APELLIDOS:	**DNI:**
NOMBRE:	
FILA:	**COLUMNA:**

Sistemas Operativos – Facultat d´Informàtica de Barcelona - UPC

Fecha: 19 de Enero de 2009 **Duración: 3 horas**

Notas: Las notas se publicarán el **viernes 23 de Enero** en el RACÓ a las 16:00 horas.

Revisión: La fecha, hora y lugar exactos de la revisión del examen se publicarán junto con las notas. Estad atentos.

ATENCIÓN: Las preguntas se tienen que contestar en las mismas hojas de examen utilizando el espacio reservado a tal efecto. Asegúrate de poner NOMBRE y APELLIDOS, DNI, fila y columna en cada una de las hojas. El examen se tiene que entregar en bolígrafo negro o azul. Se puede utilizar el documento con las llamadas al sistema.

Ejercicio 1 (3 puntos)

Contesteu al mateix enunciat. Les respostes han de ser clares, concises i complertes. RAONEU SEMPRE LA VOSTRA RESPOSTA, una solució no raonada es considerarà malament.

a) (0.3 punts) Un procés UNIX llegeix d'una pipe i cap procés té obert el canal d'escriptura. Es bloqueja el procés? Què retorna? Es genera algun *signal*?

b) (0.3 punts) A la taula de pàgines es guarda, normalment, un bit de presència, un de validesa i un de modificació. Explica què indica cada un d'ells.

c) (0.3 punts) En un procés UNIX amb gestor de disc, executem una instrucció *read*, que llegeix 100 bytes d'un fitxer que ja estava correctament obert. Indica quines estructures del SO consultarem fins que l'operació acabi.

d) (0.3 punts) Un SF té blocs de dades de 2KB, i utilitza un sistema d'indexació multinivell, amb index de 32 bits i on l'inode té 10 punters directes, un indirecte i un doble indirecte. Quina és la mida del fitxer més gran que puc tenir?

e) (0.3 punts) En un SF que implementa RAID 5, amb 5 discs de 200GB...

Quina és la capacitat real d'emmagatzemament del SF?

Quantes lectures en paral·lel podré fer com a màxim? (i en quines circumstàncies)

Quantes escriptures en paral·lel podré fer com a màxim? (i en quines circumstàncies)

f) (0.2 punts) Un procés no ha executat cap instrucció SIGNAL, i rep un signal d'un altre procés. Això implica que el procés receptor mor necessàriament? Raona la teva resposta i, si és negativa, posa un contraexemple.

g) (0.2 punts) Ampliem la memòria física d'un computador d'1 GB a 2 GB. A quines de les següents taules afecta, i com?
Taula d'inodes :

Taula de fitxers oberts :

Taula de pàgines del procés :

Taula de canals del procés :

h) (0.2 punts) Tenim un sistema de segmentació paginada sota demanda, amb pàgines de 2KB. En un instant donat, un procés ocupa 10 KB de codi, 4KB de pila i 100 KB de dades. A partir d'aquell instant, i sense declarar cap variable més, el procés crea una pipe (que ocupa 4KB), crea un fill i rep d'aquest 22 KB de dades, que guarda en una variable global. També obre un fitxer i llegeix 1 KB, que guarda en una variable local. En quantes pàgines ha augmentat cada segment?

i) (0.2 punts) Un sistema operatiu tipus UNIX es penja. Quina informació del Sistema de Fitxers pot quedar incoherent?

j) (0.2 punts) Estem en un SO tipus UNIX i una arquitectura IA32. Com implementaries que la crida *int fork()* retornés un valor diferent al pare i al fill?

k) (0.2 punts) Un procés en *running* rep una interrupció de rellotge. Durant l'execució de la rutina d'atenció a la interrupció (RAI) no es crea cap procés, ni cap dels que està en *blocked* es desbloqueja. Què ha de passar perquè quan s'acabi d'atendre la RAI, el procés en *running* hagi canviat?

l) (0.2 punts) Explica en què consisteix la tècnica *copy-on-write*, especialment el que es fa en cas d'escriptura.

m) (0.1 punts) En un SO tipus UNIX executo la crida *execlp("aux", "aux", ">res", NULL);* què fa?

