO
- a treia generație '60, multiprogramming
- UNIX: S. C. F. I. T-S. M-U. M-T.
- stdin 0, stdout 1, stderr 2

## Curs 2

- $c_1 \& c_2$  -  $c_2$  se exec. dacă  $c_1$  se term cu succes
- $c_1 || c_2$  -  $c_2$  se exec. dacă  $c_1$  se term cu insucces
- $c_1; c_2; c_3$  -  $c_1, c_2$  și  $c_3$  sunt ex. resequential
- $c_1 & c_2$  -  $c_1$  în background și  $c_2$  în foreground
- {list} - subshell
- {list} - shell curent
- ((aritmExpr)) - eval. aritmetică
- [[cond Expr]] - eval. expr. cond.

## Curs 3

- continue - cont cu urm iteratie for, while, until
- break - încheie o buclă: for, while, until
- expr expresie - va evalua și a expr și afisează
- expr string: expr - nr. caract din string iden  
tice cu cele din expr

## Curs 4

- sed [-n] [-e scenario] [-f fisier-scenario] [lista-fisi]  
↳ silent
- instrucțiuni:
  1. p - afisează bufferul temporar la stdout
  2. g/n1/n2 - inloc. y/abcd/1234

3. S/expr regulata / vir / [flaguri]

i) numic - prima aparitie

ii) intre 1-5 12 - se va anunta aparitie

iii) g - toate aparitiile

iv) p - afiseaza linoul tampon la ieire  
daca s-a produs urea modif in  
linia respectiva

Ex: echo Sunday | sed S/day | night | #Sunright

• grep [-e] expr-regulata |-f fis-scenariu | [lista-fis]

• awk [-f fisier-scenariu] [-F c] [scenariu] [-v var=val] [lis-fis]

! BEGIN este adversata inainte de prima linie

! END este adversata dupa ultima linie

• NF - nr. curante/campuri din linie curenta

• NR - nr. de ordine al liniei curente

• sort [-o ieire] [lista-fisiere]

↳ sort -c - verif daca e sortat

↳ sort -m - interclararea fisierelor de intrare

↳ sort -u - elim liniile duplicate din ieire

↳ sort -d - compararea doar litere, cifre si spatii

↳ sort -M - compararea numerelor de luni

↳ sort -n - compararea liniilor numeric

↳ sort -r - sortarea in ordine inversa

• uniq [fis-intrare [fis-ieire]]

↳ uniq -c - prefizarea liniilor cu nr de aparitie

↳ uniq -d - afiseaza doar liniile duplicate

↳ uniq -i - nu face dif dintre lit. mari si mici

↳ uniq -u - afiseaza doar liniile unice

- wc [ lista - fisiere ]
  - ↳ wc -c - nr de caractere / octeti
  - ↳ wc -l - nr de linii
  - ↳ wc -w - nr de cuvinte
- ps - afiseaza informatii despre procese

## Curs 5

- UNIX - primul SO care a utilizat structura arborescenta
  - extinde structura de graf aciclic
  - reporta simultan mai multe tipuri de fisiere ( chiar si a altor SO)
- Tipuri de fisiere
  1. , " legături hard
    - ↳ fisier pointer către alt fisier existent
  2. , l " legături simbolice
    - ↳ folosite pt. comunicatia prin retea intre procese, ruleate pe sisteme diferite
  3. , > " racheturi
    - ↳ folosite pt. comunicatia prin retea intre procese, ruleate pe sisteme diferite
  4. , / " pipe cu nume
    - ↳ datele sunt citite in ord. stricta scrierii lor, folosite pt. comunicarea pe acelasi int
  5. , c " periferice caracter
    - ↳ identifică un dispozitiv I/O de tip caracter
  6. , b " periferici bloc
    - ↳ identifică un dispozitiv I/O de tip bloc
  7. Pipe anumite
    - ↳ folosite pentru a transmite date intre com. sau programe

### 8. Segmente de memorie partajată

↳ permit mai multor programe să acceseze un spațiu comun de memorie

### 9. Cori de mesaje

↳ folosite de procese pentru a comunica între ele prin mesaje

### 10. Semafoare

↳ folosite pentru sincronizarea proceselor active concurente

## • Principale directoare UNIX

1. /bin/ - comenzi executabile utilizate frec.
2. /boot/ - conține nucleul sistemului și fișierele necesare procesului de încărcare
3. /dev/ - conține toate dispozitivele din sist.
4. /etc/
  - i) /passwd - lista de date a utilizatorilor
  - ii) /group - descrie grupei - crențător ↑
5. /home - conține dir. personale ale utilizatorilor
6. /lib - conține biblioteci de sistem și unele fișiere critice
7. /usr - conține cea mai mare parte a sistemului de sistem sau inst de utilizatori
8. /var - conține loguri, fișiere cu date variabile pe care sistemul le modifica în permanentă

