1.Ce primește funcția unlink ca parametru? De ce?

const char *pathname - deorece functia va sterge fisieru aflat la calea data ca parametru

2.Ce întoarce apelul read?

numarul de octeti cititi (size_t) sau negativ in caz de eroare

3.Fie următorul output:

student@localhost:~$ ls -l -r--r--r--. student student 12 Jan 28 03:05 in.txt

Care este rezultatul următorului apel? Câți octeți va avea fișierul în urma acestui apel? int rc = truncate("in.txt", 0);

Functia truncate va returna eroare, deoarece fisierul in.txt nu are drepturi de scriere; asadar in.txt isi va pastra dimensiunea.

4. Fie un apel lseek ce încearcă deplasarea în cadrul unui fișier cu 100 față de sfârșitul fișierului. Ce rezultat va întoarce lseek în această situație?

lseek permite pozitionarea cursorului dupa sfarsitul unui fisier inserandu-se o zona goala, nealocata pe disk. Valoarea de return va fi astfel 0 bytes.

5. De ce este necesar ca în cadrul procesului părinte, apelul fork să întoarcă pid-ul noului proces?

Se returneaza 0 pentru procesul copil, -1 pt eroare si o alta valoare diferita de 0 pt procesul parinte (pid-ul noului proces). Astfel pentru ca parintele sa stie ce nepoti are, are nevoie de pid-ul procesului copil.

6. Daţi exemplu de o situaţie în care apelul CreateProcess poate să eşueze.

Eventual nu poata aloca memorie, a primit parametrii gresiti sau procesul nu poate executa procesul cerut.

7. În ce condiţii sunt moştenţi descriptorii de fişier în urma unui apel exec în Linux?

Descriptorii sunt mosteniti in mod normal, exceptie cei care au flag-ul CLOSE_ON_EXEC.

8. Care este asocierea între un program și un proces?

Un proces reprezinta instanta unui program.Programul este definit printr-un executabil care urmeaza sa fie rulat. In momentul rularii se creeaza un proces care este definit printr-un pid, memorie alocata, registri, SP(stiva),etc. Un program poate avea mai multe instante(procese).Invers nu este adevarat.

9. Care este utilitatea variabilelor de mediu?

Variabilele de mediu sunt utilizate pentru a transmite mai multor programe anumiti parametri ai sistemului. Exemple: PATH-locatiile fisierelor binare LD_LIBRARY_PATH-locatiile bibliotecilor

10. Cum se numeşte funcţia POSIX folosită pentru crearea unui pipe anonim?

Functia posix pentru crearea unui pipe anonim are semnatura int pipe(int filedes[2]);

11. Presupunând că apelurile fork() se execută cu success, câte procese noi se creează rulând următorul cod: if (fork() > 0) fork();

2, deoarece procesul curent creaza 2 procese noi in urma celor 2 apeluri de fork. copilul nu trece de if deoarece este pusa conditia fork() > 0, iar fork() returneaza 0 in cazul unui proces copil.

12. Ce funcţie este folosită pentru a determina pid-ul procesului curent în Windows?

DWORD GetCurrentProcessId(void)

13. Se da următoarea declaraţie int v[10]. Ce dimensiune are sizeof(v) pe un sistem pe 32 de biţi?
v[10] - 10*sizeof(int). "int" pe 32 de biti are 4 bytes. 10*sizeof(int) = 40 (sizeof(v)).

14. Câte cozi de mesaje sunt necesare pentru a crea o comunicaţie de tip REQUEST/REPLY între 2 procese într-un sistem Windows?

2 cozi deoarece cozile sunt unidirectionale

15. Care este funcţia din WIN32 API folosită pentru a extrage un mesaj dintr-o coadă de mesaje (mailslot)?

Pentru a extrage(a citi) un mesaj dintr-o coada de mesaje se foloseste functia ReadFile().

16. Daţi două exemple de situaţii ce generează semnale sincrone.

incercarea de accesare a unei locatii de memorie nevalide/nepermise, impartire la zero

17. Ce presupune/înseamnă acţiunea de blocare a unui semnal pentru un proces?

Actiunea de blocare a unui semnal pentru un proces presupune ca procesul respectiv sa specifice ca doreste sa blocheze un anume semnal, iar sistemul de operare nu va mai trimite semnalele de acel tip spre acesta, dar va salva numai primul semnal, pe restul ignorandu-le. Atunci cand procesul decide ca vrea sa primeasca, din nou, semnale de acel tip, daca există vreun semnal in asteptare, acesta va fi trimis.

