Intrebari SO

1. Write a regular expression that accepts lines that contain the letter “a” but do not contain the letter “b”.

R: ^[^b]*a+[^b]*$

2. What is the maximum number of child processes, created by the code fragment below, that can coexist simultaneously?

R: 5 procesese

3. Processes A, B and C communicate through FIFOs X, Y, Z according to the

diagram below. Sketch the code fragments that open the FIFOs in the 3 processes.

A -- X --> B

B -- Y --> C

C -- Z --> A

A: A2B = open (“X”, O_WRONLY)

C2A = open (“Z”, O_RDONLY)

B: B2A = open (“X”, O_RDONLY)

B2C = open (“Y”, O_WRONLY)

C: C2B = open (“Y”, O_RDONLY)

C2A = open (“Z”, O_WRONLY)

4. How many threads would you use for processing a million files? Justify your Choice.

R: O sa folosim numarul maxim de threaduri pe care ni le ofera procesorul nostru.

5. Give a set of values for T, N1, N2 and N3 for which the program will end.

```
pthread_barrier_t b1, b2, b3;
void* f1(void* a) {
pthread_barrier_wait(&b1);
return NULL;
}
void* f2(void* a) {
pthread_barrier_wait(&b2);
return NULL;
}
void* f3(void* a) {
pthread_barrier_wait(&b3);
return NULL;
}
int main() {
int i;
pthread_t t[T][3];
pthread_barrier_init(&b1, N1);
```

```
pthread_barrier_init(&b2, N2);
pthread_barrier_init(&b3, N3);
for(i = 0; i < T; i++) {
pthread_create(&t[i][0], NULL, f1, NULL);
pthread_create(&t[i][1], NULL, f2, NULL);
pthread_create(&t[i][2], NULL, f3, NULL);
}
for(i = 0; i < T; i++) {
pthread_join(t[i][0], NULL);
pthread_ join (t[i][1], NULL);
pthread_ join (t[i][2], NULL);
}
pthread_barrier_destroy(&b1);
pthread_barrier_destroy (&b2);
pthread_barrier_destroy (&b3);
return NULL;
}
```

R: T = N1 = N2 = N3

T – o valoare, N1, N2 si N3 div ai lui T

6. Why I/O operations cause a process to move from the state RUN to the state WAIT?

R: Ideea este ca operatiile de input si output sunt mult mai lente fiind realizate de utilizator, in timp ce in starea RUN, operatiile sunt realizate de catre procesor, a carui viteza este net superioara utilizatorului.

7. HOW is the address calculation done in the absolute fixed partition allocation?

R: Nu sunt efectuate calcule pentru a afla adresa intr-o partitie fixa absoluta deoarece sunt cunoscute inca de la alocare adresele acestora, acestea fiind batute in cuie.

8. Give an advantage and a disadvantage of the First-Fit placement policy versus the Worst-Fit.

Avantaj : First-fit placement este mai rapid decat Worst-fit

Dezavantaj: Fragmentarea se produce mai rapid la First-Fit.

9. What is the most prioritary memory page that the NRU replacement policy chooses as victim page?

R: Pagina pe care NRU replacement policy o alege ca si victima este cea de clasa minima, incepand de la clasa 0 (00) pana la clasa 4(11).

10. Considering that the size of a block is B and the size of an address is A, how many data blocks are addressed by the triple indirect addressing of an i-node?

R: (B/A) ^ 3

SO Exam 27 iun …

1) Scrieti o comanda UNIX care afiseasza toate liniile din fisierul a.txt care contin cel putin un numar binar multiplu de 4 cu cinci sau mai multe cifre (ex : 010100).

R: grep -E "^.*[10][10][10]+00[^1]*$" a.txt

2) Scrieti o comanda UNIX care inverseaza toate perechile de cifra impara urmata de vocala(ex: a23e8i97u3 -> a2e38i9u73)

R: echo "a23e8i97u3" | sed -E 's/([13579])([aeiouAEIOU])/\2\1/g'

3) Scrieti o comanda UNIX care afiseaza toate scorurile de fotbal unice(ex:4-0) unice care apar in fisierul a.txt. Numarul de goluri poate avea maximum 2 cifre

R: grep -E "^[^0-9]*[1-9]?[0-9]-[1-9]?[0-9][^0-9]*$" a.txt | sed -E 's/([^0-9-])//g' | sort | uniq

6) Cate procese va crea fragmentul de cod de mai jos, excluzand procesul parinte initial?

