

Ch2:

Functii de sistem

Structura sist de operare

Ch3: Procese

Cum arata un proces in memoria (stiva, heap, adresa unui proces)

Starile prin care trece un proces (load, running, asteapta eveniment io, gata de executie da nu se execute din varii motive, terminat)

Coada de gata

Context switch (schimb de la un proces la altul)

Struct arborescenta (fork + execve + wait)

Modele de mem (mem partajata, coada de ms (doar creare + citire ms noi))

Prob clasice (prodicator-consumator)

Ch4: Fire de executie

Threadurile au loc in zona aceluiasi proces

(mai putin registrii, stiva si thread local storage)

Modele de threaduri (embeded (many to one), moder (one to one), mixt (many to many))

Pthreads

Ch5: Sincronizare

Race condition (modelul producator-consumator) (nu apare mereu) => zona critica

Scheduling -> nivel masina (primitive de sincronizare)

Zona critica

Excluziunea mutuala (1 sau mai multe procese nu se pot executa concomitent in zina critica)

Progresul (un sg proces intra in zona critica)

Timpul finit de asteptare (orice proces din lista de asteptare v-a intra in zona critica intr-un timp finit) => parcurgere proces + dat drumul la procese in mod echitabil

Deadlock (se blocheaza mai multe procese

indefinit)

Starvation (un proces se blocheaza definit, nu ii mai vine randul sa se execute)

Priemptiv (un proces poate fi intrerupt oricand din executie) vs Nonpriemptiv (mai eficient) (prioritati)

Alg lui peterson

Analizarea unui alg pt cele 3 prop

Operatii atomice (nu pot fi intrerupte in timp ce se executa) - test and set, compare and swap (mai flexibil)

Mutex => busy waiting + spinlock

Semafor (generalizare mutex - pot intra si procese)

Pb clasice (bounded-buffer (similar producator-consumator), cititori-scriitori, filozofi (stau si se gandesc, mananca, le este foame))

Analizare + rez implementare pt una din cele 3 pb

Ch6: Scheduling

Operatii de tip CPU + IO

Indicatori (Throughput (cate procese se executa pe min), Turnaround time, Waiting time (cat astepta un proces ca sa fie executat), Respons time)

Alg de scheduling (first come first served, shortest job first, priority scheduling, round robin (timp finit de executie pt fiecare proces), cu coada, in timp real (cu deadlineuri))

Ch8: Memoria principală

Segment

Pagina

Ce emite un CPU?

Adrese logice vs virtuale vs fizice

Swaping

Alocare de mem (first fit, best fit, worst fit)

Fragmentare int vs ext

Segmentarea (imp unui program in mai multe

bucati)

Paginare (caz part de segmentare) - doar fragmentare interna

Tabela de pagini

Ch9: Memoria virtuala

= optimizare paginare

Page fault (intelegerere mecanism)

Alg de page replace (first in first out, alg optim(alegem pg care nu va fi fol cel mai mult timp de acum in colo), cea mai recenta utilizata (inlocuieste pg care nu a fost fol de cel mai mult timp))

Ch11: Sisteme de fisiere

Fisier

Director

Partitie (contine ierarhia de directoare)

Disc

Operatii pe fisiere

Acces secvential vs direct

Structuri clasice (cu un sg nivel (lista fisiere

ce indica catre datele de pe disc), 2 nivele (directoare pt fiecare utilizator), arborescenta (clasica))

Mountpoint (cum aduc o partitie si cum o leg in partitia prezenta a sistemului)

Drepturi de acces (read, write, execute pt owner, group, writers)

Ch12: Implementare
Ierarhie sistem fisiere

Tipuri de sist fisiere

Director -> file control block

Virtual file system

Alocare contigua (fiecare bloc de pe disc reprezinta un o bucata de fisier ce se lega de urm bucata)

Alocare bazata pe extent (un fis poate fi imp in mai multe bucati contigue de mem - segmentare)

Alocare prin lista inlantuita (continutul bucatii din fisier + urm adresa a acestuia in mem)

Alocare fat ???

Alocare indexata (bloc principal ce contine adresele celorlalte blocuri unde se gasesc bucatile de fisier)

Spatiul liber (vector de biti - primul bit setat reprezinta primul bloc liber)

!!!!Avem voie cu materiale

Treci cu 25 de p ?????