## Curs 6

- int dup( int oldfd );
  - ↳ duplică un descriptor de fișier deja existent
- int dup2( int oldfd, int newfd );
  - ↳ oldfd și newfd referă același fișier fizic
  - ↳ modul de acces se păstrează la deschidere
  - ↳ ambele descriptori partajăruță același pointer current la fișier

## • Blocări de fișiere

1. blocare conciliantă (advisory)
2. blocare obligatorie (mandatory)
3. blocare exclusivă
  - ↳ un singur proces are acces la fișier sau la partimia de fișier
4. blocare partajată
  - ↳ partimia partajată poate fi citită simultan de către mai multe procese, dar niciun proces nu scrie

## Curs 7

- Proces = program în execuție care folosește un set de resurse ale sistemului (memorie, procesor, disc, etc)
- umask - marca drepturilor de acces pentru fișierele noi create de către acest proces

- `pid = fork();`
  - ↳ -1 la eroare
  - ↳ 0 în codul copilului
  - ↳ pid în codul părintelui, unde pid este identificatorul de proces al copilului nou-creat
- fork crează un proces copil, clonă a părin.
  - ↳ copilul are o copie virtuală a memoriei virtuale a părintelui
  - ↳ copilul rulează același program ca și părintele
  - ↳ copilul menține descriptorii de fisiere deschise de la părinte
  - ↳ copilul începe viață cu aceleasi valori ale registrilor ca și părintele
- `wait()` și `waitpid()`
  - ↳ -1 la eroare
  - ↳ pid-ul procesului copil care s-a terminat

PE URM. PAGINĂ S. P.!!!

## Stările unui proces

Created - procesul este creat prin intermediul unui apel sistem fork()

Ready Memory - procesul ocupă loc în memoria RAM și e gata pt a fi executat  
- nucleul decide momentul intrării lui în execuție

Ready Swapped - procesul este gata pt a fi executat  
- pentru a intra în execuție trebuie să intre mai întâi în starea Ready memory

Run Kernel - instrucțiunile procesului sunt executate în mod nucleu

Run User - sunt executate instrucțiunile programului

Preeempted - dacă în starea Ready Memory există un proces mai prioritar, procesul curent este trecut în starea Preeempted iar procesul mai prioritar trece în starea Run User, altfel procesul curent trece în starea Run User

Sleep Memory - procesul se află în memorie dar nu poate fi executat până se produce un eveniment care să-l scoată din acestă stare

**Sleep Snagged** - procesul este erasat pe disc, el nu poate fi executat pînă se produce un eveniment care să-l scoată din această stare

**Zombie** - procesul nu mai există însă informarea faptului că s-a terminat, trimisă spre procesorul parinte, nu a fost încă preluată de acesta

## Curs 8

Comunicarea între procese

1. **pipe-uri interne** - aceste "conducte" sunt create în memoria internă a sistemului UNIX respectiv

2. **pipe-uri externe** - aceste "conducte" sunt fisiere de un tip special, numit fifo, deci sunt păstrate în sistemul de fisiere

Canale interne - pipe-ul fără nume

Canale externe - un canal prin care pot comunica două sau mai multe procese, comunicarea facîndu-se printr-un fisier de tip FIFO

**FDE[0]** - citire    **FDE[1]** - scriere

## Descheluri fătă de canalele interne

- funcția de creare a unui canal extern nu produce și deschiderea automată a celor două capete
- un canal extern poate fi deschis la oricare din capete, de orice proces care are drepturi de acces pentru acel fizier fifo
- după ce un proces include un capăt al unui canal fifo, acel proces poate redeschide din nou acel capăt pentru a face alte operații I/O asupra sa

## Curs 3

### Tehnologii de stocare

1. Magnetică - sub forma unor particule magnetizabile încărcate negativ sau pozitiv  
ex: HDD, dischetă, cărătă, bandă mag
2. Optică - sub forma unor puncte sau cavitate micșorabile ce au rolul de a reflecta diferit lumină  
ex: CD-ul, DVD-ul
3. Electronica - prezinta / absenta unui curent electric în anumite circuite sau modificată anumitor circuite  
ex: RAM, memoria flash

## Tabelul FAT

- conține informații de alocare a spațiului pe disc pentru fiecare fișier
- conține atâta intrări câtă clusteri (grupuri de sectoare) există pe disc

## Curs 10

Tehnica zonelor tampon temporare

- La citirea din fișier: producător: purif. extern  
consumator: procesul
- La scrierea în fișier: producător: procesul  
consumator: purif. extern
- cache disc = copii al celor mai recente  
sectoare disc accesate
- cache Kilt = copii al celor mai recent pagini  
nile accesate
- În memoria cache producătorul și con-  
sumatorul este memoria internă.

## Curs 11

Procesele paralele sunt acele procese care pot fi executate în paralel; acestia nu se interconditioñă reciproc, în timpul execuției, și nu colaborează între ele.

Procesele concurente sunt acele procese care pot interconditioñă reciproc în timpul execuției în paralel, de exemplu prin folosirea în comun a unor resurse.