18. Ce semnal este trimis atunci când este accesată o zonă de memorie invalidă? Dar când se face o împărţire la 0?

Raspuns corect, dar incomplet. incercarea accesarii unei zone de memorie nevalida - SIGSEGV incercarea impartirii la 0 - SIGFPE

19. Care este funcţie folosită în Linux pentru a asocia un semnal cu un handler? Nu este necesară semnătura funcţiei.

Semnatura functiei: sighandler_t signal(int signum, sighandler_t handler);

20. Care sunt cei trei paşi în folosirea unui timer pe Linux?

creare - timer_create
armare - timer_settime
stergere - timer_delete

21. Este posibil ca apelul mmap intoarca adresa 0xC0000003? Justificaţi.

Adresa de la care incepe maparea trebuie sa fie multiplu de dimensiunea unei pagini(pagina are dimensiunea intre 1-4k), astfel adresa intoarsa de mmap nu poate fi 0xC0000003. Daca se cere explicit ca aceasta sa porneasca de la 0xC0000003, sistemul de operare va gasi o alta adresa apropiata multiplu de dimensiunea unei pagini, iar mmap o va returna pe aceasta.

22. Știind că o pagină are 4K, care este numărul maxim de pagini virtuale alocate pentru vector v, definit astfel: char v[4000].

O pagina.

23. Precizaţi un tip de fişier ce nu poate fi mapat în spatiul de adresă al unui proces.

Un fisier pipe creat cu mkfifo

Nu orice descriptor de fișier poate fi mapat în memorie. Socket-urile, pipe-urile, majoritatea dispozitivelor nu permit decât accesul secvențial și sunt incompatibile din această cauză cu conceptele de mapare. Există cazuri în care fișiere obișnuite nu pot fi mapate (spre exemplu, dacă nu au fost deschise pentru a putea fi citite).

24. Ce semnal este trimis de sistemul de operare unui proces care scrie într-o zonă de memorie protejată la scriere?

SIGSEGV sau SIGBUS

25. Câte pagini virtuale pot referenţia la un moment dat o pagină fizică?

Oricate

26. Precizați un tip de protecție care nu trebuie să lipsească atunci când mapăm codul unui biblioteci în spatiul de adresa al unui proces. Justificați.

Nu trebuie sa lipseasca atunci cand mapam codul unei biblioteci in spatiul de adresa al unui proces tipul de protectie PROT_EXEC. Pentru ca libraria sa poata fi executata/folosita, aceasta trebuie sa aiba PROT_EXEC setat intrucat bitii de protectie ai paginilor virtuale sunt cuplati de kernel cu mecanismul de protectie al paginilor fizice.

27. Care este diferența esențială dintre un mutex și un futex?

Optimizarea constă în testarea și setarea atomică a valorii mutex-ului (printr-o instrucțiune de tip test-and-set-lock) în user-space, eliminându-se trap-ul în kernel în cazul în care nu este necesară blocarea.

28. Care este diferenta dintre mecanismele de sincronizare reprezentate de mutex/semafor si variabilele de conditie? (de reformulat)

Mutex-urile și semafoarele permit blocarea altor fire de execuție. Variabilele de condiție se folosesc pentru a bloca firul curent până la îndeplinirea unei condiții.

29. Precizați o situație în care este mai avantajoasă folosirea kernel level threads decât a user level threads.

Kernel level threads au avantajul ca sunt executate indepentent pe procesor spre deosebire de user level threads. Daca un user level thread se blocheaza atunci se blocheaza si celelalte user level threads (tot procesul).

30. Precizați o situație în care este mai avantajoasă folosirea user level threads decât a kernel level threads.

Folosirea user level threads este mai avantajoasă deoarece se pot alege de catre aplicatie elementele cele mai avantajoase pentru aceasta (de exemplu planificarea acestor thread-uri). Atunci cand se doreste rularea threadurilor pe un singur thread de kernel, un singur core.

31. Prin ce mecanism se pot defini date specifice fiecărui fir de execuție?

Answer:

Thread Local Data (TLD)

Pentru creare/eliberare:
pthread_key_create
pthread_key_delete

Pentru accesare:
pthread_key_setspecific
pthread_key_getspecific

32. Precizați două diferențe între procese și thread-uri.

Procesele nu impart intre ele memorie implicit pe cand threadurile da.
Procesele au alocat un PID, memorie virtuala separata pe cand threadurile folosesc aceeasi zona de memorie sub acelasi proces.

