Examen la Sisteme de Operare
2018-2019/2 - Test J

Nume/prenume: _____

Utilizator: _____

Grupa: _____ **Nota:**

Data: _____

1. Scrieţi o comandă UNIX care afiseazătoate liniile din fişierul a.txt care conţin cel puţin un număr binar multiplu de 4 cu cinci sau mai multe cifre (ex: 010100).

2. Scrieţi o comandă UNIX care inversează toate perechile de cifră impară urmată de vocală (ex: a23e8i97u3 -> a2e38i9u73)

3. Scrieţi o comandă UNIX care afişează toate scorurile de fotbal unice (ex: 4-0) unice care apar în fişierul a.txt. Numărul de goluri poate avea maximum două cifre.

4. Afişaţi numărul de procese ale fiecărui utilizator activ din sistem.

5. Scrieţi un script Shell UNIX care calculează media de fişiere cu extensia .txt per director din directorul curent şi toate subdirectoarele lui.

6. Câte procese va crea fragmentul de cod de mai jos, excluzând procesul părinte iniţial?

```
f(fork() != fork()) {
   fork();
}
```

7. Desenaţi 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);
}
```

8. Ce tipăreşte în consolă 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);
  }
}
```

9. Ce face apelul sistem "write" când în PIPE este spaţiu, dar nu suficient pentru cât i se cere să scrie?

10. Ce tipăreşte fragmentul de cod de mai jos dacă niciun alt proces nu deschide FIFO-ul "abc"? Justificaţi răspunsul.

```
int w, n, k=10;
r = open("abc", O_WRONLY);
n = write(r, &k, sizeof(int));
printf("%d\n", n);
```

11. Ce se întâmplă cu procesel zombie ale căror părinte s-a terminat?

12. Consideraţi că funcţia f este executată simultan de 10 thread-uri. Adăugaţi liniile de cod necesare ca să asiguraţi că n va avea valoarea 10 după ce thread-urile îşi încheie execuţia?

```
int n = 0;

void* f(void* p) {

    n++;

    return NULL;
}
```

13. Planificaţi execuţia job-urilor următoare (date ca Nume/Durată/Termen) încât suma întârzierilor job-urilor să fie minimă: A/22/27, B/2/15, C/4/5

14. Dați un avantaj și un dezavantaj a cache-urilor set-asociative față de cele directe.

15. Care este cea mai prioritară categorie de pagini de memorie din care politica de înlocuire NRU are alege o pagină victimă?

16. Ce ați adăuga la fragmentul de program de mai jos încât să tipărească în consolă "1 3 3"? Scrieți liniile de cod și specificați între ce linii ale codului existent le-ați adăuga. Modificările nu au voie să elimine din execuție liniile de cod originale.

```
1  int n = 0;
2  pthread_mutex_t m[3];
3  void* f(void* p) {
4     int id = (int)p;
5     pthread_mutex_lock(&m[id]);
6     n += id;
7     printf("%d ", n);
8     pthread_mutex_unlock(&m[(id+1) % 3]);
      return NULL;

  in() {
    i;
    read_t t[3];
    (i=0; i<3; i++) {
    thread_mutex_init(&m[i], NULL);

    (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. Fiindu-se două cache-uri set-asociative, unul cu 2 seturi de 4 pagini și unul cu 4 seturi de 2 pagini, care va da rezultate mai bune pentru secvența de cereri de pagini: 14, 23, 1, 16, 1, 23, 16, 14. Justificați răspunsul.

18. Câte blocuri de date pot fi referite prin tripla-indirectare a unui i-node, dacă un bloc are dimensiunea B și o adresă are dimensiunea A?

19. Ce se întâmplă cu un link hard când fișierul spre care punctează este șters?

20. Dați o metoda pentru prevenirea (evitarea) impasului (deadlock-ului) în condițiile în care nu poate fi evitată modificarea concurentă a resurselor.