UNIVERSIDAD DE MURCIA

Introducción a los Sistemas Operativos

Examen de teoría (2ª parte del final: temas 5 y 6)
20 de diciembre de 2023. Tipo A.

Apellidos:	Nombre:
Grupo: ☐ 1 ☐ 2 ☐ 3 ☐ PCEO	DNI:

Instrucciones para realizar el test:

1. La puntuación de este test es de 7 puntos.
2. Marca con ≪X≫ y a bolígrafo en la tabla de respuestas la opción que creas correcta.
3. Una respuesta incorrecta resta una respuesta correcta. Una pregunta sin contestar ni suma ni resta.
4. **Debes entregar la hoja del examen al acabar**.
5. Supondremos que 1 KiB = 2^{10} bytes, 1 MiB = 2^{20} bytes y 1 GiB = 2^{30} bytes. También supondremos que **no existe caché de disco** en aquellos casos en los que haya lecturas y/o escrituras de disco.

Respuestas del test

Nº	a	b
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		
12		
13		
14		
15		
16		
17		
18		

Nº	a	b
19		
20		
21		
22		
23		
24		
25		
26		
27		
28		
29		
30		
31		
32		
33		
34		
35		

Preguntas:

1. Uno de los problemas de la multiprogramación con particiones de tamaño fijo explicada en clase es que:
   - *a*) El número de particiones limita el número de procesos que pueden estar en ejecución.
   - *b*) Se ha de conocer de antemano la partición que va a ocupar cada proceso para que el compilador pueda generar las direcciones de memoria correctas.

2. De entre las políticas vistas en clase para la asignación de huecos en un esquema de administración con particiones variables:
   - *a*) La ≪Peor en ajustarse≫ suele aprovechar mejor la memoria que la ≪Mejor en ajustarse≫.
   - *b*) La ≪Mejor en ajustarse≫ es una variante de la ≪Primera en ajustarse≫, salvo que la búsqueda del hueco no se inicia siempre desde el principio sino desde el punto en el que se quedó la búsqueda anterior.

3. La técnica de ≪Compactación≫:
   - *a*) Se utiliza en el esquema de administración de memoria con particiones variables para conseguir agrupar la memoria libre fragmentada, a costa de un consumo apreciable de CPU en el proceso.
   - *b*) Se utiliza tanto en el esquema de administración de memoria con particiones fijas como variables, y tiene la ventaja de conseguir agrupar la memoria libre fragmentada disponible sin un consumo de CPU apreciable.

4. En cuanto a los esquemas de administración de la memoria libre, dado un esquema de administración de memoria con particiones variables, cuando un proceso termina, la tarea de fusionar el hueco que deja este proceso con los huecos adyacentes...
   - *a*) si se usa un mapa de bits se realiza de forma implícita.
   - *b*) si se usa una lista de procesos+huecos con los nodos ordenados por dirección, simplemente se añade el tamaño del nuevo hueco creado al nodo inmediatamente anterior en la lista.

5. Tenemos un proceso P, ejecutándose sobre un computador que emplea un esquema de administración de memoria con particiones variables. El registro base tiene el valor B y el registro límite tiene el valor L. Si P genera un dirección lógica D:
   - *a*) D será válida si es menor que $(B + L)$.
   - *b*) D será inválida si es mayor que L.

6. A la hora de resolver el problema de la protección entre procesos que surge en los esquemas de administración de memoria con particiones:
   - *a*) El sistema operativo supervisa cada acceso a memoria y abortará el proceso si la dirección de memoria del acceso no está dentro del rango permitido.
   - *b*) El hardware debe proveer necesariamente algún mecanismo, por ejemplo, registros especiales, y realizar el chequeo de las direcciones de memoria generadas por los procesos.

7. En el esquema de memoria virtual con paginación visto en clase:
   - *a*) El programador debe indicar qué rangos de memoria del mapa de memoria del proceso (datos, código y pila) deben tener asociadas páginas físicas.
   - *b*) El sistema operativo es el que decide qué partes de cada proceso se encuentran en memoria en cada momento.

8. En relación a la información que aparece en la entrada i de una tabla de páginas directa de un único nivel:
   - *a*) Cuando se produce una escritura en algún lugar de la página virtual i se activan sus bits referenciado y modificado.
   - *b*) Cuando se produce una escritura en algún lugar de la página virtual i se activa el bit modificado, pero no el bit referenciado.

9. De entre las siguientes dos situaciones, indica cuál podría darse al traducir una dirección virtual a dirección física:
   a) Acierto de TLB y Fallo de Protección.
   b) Acierto de TLB, Fallo de Página y Fallo de Protección.

10. Con respecto al TLB, es cierto que:
   a) Trata de reducir el tamaño de la tabla de páginas.
   b) Trata de reducir el número de accesos a la tabla de páginas.