If(fork() != fork())

Fork();

R: 6

8) Ce tipareste in consola fragmentul de cod de mai jos?

```
char* s[3] = {"A", "B", "C"};
for(i = 0; i < 3; i++) {
        if(fork() != 0)
        execl("/bin/echo", "/bin/echo", s[i], NULL);
}
```

R: A

B

C

9) Ce face apelul sistem “write” cand in Pipe este spatiu, dar nu suficient pentru cat I se cere sa scrie?

R: Apelul sistem write o sa scrie atat spatiu cat este disponibil in write

10) Ce tipareste fragmentul de cod de mai jos daca niciun alt proces nu deschide FIFO-ul “abc”? Justificati raspunsul

```
Int w, n, k = 10;

R = open(“abc”, O_WRONLY);

N = write(R, &k, sizeof(int));

Printf(“%d\n”, n);
```

R: Nu tipareste nimic, programul blocandu-se pentru ca nu este respectat mecansimul de lucru cu FIFO (nu este un alt program ce sa deschida fifo-ul pentru citire).

11) Ce se intampla cu procesele zombie ale caror parinte s-a terminat?

R: Procesul o sa ramana zombie.

12) Considerati ca functia f este executata simultan de 10 thread-uri. Adaugati liniile de cod necesare ca sa asigurati ca n va avea valoarea 10 dupa ce thread-urile isi incheie executia?

```
Int n = 0;

Void* f(void* p){

N++;
```

```
Return NULL;
}
```

R:

```
#include <pthread.h>
int n = 0;
Pthread_mutex_t mtx = PTHREAD_MUTEX_INITIALIZER;
Void* f(void* p){
Pthread_mutex_lock(&mtx);
N++;
Pthread_mutex_unlock(&mtx);
Return NULL;
}
```

13) Planificati executia job-urilor urmatoare (date ca Nume/Durata/Termen) incat suma intarzierilor job-urlior sa fie minima: A / 22 / 27, B / 2 / 15, C / 4 / 5

R: C B A

14) Dati un avantaj si un dezavantaj a cache-urilor set-asociative fata de cele directe

Avantaj: Sansele de cache collision sunt mai mici la set-asociative

Dezavantaj: In momentul cautarii unui bloc, la set-asociative, blocul trebuie cautat secvential in grup, in timp ce la cea directa ii aflam pozitia instant(fiind un singur bloc asignat unei valori)

15) exact ca intrebarea 9

16) Ce ati adauga la fragmentul de program de mai jos incat sa tipareasca in consola “1 3 3”? Scrieti liniile de cod si specificati intre ce linii ale codului existent le-ati adauga. Modificarile nu au voie sa elimine din executie liniile de cod originale.

R: #include <stdio.h>

#include <pthread.h>

```
int n = 0;
pthread_mutex_t m[3];
void* f(void* p){

int id = (int)p;
pthread_mutex_lock(&m[id]);
n += id;
printf("%d ", n);
pthread_mutex_unlock(&m[(id+1)%3]);
return NULL;
}

int main(){

int i;
pthread_t t[3];
for(i = 0; i < 3; i++)
pthread_mutex_init(&m[i], NULL);
```

```
pthread_mutex_lock(&m[0]);          // linia 1 adaugata
pthread_mutex_lock(&m[2]);          // linia 2 adaugata


for(i = 0; i < 3; i++)
pthread_create(&t[i], NULL, f, (void*)i);


for(i = 0 ; i < 3; i++)
pthread_join(t[i],NULL);


for(i = 0 ; i < 3; i++)
pthread_mutex_destroy(&m[i]);


return 0;
}
```

17) Dandu-se doua cache-uri set-asociative, unul cu 2 seturi de 4 pagini si unul cu 4 seturi de 2 pagini, care va da rezultate mai bune pentru secventa de cereri de pagini: 14, 23, 1, 16, 1, 23, 16, 14. Justificati raspunsul!

