Exam – 2018/2019

**1.** Scrieti o comanda UNIX care afisaza toate liniile din fisierul a.txt care contin cel putin un numar binar multiplu de 4 cu cinci sau mai multe cifre.

**SOLUTION**

grep -E ".\*[01]{2,}100.\*" a.txt

grep -E "\<[01]{2,}100\>" a.txt

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

**SOLUTION**

sed -E "s/([13579])([aeiou])/\2\1/gi" a2.txt

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

**SOLUTION**

grep -E "\<[0-9]{1,2}-[0-9]{1,2}\>" a3.txt | sort | uniq -c | awk '$1==1 {print $2}'

**4.** Afisati numarul de procese ale fiecarui utilizator active din sistem.

**SOLUTION**

ps -ef | awk '{print $1}' | sort | uniq -c

**5.** Scrieti un script Shell UNIX care calculeaza media de fisiere cu extensia .txt per director din directorul current si toate subdirectoarele lui.

**SOLUTION**

#!/bin/bash

nrDir=0

nrFiles=0

nrDir=`find -type d | wc -l`

nrFiles=`find -type f -name "\*.txt" | wc -l`

echo $((nrFiles/nrDir))

**6.** Cate procese vac rea fragmentul de cod de mai jos, excluzand procesul parinte initial?

if(fork(0 != fork()){

fork();

}

**SOLUTION**

Vor fi 6 procese fara parinte.

**7.** Desenati ierarhia de procese generate de codul de mai jos:

int p=0;

for(i=0;i<3;i++){

if(p==0)

p=fork();

else

wait(0);

}

**SOLUTION**

4 procese cu tot cu parinte

**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);

}

}

**SOLUTION**

ABC

**9.** Ce face apelul sistem “write” cand in PIPE este spatiu, dar nu suficient pentru cat i se cere sa scrie?

**SOLUTION**

It will write to the pipe the data for which there is actually enough space and will return the number of bytes written.

**10.** Ce tipareste fragmentul de cod de mai jos daca niciun alt process 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);

**SOLUTION**

It won’t print anything to the console because the process will wait for the other process to open the same FIFO for read, reason why it will be locked.

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

**SOLUTION**

They will either be closed by the operating system after some time after the execution of the program finished or you will have to close them manually.

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

**SOLUTION**

t=0 initial time

t=0+4 => C finishes at time 4

t=4+2 => B finishes at time 6

t=6+22 => A finishes at time 28

The possible order is: CBA and the delay is 1.

**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.

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]);**

**pthread\_mutex\_lock(&m[2]);**

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.

**SOLUTION**

Cache-ul cu 4 seturi de 2 pagini va da rezultate mai bune pentru secventa de cereri de pagini data, pentru ca din punct de vedere al cautarii elementelor, va fi mai usor si mai rapid sa se parcurga un set de doua pagini, decat un set de 4.

**18.** Cate blocuri de date pot fi referite prin tripla-indirectare a unui i-node, daca un bloc are dimensiunea B si o adresa are dimensiunea A?

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

**SOLUTION**

A hard link of a file points to the data of that file. If that file is deleted, only the pointer for the file to the data is deleted, but the pointer from the hard link to the data will remain, in fact the data on the hard-disk never gets deleted.