11. Supón una tabla de páginas de 2 niveles, en la que la tabla de primer nivel tiene $N1$ entradas, cada una de tamaño $S1$, de las que $U1$ están siendo utilizadas. Cada una de las tablas de páginas de segundo nivel tiene $N2$ entradas, cada una de tamaño $S2$, de las que están siendo utilizadas $U2$ entradas (entre todas las tablas de segundo nivel). Con estos datos:
   a) El tamaño total de la tabla de páginas de 2 niveles es: $N1 \times S1 + U1 \times N2 \times S2$
   b) El tamaño total de la tabla de páginas de 2 niveles es: $N1 \times U1 \times S1 + N2 \times U2 \times S2$

12. Dado un tamaño de página fijo establecido, el número total de entradas, tanto libres como ocupadas, en una tabla de páginas invertida...
   a) es directamente proporcional al número de procesos que se estén ejecutando.
   b) es directamente proporcional a la cantidad de memoria física que tengamos.

13. Dado un sistema que implementa TLB y una tabla de páginas invertida, un acceso a memoria que no produce un fallo de página...:
   a) supondrá siempre un acceso al TLB.
   b) supondrá siempre un acceso a la tabla de dispersión seguido de un acceso a la tabla de traducción.

14. Dado un sistema que implementa TLB y una tabla de páginas multinivel con N niveles, si el proceso que está en ejecución realiza una operación de lectura o escritura a la dirección virtual D_V, el número total de accesos a memoria que se producirán hasta completar dicha operación es:
   a) N, si se produce fallo de TLB pero se acierta en la tabla de páginas.
   b) 1, si se produce acierto de TLB.

15. La paginación permite la protección de memoria entre procesos siempre que:
   a) Dichos procesos no compartan marcos de ninguna manera o únicamente compartan marcos de solo lectura.
   b) Desde un punto de vista lógico, cada proceso posea su propia tabla de páginas, aunque tengan páginas que compartan marcos.

16. Dado un sistema de memoria virtual:
   a) El mapa de memoria de un proceso se compone de *regiones*, que no son más que rangos de direcciones virtuales contiguas dedicadas a un mismo propósito.
   b) El mapa de memoria está compuesto de la lista de zonas del espacio de direcciones físicas que están siendo usadas.

17. En un esquema de memoria virtual con paginación, si tenemos un proceso P que realiza una llamada al sistema `fork()` para crear el proceso H:
   a) La tabla de páginas de H se construye copiando la de P, pero haciendo obligatoriamente que apunte cada página virtual a una copia del marco físico correspondiente, ya que los espacios de direcciones de ambos procesos deben permanecer aislados.
   b) La tabla de páginas de H se construye copiando la de P, pero pueden compartirse los marcos físicos de ambos procesos, porque gracias a un mecanismo denominado *copia en escritura*, permanecen aislados los respectivos espacios de direcciones.

18. En un esquema de memoria virtual con paginación como el implementado en UNIX estudiado en clase:
   a) El sistema operativo forma parte del mapa de memoria de un proceso, facilitando, entre otras cosas, la transferencia de datos en las llamadas al sistema que lo requieran.
   b) El sistema operativo no forma parte del mapa de memoria de un proceso de cara a que cualquier intento de acceso a la región de memoria ocupada por el sistema operativo produzca una violación de memoria.

19. Supongamos un sistema operativo que utiliza una política de asignación estática y un proceso que tiene asignados 3 marcos de página. El proceso ha realizado los siguientes accesos (en este orden) R_{P2}, W_{P3}, R_{P1}, tras ello ha llegado un tic de reloj y, a continuación, R_{P2} (R y W son operaciones de lectura y escritura respectivamente, mientras que P1, P2 y P3 son tres páginas virtuales del proceso). Si ahora se produce el acceso R_{P4}, éste ocasiona un fallo de página que supone el reemplazo de:
   a) La página $P2$ si se utiliza un algoritmo FIFO y la página $P3$ si se utiliza NRU.
   b) La página $P2$ si se utiliza el algoritmo FIFO de segunda oportunidad y la página $P1$ si se usa NRU.

20. ¿En qué tipo de tabla de páginas se puede encontrar en sus entradas un campo con el PID del proceso?
   a) Tabla de páginas invertida
   b) Tabla de páginas clasica multinivel

21. Gracias a la labor del demonio de paginación del sistema operativo:
   a) Una página podría escribirse en disco aunque no se haya producido ningún fallo de página.
   b) Un fallo de página podría ocasionar varias lecturas de páginas de disco (la página demandada y las páginas siguientes).

22. En cuanto a las políticas de reemplazo de páginas vistas en clase:
   a) El algoritmo de maduración no puede saber cuál de dos páginas con el mismo número de maduración tiene la referencia más antigua.
   b) La implementación de LRU con una matriz de bits es fácilmente aplicable a sistemas de memoria virtual porque al usar un bit por página ocupa muy poca memoria.

23. En cuanto a la segmentación pura estudiada en clase es cierto que:
   a) Existe una tabla de segmentos con una entrada por proceso. Cada entrada contiene un campo base y un campo límite que permite tanto reubicar como proteger cada proceso en memoria física.
   b) Existe una tabla de segmentos por proceso. Cada entrada contiene un campo base y un campo límite que permite tanto reubicar como proteger cada segmento en memoria física.