R: Ambele variante pot fi aplicate pentru a retine secventa de pagini, dar a doua varianta este mai eficienta datorita numarului de pagini mai redus per set, lucru eficientizeaza cautarea unei pagini.

18) e ca 10 de la info eng

19) Ce se intampla cu un link hard cand fisierul spre care puncteaza este sters?

R: Linkul ramane acelasi si fisierul poate fi accesat in continuare, continutul acestuia fiind identic cu, continutul fisierului sters.

20) Dati o metoda pentru prevenirea(evitarea) impasului (deadlock-ului) in conditiile in care nu poate fi evitata modificarea concurenta a resurselor.

R: Alegem o ordine (oricare) pentru resurse (folosind un mecanism de sincronizare) si le blocam intotdeauna in aceeasi ordine.

So Exam 24 iun

1) Afisati lin ce contin cel putin un timestamp hh:mm:ss:SSS

R: grep -E "^[^0-9:]*([0-9]{2,2}:){3,3}[0-9]{3,3}[^0-9:]*$" a.txt

2) e ca 2 de mai sus

3) Afisati lin din a.txt ce contin toate ap unice ale lui Pi cu 2 sau mai multe zecimale pi = 3.14...

R: grep -E -o "\<3\.14[0-9]*\>" a.txt | sort | uniq

6) Cate procese fara parinte?

```
If(fork() || fork()){
fork()
}
```

R: 4

7) Cate procese se fac cand parintele apeleaza f(3)?

```
Void f(int n){
If(n > 0 || fork() == 0){
f(n-1);
```

```
exit(0);
        }
        Wait(0);
}
```

R: E un fork bomb(prietenii stiu de ce).

8) Ce afiseaza codul?

```
char* s[3] = {"A", "B", "C"};
for(i = 0; i < 3; i++) {
        if(fork() == 0)
        execl("/bin/echo", "/bin/echo", s[i], NULL);
}
```

R: Programul afiseaza cele 3 litere, dar se blocheaza pentru ca nu se apeleaza wait-ul.

9) Ce face apelul sistem “read” cand in Pipe nu este informatie suficienta pentru cat este ceruta? Citeste octetii ce sunt in pipe, returnand un numar mai mic decat cel cerut de octeti.

R: Citeste octetii ce sunt in pipe, returnand un numar mai mic decat cel cerut de octeti.

10) Ce o sa afiseze codul?

```
Int p[2];
Char buf[10];
Int n;
Pipe(p);
```

N = read(p[0], buf, 10);

Printf(“%d \n”, n);

R: In cazul de fata nu se va afisa nimic deoarece capatul de scriere al pipe-ului nu a fost inchis. In cazul in care acesta era inchis, se afisa valoarea 0.

11) De ce procesele zombie sunt o problema?

R: Procesele zombie ocupa loc degeaba in tabela de procese, aceasta avand o capacitate limitata.

12) e exact ca pb 12

13) Planificati urmatoare actiuni obtinand astfel un timp de intarziere minim:

A / 7 / 13 | B / 5 / 9 | C / 2 / 4

C B A – timp de intarziere 1 sec

14) e facut mai sus

15) Care pagina are prioritate in a fi eliminata de catre strategia LRU?

R: Pagina ce are prioritate in a fi eliminata este cea cu suma minima pe linie.

16) Dandu-se doua cache-uri set-asociative, unul cu 2 seturi de 4 pagini si unul cu 4 seturi de 2 pagini, care va da rezultate mai bune pentru secventa de cereri de pagini: 20, 9, 18, 27, 20, 9, 18, 24 Justificati raspunsul!

R: In cazul de fata nu este buna vreo varianta. Dar, daca ar fi fost 27 in loc de 24, ar fi fost bune ambele variante, dar mai eficienta este a 2-a varianta cu 4 seturi de cate 2 pagini

17) Cate blocuri pot fi accesate de catre tripla indirectare a unui I-node, daca un bloc contine N blocuri ce pot fi adresate?

R: N ^ 3

19) e ca 20 de la testul anterior

SO Exam ex 1

1) Write a UNIX Shell command that displays the lines in file a.txt that contains words starting with capital letters.

