1. Sa se scrie o comanda UNIX care afiseaza liniile care au numarul total de caractere divizibil cu 3.

awk ' {if(length($0)%3==0) print $0;}' fis.txt

2. Sa se scrie o comanda UNIX care duplica primele 5 caractere de pe fiecare linie

ex: abc01xyz => abc01abc01xyz

echo "abc01ana" | sed "s/\(^.....\)/\1\1/g"

3. Sa se scrie o comanda AWK care numara cate cate linii au primul cuvant egal cu ultimul.

awk 'BEGIN {g=0} { if($1==$(NF)) g++ } END{print g}' fis.txt

4. Un program Shell care afiseaza numere care sunt multiplu de 15. Fara validari.

#! /bin/sh

while [ $# -gt 0 ]; do

nr=$(($1%15))

if [ $nr -eq 0 ]; then

echo $1

fi

shift

done

5. Un program Shell care afiseaza fisierele date in linia de comanda care sunt executabile si citibile.

#!/bin/sh

while [ $# -gt 0 ]; do

i=$1

v=`find $i -perm -ugo=x -perm -ugo=r`

if [ "$i" = "$v" ]; then

echo $v

fi

shift

done

SAU

while [ $# -gt 0 ]; do

i=$1

if [ -x $i ] && [ -r $i ]; then

echo $i

fi

shift

done

6. Cate procese se creaza (fara procesul parinte) in urma executarii urmatoarelor linii de cod?

int i;

for(i = 0; i < 6; i++)

if(i % 3 == 1) - 3

fork();

7. Ce se va afisa in urma comenzilor cd ~/.././../xyz ; pwd

8. Care este valoarea de adevar a unei functii (parca exit(2)) ?

False

9. Un program Shell care face suma argumentelor din linia de comanda de pe pozitii divizibile cu 8. Fara validari.

#!/bin/sh

suma=0;

pozitie=1;

for i in $@; do

c=$(($pozitie%8))

if [ $c -eq 0 ]; then

suma=$(($suma+$i))

fi

pozitie=$(($pozitie+1))

done

echo $suma

10. Enumerati 2 din 3 tipuri de alocare virtuala de memorie.

alocare virtuala paginata, alocare virtuale segmentata

11. Functie pentru crearea unui FIFO.

- in linie de comanda "mkfifo nume.fifo"

- in C mkfifo("nume.fifo", 0666)

12. Comportamentul functiei read(pentru pipe/FIFO) fara O\_NDELAY.

asteapta pana apar date sau pana cand nu mai exista un alt proces care deschide pentru scriere.

intoarce cate date a citit sau 0 daca nu exista alt proces scriitor

13. Un motiv pentru care un proces poate trece de la starea RUN in WAIT.

-asteapta o operatie de intrare/iesire

-Un proces trece din starea RUNNING ĂŽn WAITING atunci cĂ˘nd executÄ o acĹŁiune blocantÄ.

14. Sa se precizeze intre ce linii trebuie facuta dealocarile pentru urmatoarea secventa de cod:

void\* f(void\* p){

printf("%d\n",\*(int\*)p);

//aici

return NULL;

}

int main(){

int i;

pthread\_t t[3];

int\* x,z;

z = (int\*) malloc(15\*sizeof(int));

//aici

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

x = (int\*) malloc(sizeof(int));

\*x = i;

pthread\_create(&t[i],NULL,f,x);

}

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

pthread\_join(t[i],NULL);

}

return 0;

}

15. Completati programul C astfel incat suma sa se calculeze corect.

//pthread\_mutex\_t mutex = PTHREAD\_MUTEX\_INITIALIZER

int sum = 0;

void \*f(void \* arg){

//pthread\_mutex\_lock(&mutex);

sum++;

//pthread\_mutex\_unlock(&mutex);

}

int main(){

int i;

pthread\_t t[10];

for(i = 0 ; i < 10; i++){

pthread\_create(&t[i],NULL,f,NULL);

}

for (i = 0 ; i < 10; i++)

pthread\_join(t[i],NULL);

}

return 0;

}

16. Componentele unui semafor.

v(s) - valoarea semaforului (numar intreg)

c(s) - coada de asteptare (pointeri la procesele care asteapta la semafor)

17. Programul A scris in programul B prin canalul "a2b", iar programul B

scrie in programul A prin canalul "b2a". Scrieti functiile pentru open in ordinea

corecta astfel incat sa mearga programul.

Programul A: Programul B:

open( "a2b" , O\_WRONLY ); open( "a2b" , O\_RDONLY );

open( "b2a" , O\_RDONLY ); open( "b2a" , O\_WRONLY );

18. exec - ce face in cazul in care se executa cu succes(nu da eroare)

creeaza un program nou in programul curent.

-instructiunile aflate in progrmaul curent nu se mai executa, in locul lor se lanseaza instructiunile noului program

19. Problema cu semafor ... Ce afiseaza programul?

semaphore a,b,c,d;

a.value = 0;

b.value = 0;

c.value = 1;

d.value = 0;

while(1){ while(1){

P(a); P(b);

print "A\n"; print "B\n";

V(d); V(a);

P(c); P(d);

print "C\n"; print "D\n";

V(b); V(c);

//a c d b

20. O metoda de rezolvare a unui impas(deadlock)

alegerea unui proces ,,victima" din procesele implicate in impas si oprirea fortata a lui.