Grila/ campuri numerice de completat

1. 10 p

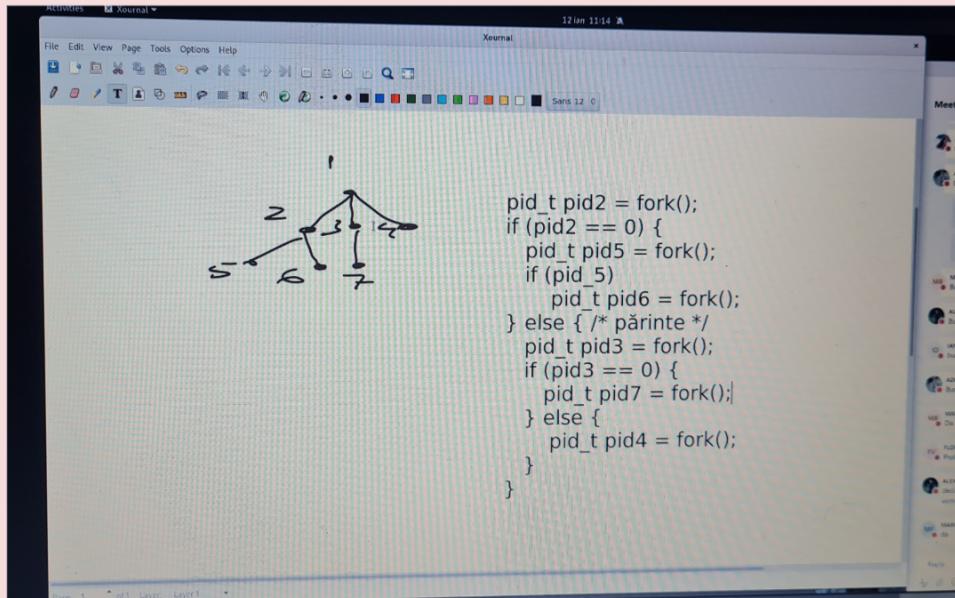
Dif dintre pagina si segment (cum imp mem in tabela de mem)

Sau

Dif de alocare a blocurilor intre liat si fat + exsmplificafe fisier de 4 blocuri

2. 10 p

Arboreacenta data de procese, de scris cod ca sa creeze arborescenta



Sau

Primind secv de cod sa se reproducă arborescenta

3. 15 p

Data sol pb bounded buffer

```
do {  
    wait (empty);  
    wait (mutex);  
    /* PRODUCE ITEM */  
    signal (mutex);  
    signal (full);  
} while (true);
```



```
do {  
    wait (full);  
    wait (mutex);  
    /* CONSUME ITEM */  
    signal (mutex);  
    signal (empty);  
} while (true);
```



```
mutex = 1  
empty = n  
full = 0
```

Care este rolul lui empty (5 p) (2-3 prop) - sa nu pot lua dc nu exista
Adaugare cod consumator, unde e comentat (5 p) - facut ceva cu bufferul
Aratati dc sol satisface timp finit de asteptare (5 p) - nu avem timp finit de asteptare (nu avem cod explicit pt asta, dc avem n+1 producatori cum te aaiguri a unul nu asteapta indefinit) (contraexemplu sau explicatie dc nu)

Sau

Pb scriitorilor

Rolul read count (5 p)

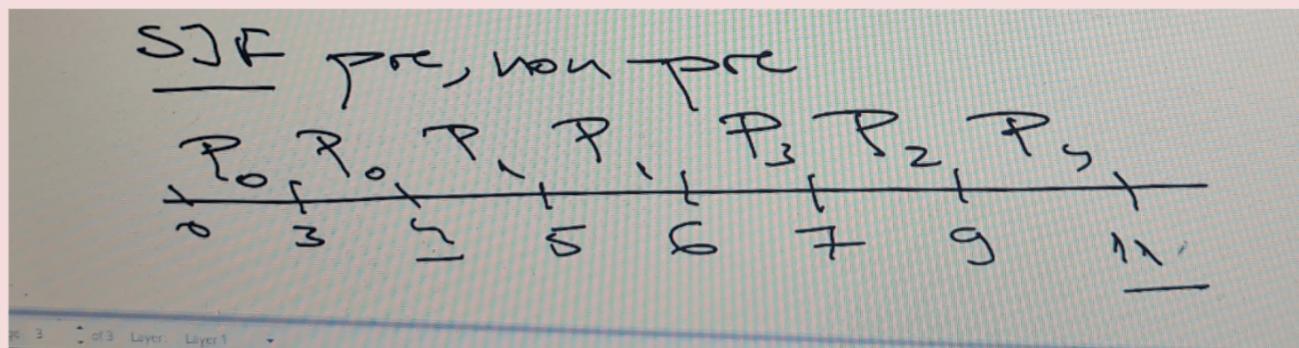
Se satisfac prop de scheduling (10 p)