R: grep -E "\<[A-Z]+[a-z]*\>" a.txt

2) Write a UNIX Shell command that inverts in file a.txt all pairs of neighboring digits(ex: a3972b -> a9327b)

R: echo "a3972b" | sed -E 's/([0-9])([0-9])/\2\1/g'

4) Display only the lines of file a.txt that appear only once(not duplciated).

R: cat a.txt | sort | uniq -c | grep -E "^[ ]*1 " | sed -E "s/^([ ]+1)//"

R: cat a.txt | sort | uniq -c | awk '{if($1 == '1') print $0}' | sed -E 's/(^[ ]*1)//g'

R: cat a.txt | sort | uniq -c | grep -E "^[ ]*1.*" | sed -E "s/([ ]*1)//g"

5)

```
#!/bin/bash
for F in `find $1 -maxdepth 1 -type f -name "*.txt"`; do
    if `cat $F | grep -E -q "\<cat\>"`; then
        echo $F
    fi
```

Done

6) In the program fragment below, mark which process executes each line: the Parent, the CHild, or both.

K = fork();(1)

If(k==0){(2)

Printf("A\n"); (3)

}else{(4)

Print B (5)

}

Print C (6)

	P	C
1	x	x
2	x	x
3		x
4	x	
5	x	
6	x	x

7) How many processes will be created by the code fragemnt below, excluding the initial parent process?

Fork(); wait(0); fork(); wait(0); fork();

R: 7 = 2^k – 1 (k = nr de fork-uri)

8) What are the possible console outputs of the following code fragment (ignoring any output that execl might generate), and when will they happen?

R: Depinde A sau AB, A daca execl executa cu succes comanda primita ca si parametru, altfel AB.

9) What does the system call "read" do when the pipe is empty?

R: Citeste octetii ce sunt in pipe, returnand un numar mai mic decat cel cerut de octeti.

10) What does the system call “open” do before returning from opening a FIFO?

R: Sistemul asteapta ca FIFO-ul sa fie deschis in alta parte(dar in modul complementar), dupa care returneaza descriptorul acestuia.

11) Give a reason for choosing threads over processes?

R: Threadurile nu folosesc PID-uri si nu dubleaza memoria.

12) Considering that functions “fa” and “fb” are run in concurrent threads, what will the value of “n” be after the threads are finished? Why?

R: Neavand main-ul nu putem spune ordinea in care se executa functiile, dar in cazul in care sunt pornite in acelasi timp nu putem spune cu acuratete valoarea finala a lui n deoarece sunt folosite 2 mutex-uri diferite.

13) Schedule the following jobs(given as Name/ Duration/Deadline) so that they all meet their deadlines: A / 5 / 9, B / 7 / 13, C / 1 / 10

R: ACB / CAB

14) Give one advantage and one disadvantage of the segmented allocation method over the paged allocation method.

R: Advantage: Se poate accesa mai multa memorie decat dimensiunea maxima a adresei.

Disadvantage: Se fragmenteaza memoria.

15) When would you load into memory the pages of a program that is being started?

R: La inceput o sa incarcam o pagina si vecinii acesteia, dupa aplicam principiul vecinatatii.

16) When does a process change state from RUN to READY?

R: Un proces trece din starea Run in starea Ready in momentul in care ii expira timpul de executie alocat.

17) Given a UNIX file system configured with a block size of B bytes that can contain A adresses, and I-nodes having S direct link, one simple indirection link, one double indirection link, and one triple indirection link, give the formula for the maximum file size possible.

R: S * B + A * B + A ^ 2 * B + A ^ 3 * B bytes

18) am mai avut-o

19) si pe aceasta

20) What is a binary semaphore, and what is the effect of its P method, when called by multiple concurrent processes/threads?

R: P – Wait

V – Signal

Un semafor binar functineaza aproape ca un mutex, iar metoda lui P, are rolul de a bloca resursa pe care acesta o protejeaza.