24. Supongamos un sistema de memoria virtual basado en segmentación paginada. Tenemos 3 procesos en marcha $P1$, $P2$ y $P3$; cada uno de los cuales maneja $S1$, $S2$ y $S3$ segmentos respectivamente. El número total de páginas por proceso es $N1$, $N2$ y $N3$ respectivamente y el tamaño de página es de B bytes. Entonces, es cierto que:
   a) El tamaño total de la memoria virtual de los 3 procesos, incluyendo todos sus segmentos es $(S1 \times N1 + S2 \times N2 + S3 \times N3) \times B$ bytes.
   b) El tamaño total de la memoria virtual de los 3 procesos, incluyendo todos sus segmentos, es $(N1 \times B + N2 \times B + N3 \times B)$ bytes.

25. Uno de los objetivos del software de E/S es la conversión de las transferencias asíncronas en síncronas. Entre otras cosas, eso significa que el sistema operativo debe:
   a) Devolver inmediatamente el control al proceso que ha solicitado la E/S, siendo tarea del código de usuario del proceso estar esperando a que el dispositivo indique que los datos estén disponibles y recogerlos para continuar con la ejecución.
   b) Bloquear al proceso que solicita la E/S hasta que el dispositivo proporcione los datos, en cuyo momento se desbloquea el proceso, que continuará su ejecución con los datos ya disponibles.

26. Entre las funciones del software independiente de dispositivo está el:
    a) Desbloquear al proceso que está bloqueado esperando la E/S cuando esta llega.
    b) Gestionar el tamaño de bloque que se utiliza en la E/S.

27. Según lo visto en clase, cuando una cierta parte de un sistema operativo quiere solicitarle una operación de E/S a un manejador de dispositivo, dicha parte debe usar:
    a) Una de las funciones definidas por el sistema operativo en la interfaz uniforme de manejadores de dispositivo.
    b) Directamente una de las funciones implementadas en el manejador de dispositivo, utilizando para ello el nombre que internamente tiene la función en dicho manejador.

28. Cuando se produce una interrupción indicando que ha terminado una operación de E/S, el manejador de interrupciones:
    a) Termina el procesamiento de dicha operación, desbloquea al manejador de dispositivo que lanzó la operación para que lance la siguiente (si es que hay solicitudes pendientes) y reinicia las interrupciones.
    b) Desbloquea al manejador de dispositivo que lanzó la operación y reinicia las interrupciones.

29. En un sistema multiprogramado, un sistema de *spooling* es adecuado para manejar la E/S de...
    a) aquellos dispositivos a los que se accede en exclusiva, con el fin de evitar su monopolización.
    b) aquellos dispositivos que requieren de una interacción directa del usuario, con el fin de garantizar un tiempo de respuesta lo más rápido posible.

30. Una diferencia fundamental entre los dispositivos de bloques y de caracteres es:
    a) El tamaño mínimo de transferencia de datos usado por el sistema operativo para acceder a ellos.
    b) La velocidad del dispositivo, ya que los de bloques siempre son más rápidos.

31. ¿Qué componente del tiempo de acceso depende de la velocidad de giro del disco?:
    a) Tiempo de búsqueda.
    b) Tiempo de latencia.

32. Para una cierta operación de lectura de disco, suponiendo que los datos a leer no se encuentran en ninguna caché, podría darse que:
    a) El tiempo de latencia fuera 0.
    b) El tiempo de transmisión fuera 0.

33. En cuanto a los planificadores de disco vistos en clase, dado un conjunto concreto de peticiones de disco:
    a) El algoritmo SSF es óptimo en lo que al tiempo de búsqueda se refiere.
    b) Los algoritmos SCAN y C-SCAN garantizan que el tiempo de espera para cualquier petición estará siempre acotado.

34. Contabilizar el tiempo de uso de CPU de cada proceso asignando la marca de reloj al proceso en ejecución en el momento de producirse la marca...
    a) tiene el problema de que podría no contabilizar el tiempo de uso de CPU de un proceso que se despierte, se ejecute durante muy poco tiempo y se vuelva a bloquear.
    b) permite llevar un control preciso del tiempo de CPU consumido por cada proceso a pesar de utilizar solamente un único reloj hardware.

35. Son funciones del manejador de reloj:
    a) Controlar la hora del día, controlar el tiempo de ejecución de un proceso en función del quantum asignado y gestionar las cronómetros guardianes.
    b) Seleccionar un nuevo proceso y asignarle la CPU cuando el proceso actual agote su quantum, contabilizar el uso de la CPU por parte de los procesos y gestionar las alarmas.

Respuestas del test

Nº	a	b
1	X	
2	X	
3	X	
4	X	
5		X
6		X
7		X
8	X	
9	X	
10		X
11	X	
12		X
13	X	
14		X
15	X	
16	X	
17		X
18	X	
19		X
20	X	
21	X	
22	X	
23		X
24		X
25		X
26		X
27	X	
28		X
29	X	
30	X	
31		X
32	X	
33		X
34	X	
35	X	