33. Care este rezultatul dacă același fir de execuție va face lock pe un mutex de tipul PTHREAD_MUTEX_RECURSIVE pe care deja l-a ocupat?

V-a incrementa contorul de ocupari.

34. Precizați o metodă de reprezentare a firelor de execuție în sistem (din cele trei existente).

printr-un HANDLE printr-un pseudo-HANDLE, o valoare specială care indică funcțiilor de lucru cu HANDLE-uri că este vorba de HANDLE-ul asociat cu firul de execuție curent printr-un identificator al firului de execuție, de tipul DWORD, întors la crearea firului, sau obținut folosind GetCurrentThreadId.

35. Precizați două exemple în care un fir de execuție își termină execuția.c

TerminateThread, ExitThread

36. Cum se numește mecanismul prin care un fir de execuţie poate crea o zonă de memorie proprie?

mecanismul este Thread Local Storage

37. Precizați două operații atomice executate asupra variabilelor partajate.

interlockedincrement si interlockeddecrement

38. Care este avantajul adus de implementarea thread pooling din Windows?

Nu se creeaza noi threaduri de fiecare data ci se proceseaza threadurile alocate deja eliminandu-se unele probleme de performanta cum ar fi : alocarea de stive proprii pentru threaduri calculele suplimentare impuse de creearea si distrugerea noilor threaduri.

39. Ce mecanism este pus la dispoziție pentru a permite sincronizarea accesului la variabile partajate între mai multe fire de execuție.

Interlocked Variable Access prin functiile interlocked.

40. Precizaţi 2 stări în care poate fi un proces imediat după ce a apelat o funcţie blocantă

Poate fi in starea WAITING sau in starea RUNNING.

41. Cum se numeşte mecanismul cel mai scalabil pus la dispoziţie de Win32 API pentru efectuarea operaţiilor I/O asincrone?

I/O Completion Ports.

42. De ce file pointer-ul asociat cu un file HANDLE pe care se efectuează o operatie asincronă nu are nici o însemnătate?

Intradevar structura Overlapped este folosita, insa file pointer-ul asociat cu un file HANDLE pe care se efectuează o operatie asincronă nu are nici o insemnătate, deoarece un program poate porni mai multe operatii asincrone de citire sau scriere pe un singur handle de fisier asincron.

43. Precizați o diferență dintre funcțiile WriteFileEx și WriteFile.

WriteFileEx, spre deosebire de WriteFile, nu doar scrie datele la buffer-ul de iesire, scrie datele asincron si executa o rutina de terminare la sfarsitul operatiei.

44. Ce comportament are funcția ReadFile pe un HANDLE de fișier ce are flag-ul FILE_FLAG_OVERLAPPED setat?

Functia ReadFile pe un HANDLE de fisier ce are flag-ul FILE_FLAG_OVERLAPPED setat incepe operatia de I/O, iar thread-ul isi continua executia. Thread-ul va va putea folosit HANDLE-ul la terminarea operatiei.

45. Precizaţi un dezavantaj al folosirii funcţiei poll pentru manipularea operaţiilor de I/O faţă de funcţia epoll.

la fiecare apel acelasi set de descriptori este copiat în kernel si invers

46. Ce tip de fișier trebuie să aibă cel puțin unul din fișierele folosite de apelul splice?

Pipe

47. Ce efect are setarea flag-ului O_NONBLOCK pe un descriptor de fişier?

Setarea flagului O_NONBLOCK pe un descriptor de fisier are efectul de a nu bloca citirea si scrierea simultana. Operatiile de scriere si citire nu mai sunt blocante daca acest flag este setat.

48. La ce se referă mecanismul zero-copy?

Zero-copy evita operatiile de copiere a datelor dintr-o zona de memorie in alta.

49. Precizaţi un avantaj al folosirii funcţiei poll pentru manipularea operaţiilor de I/O faţă de funcţia select.

setul de descriptori nu trebuie reconstruit la fiecare apel poll

50. Ce tip de operaţie I/O este realizată cu select? (sincron/asincron, blocant/non-blocant). Explicați.

asincron, non-blocant