4. 15 p

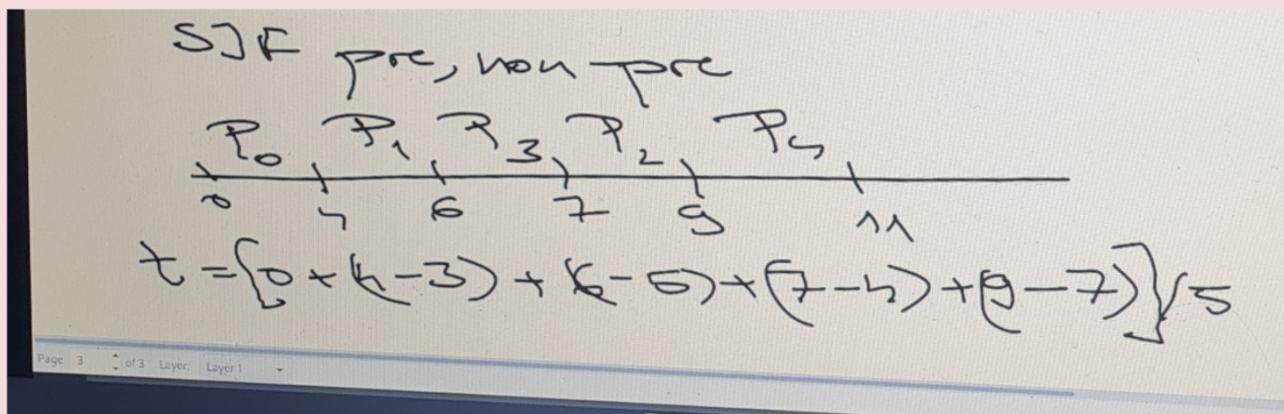
Date procesele

Proces	t	CPU
P ₀	0	5
P ₁	3	2
P ₂	5	2
P ₃	5	1
P ₄	7	2

Cum arata diagrama in modul priemptive? (5 p)

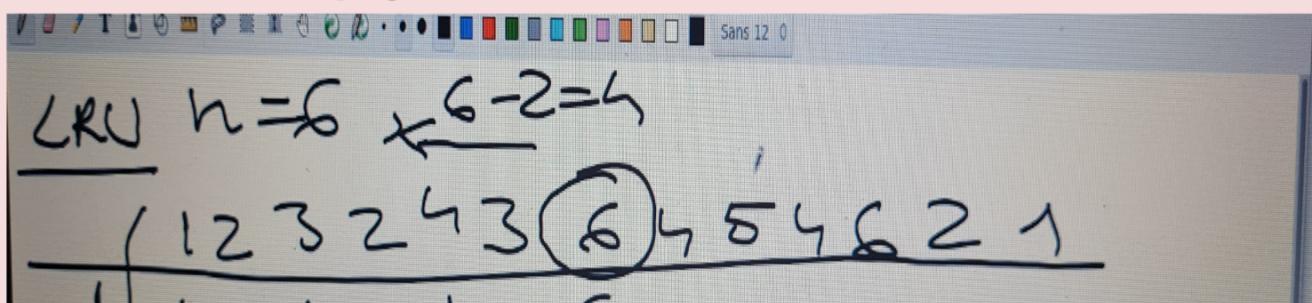


Cum arata diagrama in modul nonpriemptive? (5 p)



Sau

Aducerea de pg in mem



Cat este n? (5 p) 6

Alg de page replace (cele mai recent folosite raman)? (10 p)