APELLIDOS:	**DNI:**
NOMBRE:	
FILA:	**COLUMNA:**

Exercici 2 (1 Punt)

Tenim l'aplicació LLegirVector que s'executa amb una comanda com la següent:
_$ LLegirVector Num Sortida
Aquesta comanda crida la nostra aplicació fent que *Num* threads llegeixin un vector (inicialitzat per la pròpia aplicació amb valors aleatoris). Aquests threads escriuen els valors del vector en el fitxer *Sortida*. Cada registre del fitxer mostra el següent format:
"*Thread 1 ha llegit 2 – Element 10 – Num 12345*"
Aquesta línea indica que el thread 1 porta llegits 2 valors del vector i en aquest cas l'element de la 10ª posició del vector és el nombre 12345.

```
 1: #define MAX_POS 100
 2: #define MAX_LOCAL 10
 3: int Vector[MAX_POS], Pos, Pos_Actual, Num, LLegits;
 4: int fd;
 5:
 6: void *llegir(void *arg)
 7: {
 8:    int ThID, len;
 9:    char msn[256];
10:    ThID = (long) arg;
11:    LLegits = 0;
12:    while ((LLegits < MAX_LOCAL) && (Pos < MAX_POS)){
13:        Pos_Actual = Pos;
14:        LLegits++;
15:        Num = Vector[Pos_Actual];
16:        len = sprintf(msn, "Thread %d ha llegit %d - Element %d: Num %d
\n",ThID, LLegits, Pos_Actual, Num);
17:        write(fd, msn, len);
18:        Pos++;
19:    }
20:    len = sprintf(msn, "Thread %d ha llegit %d elements\n", ThID, LLegits);
21:    write(1, msn, len);
22:    pthread_exit(0);
23: }
24:
25: int main(int argc, char **argv)
26: {
27:    pthread_t threadID[500];
28:    int NumThreads, len;
29:    long i;
30:    char msn[256];
31:
32:    for (i=0; i<MAX_POS; i++)
33:        Vector[i] = rand();
34:    NumThreads = atoi(argv[1]);
35:    fd = open(argv[2], O_CREAT|O_WRONLY|O_TRUNC, 0666);
36:    Pos = 0;
37:    /* Creacio Threads */
38:    for (i=0; i<NumThreads; i++)
39:        pthread_create(&threadID[i], NULL, llegir, (void *) i);
40:
41:    /* Control Final Threads */
42:    for (i=0; i<NumThreads; i++)
43:        pthread_join(threadID[i], NULL);
44:
45:    close(fd);
46:    return 0;
47: }
```

Com es pot comprovar, el codi font està implementat **sense protegir les seccions crítiques amb semàfors**. Es demana contestar els següents apartats tenint en compte que:
- *Volem maximitzar el paral·lelisme i reduir el nombre d' instruccions en les seccions critiques.*
- *Cap thread pot començar a accedir al vector fins que tots els threads hagin estat creats.*
- *No podem afegir variables noves al codi si no són semàfors.*

a) Quins problemes hi han en el codi per maximitzar el paral·lelisme executant múltiples threads? Indica quin/s canvi/s faries i les raons.

b) Indica la declaració dels semàfors que afegiries, així com el codi d'inicialització. (Per exemple, "entre les línies 1 i 2: sem_init(...)")

c) Indica les línies que delimiten les seccions crítiques. En la resposta has d'indicar els sem_wait i sem_post que posaries en les línies que creguis oportunes, segons el format de l'apartat anterior.

d) Si no hi hagués informació compartida (per exemple vectors independents) i el processador no és capaç de executar més d'un thread simultàniament, quin disseny presentaria millor rendiment: multi-thread ("pthread") o multi-proces ("fork")? Raona breument la teva resposta.