SO Exam ex2.pdf

1) Scrieti o comanda unix ce afiseaza randurile ce contin cel putin un numar cu cel putin 2 zecimale

R: grep –E “^.*[0-9]\.[0-9]{2,}.*$” a.txt

2) Scrieti o comanda unix ce elimina toate caracterele ce nu sunt litere dintr-un fisier a.txt

R: sed -E 's/[^a-zA-Z]//g' a.txt

3) Scrie un awk care aplicat unui fiser retunreaza media de cuvine pe linie

R: awk '{nr_cuv = nr_cuv + NF} END {print nr_cuv / NR}' a.txt

7) Cate procese vor exista la apelul f(3)?

```
Void f(int n){
If(n > 0 && fork() == 0){
f(n-1);
exit(0);
        }
        Wait(0);
}
```

R: Numarul de procese pe care-l creeaza programul este 3, iar procesul parinte al unui fiu se termina in momentul in care copilul acestuia este terminat si el.

8) What will the code fragment below print to the console?

Char* s[3] = {“A”,”B”,”C”}

For(I=0;I<3;I++){

Execl(“/bin/echo”, “/bin/echo”, s[I], NULL);

}

R: A

Sintaxa execl(path catre functie, numele, parametrii, NULL)

9) Ce face apelul sistem “read” FIFO daca contine mai putina informatie decat ceruta?

R: Apelul sistem read citeste cate date sunt in FIFO-ul nostru si returneaza numarul de octeti cititi.

10) Ce afiseaza fragemntul de cod de mai jos daca nu deschide alt proces fifo ul abc?

R: Codul o sa ramana blocat dupa primul open.

11) Ce se intampla cu un proces inainte ca acesta sa se termine si ca parintele acestuia sa apeleze wait.

R: Procesul asteapta sa vina parintele cu wait-ul, iar in momentul in care parintele il apeleaza, se elibereaza memoria alocata pentru copil.

13) Planificati urmatoarele joburi: A/7/13 | B/5/9 | C/2/4

R: CBA – intarziere 1 sec

14) Dati un avantaj si un dezavantaj al set-associative caches vs cel asociative caches

R: Avantaj: Sansele de cache collision sunt mai mici.

Dezavantaj: Este mai lent in procesul de cautare al unei pagini pentru ca in varianta set-asociativa trebuie sa ne deplasam la un set in memorie, dupa care trebuie sa parcurgem acel set, in timp ce la asociativ trebuie doar sa ne deplasam la set-ul respectiv.

15) Care este categoria pag ce poate fi aleasa ca victima in NRU replacement?

R: Paginile de clasa minim, pornind de la clasa 0(nu s a citit, nu s a scris).

17) Cate data blocks pot fi referenced de catre double indirection of an I-node, daca un bloc contine n adrese catre alte blocuri?

R: N^2

18) Problema producator-consumator

R: Avem nevoie de 2 semafoare: un semafor numit gol ce este initializat cu N si plin initializat cu 0, atunci cand un consumator consuma, da un post la gol si un wait la plin, invers la producator. Sa nu uitam ca pentru a proteja variabila globala am mai avea nevoie de un mutex sau de un semafor binar.

19) O metoda pentru a evita deadlock-ul?

R: Alegem o ordine (oricare) pentru resurse (folosind un mecanism de sincronizare) si le blocam intotdeauna in aceeasi ordine.

20) Completati fragmentul de cod de mai jos pentru a redirecta inputul de la comanda bin/pwd sa fie citit din pipe

PIPE p;

Int p[2];

Pipe(2);

If(fork() == 0){

Execl(“/bin/pwd”,”/bin/pwd”, NULL);

Exit(0);

}

R: int p[2]; /// -> p[0]: read; p[1]: write

pipe(p);

if(fork()==0){

```
dup2(0,p[0]); // linia adaugata
execl("/bin/pwd","/bin/pwd",NULL);
exit(0);
```

}

SO Restanta iul2019.pdf

1) Scrieti o comanda unix ce afiseaza pe ecran liniile ce contin cel putin o distanta ex 5 km

R: grep –E “^.*[0-9]+ km.*$” a.txt

2) Scrieti o comanda unix ce inverseaza un grup format din o vocala mare si o cifra para

R: sed -E 's/([AEIOU])([02468])/\2\1/g' a.txt

6) Cate procese o sa fie facute de secv de cod de mai jos fara parinte?

```
if(fork() || fork())
        Fork();
```

R: 4