APELLIDOS:	**DNI:**
NOMBRE:	
FILA:	**COLUMNA:**

Ejercicio 3 (2 puntos)

Disponemos de un ordenador con unos dispositivos de entrada/salida tan rápidos y eficientes que son capaces de servir instantáneamente las peticiones que se realizan. Sabiendo esto, queremos optimizar al máximo el sistema operativo para aprovechar esta característica. Para este ejercicio supondremos que no existe en el sistema ningún mecanismo de sincronización entre procesos y/o threads.

a) ¿Se necesitan gestores de E/S en este escenario? ¿Podemos prescindir de alguna estructura de datos relativa a la entrada/salida en nuestro SO?

b) Dibuja el ciclo de vida más óptimo en este escenario. Numera las transiciones e indica qué eventos fuerzan los cambios de estado. El planificador del que disponemos en el sistema es no apropiativo.

c) Queremos modificar ahora el planificador del sistema y decidimos implementar uno con apropiación inmediata y prioridades. ¿Mejorará este cambio el rendimiento global del sistema?¿Por qué?

d) Añadimos ahora dentro del sistema todos los mecanismos de sincronización vistos en la asignatura. ¿Cambia alguna de las respuestas de los apartados anteriores?

Ejercicio 4 (2 puntos)

a) Implementa los semáforos, junto con sus llamadas al sistema vistas en clase, mediante pipes sin nombre.

b) Comenta las ventajas de esta implementación respecto a la implementación de semáforos vista en clase.

c) Comenta los inconvenientes de esta implementación respecto a la implementación de semáforos vista en clase.

d) Ahora nos planteamos hacerlo al revés: queremos implementar las pipes sin nombre mediante semáforos. Para ello, creamos la siguiente estructura:

```
struct unnamed_pipe
{
  unsigned char vector[4096];
  int pos_ini, pos_fin, num_elements;
};
```

Comenta los problemas que puede tener esta implementación respecto a la implementación de las pipes vista en clase.

APELLIDOS:	**DNI:**
NOMBRE:	
FILA:	**COLUMNA:**

Ejercicio 5 (2 puntos)

Disponemos de un S.O. que gestiona la Entrada/Salida de forma síncrona y mediante el uso de procesos gestores. Por cada dispositivo físico o lógico nuestro sistema dispondrá de un gestor de E/S dedicado a ese dispositivo.

Queremos añadir a nuestro sistema una impresora.

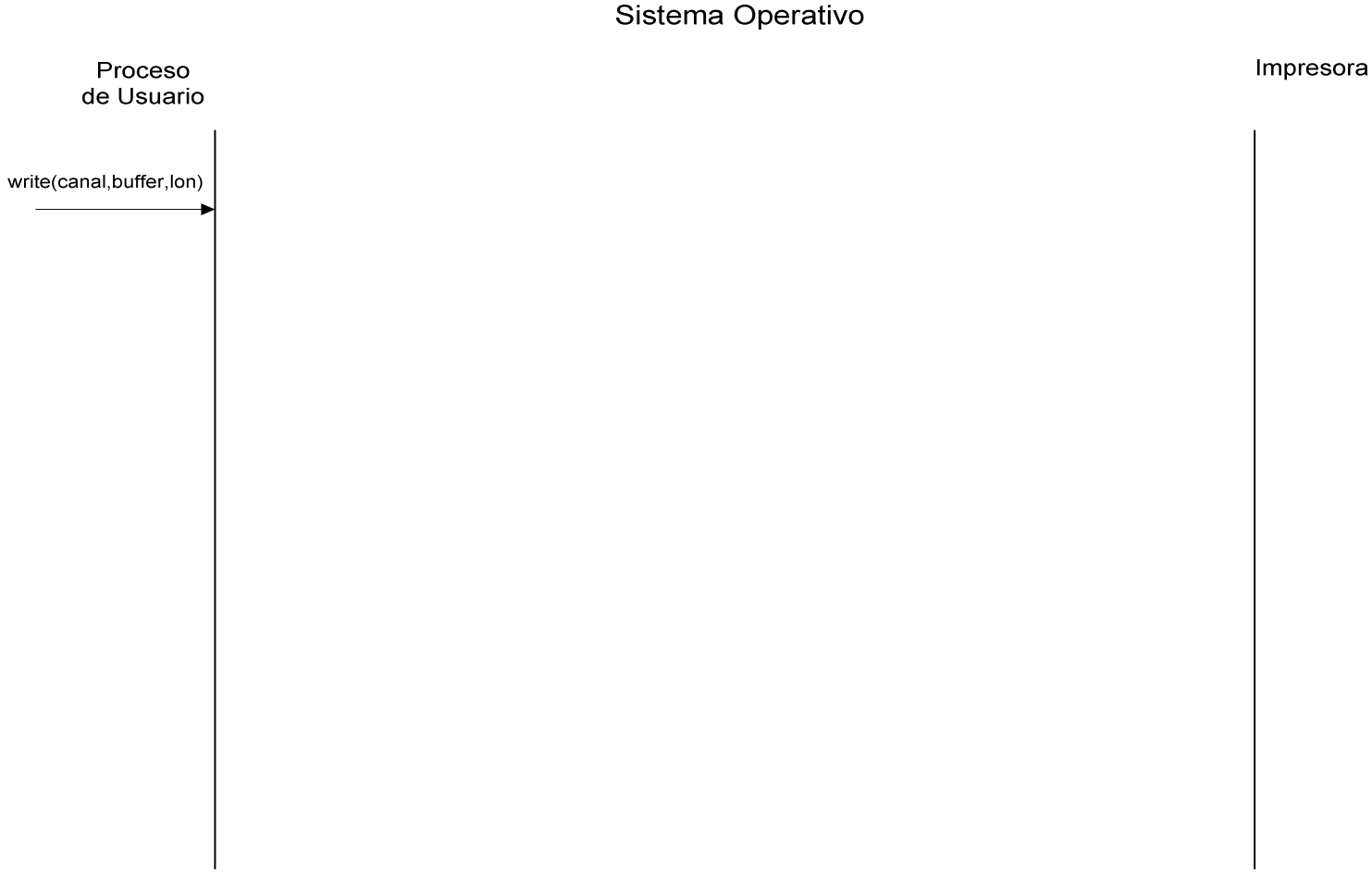
a) Indica qué operaciones/llamadas a sistema de E/S permitirá el nuevo dispositivo. Justifica el uso de cada llamada a sistema que escribas.

b) Indica y justifica qué estructuras y programas se deben añadir o modificar para que funcione la impresora. Supón en este apartado que nuestro sistema sólo permite la ejecución de un único proceso de usuario.

c) Para cada una de las estructuras y de los programas que hayas indicado en el punto anterior justifica en que nivel de E/S (dependiente o independiente) son ejecutados, consultados y modificados.

d) Dibuja esquemáticamente, en la tabla en el reverso de esta hoja, la secuencia de operaciones, junto con las estructuras de datos y programas de sistema, que se realizan en el sistema operativo cuando un proceso de usuario realiza la llamada write(canal,buffer,long) sobre la impresora. Indica el nivel en que se ejecuta cada operación.

e) Se ha modificado el sistema operativo para que sea capaz de gestionar múltiples usuarios y procesos, que pueden compartir concurrentemente la impresora. Suponiendo que la unidad de transferencia mínima de la impresora es la página, ¿plantea este nuevo escenario algún inconveniente al trabajo habitual de los usuarios y sus procesos? Justifica tu respuesta. Si esta es afirmativa, explica cómo solucionarías los inconvenientes.



f) Si el dispositivo impresora está mapeado en /dev/printers/printer001 y si el sistema operativo dispone de un sistema de ficheros basado en indexación encadenada en tabla, ¿cuántos y qué accesos a disco son necesarios para asignar la impresora a un canal de un proceso de usuario? La tabla de encadenamientos siempre está en memoria.

g) Después de imprimir el fichero /usr/users/pepe/so/examen.doc decidimos borrarlo. Este fichero ocupa 53 bloques en disco. ¿Cuántos accesos a disco son necesarios para realizar la operación de borrado? Justifica tu respuesta.