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	

Nº	a	b
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
36	X	
37		X
38	X	
39		X
40		X

UNIVERSIDAD DE MURCIA

Introducción a los Sistemas Operativos

Examen de teoría (Final Julio 2023)

Parte Problemas – 19 de junio de 2023

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

Instrucciones:

- Todas las respuestas deben escribirse con **bolígrafo**.
- Se valorará la **exactitud, completitud y brevedad** de todas las respuestas, que deberán ser **RAZONADAS**. No tendrán validez las respuestas no justificadas.
- Esta parte vale 3 puntos y, por tanto, representa el 30 % de la nota del examen final de teoría.

1. (1,5 puntos) Tenemos un disco duro con 16 cabezas, 4 096 cilindros y 256 sectores por pista de 512 bytes cada uno. Usando todo el disco, se ha creado un sistema de ficheros UNIX estándar con bloques lógicos de 8 KiB. Tenemos un bloque para MBR, otro bloque para superbloque y a continuación una zona de mapa de bits de nodos-i, seguida de una zona para nodos-i (de 64 bytes cada uno) de 16 384 bloques; y una zona de mapa de bits de bloques de datos seguida de la zona de bloques de datos hasta terminar el disco. El nodo-i es el estándar, con 10 entradas directas de bloque, 1 BSI, 1 BDI y 1 BTI. Usa 4 bytes para guardar un número de bloque. Con estos datos, se pide:

   a) (0,2 puntos) ¿Cuántos ficheros de cualquier tipo (regulares, directorios, enlaces, etc.) pueden crearse, como máximo, en ese sistema de ficheros?

   b) (0,6 puntos) ¿Cuál es el tamaño, en bloques lógicos, de cada una de las zonas descritas del sistema de ficheros?

   c) (0,2 puntos) ¿Cuál sería el tamaño máximo teórico, en KiB, de un fichero con la representación de nodo-i propuesta? Supón un disco infinito.

   d) (0,2 puntos) ¿Y el espacio máximo teórico, en KiB, ocupado por los metadatos del fichero anterior?

   e) (0,3 puntos) ¿Cuál sería la ruta de accesos a metadatos y datos necesarios para llegar al último byte de un fichero de tamaño 33 628 160 bytes? Indica también cuál de los bytes del bloque de datos leido es el correspondiente a dicho último byte. Un ejemplo de ruta sería el siguiente: en el BTI se accede al BDI de la entrada 3, dentro de ese BDI se accede al BSI de la entrada 16, donde estará el número de bloque, en cuyo desplazamiento 300 estará el byte deseado.

2. (1,5 puntos) Tenemos un computador que usa como mecanismo de memoria virtual la segmentación paginada. Dentro de cada segmento, la paginación se gestiona mediante una tabla de traducción de páginas directa de un solo nivel. El número de segmento tiene 16 bits, el desplazamiento dentro de cada segmento es de 32 bits. Las direcciones físicas son de 32 bits. El tamaño de página virtual es de 16 KiB. El SO ha asignado a nuestro proceso 4 marcos de memoria a partir de la dirección física 0x00010000. Los llamaremos, de momento, *marco0-marco3*.

   a) (0,2 puntos) ¿Qué número de segmentos podemos tener como máximo? ¿Qué número de entradas tiene la tabla de traducción de páginas de cada segmento? **Razona tu respuesta**.

   b) (0,7 puntos) Sabiendo que la política de reemplazo es **LRU**. Indica para cada referencia a memoria de la tabla que se muestra a continuación:

   - Los valores de Número de Página Virtual (NPV).
   - Si se produce acierto o fallo de página.
   - El contenido de los 4 marcos de página.
   - El valor de los bits de referencia y modificación (R y M) para los 4 marcos de página.

   c) (0,6 puntos) Dibuja el estado de la tabla de segmentos y de las diferentes tablas de páginas, indicando el número de marco de página (no marco0-marco3 sino el número del marco físico correspondiente), el bit de validez, y los bits R y M de este único proceso al finalizar.

Orden	Segm	Desplazamiento	L/E	NPV	A/F	Marco0 | RM	Marco 1 | RM	Marco 2 | RM	Marco 3| RM
1	1	0x00440123	E						
2	1	0x00662332	L						
3	2	0x00440434	E						
4	1	0x00441222	L						
5	0	0x00663332	L						
6	2	0x00552443	E						
TIC									
7	2	0x00440555	E						
8	1	0x00551443	E						
9	1	0x00661332	E						

Soluciones

1. A continuación se muestran las soluciones de los diferentes apartados.

   a) Se podrán crear tantos como nodos-i tengamos. Si tenemos dedicados 16 384 bloques para nodos-i, como cada uno tiene 64 bytes, podemos calcular el número de nodos-i que hay: $(16\,384 \times 8\,192)/64 = 2\,097\,152$ nodos-i. Por tanto serán $2\,097\,152$ ficheros o directorios.

   b) En primer lugar hay que determinar el tamaño del disco, para lo cual multiplicamos las cabezas, por los cilindros, por los sectores que tiene cada cilindro y por 512 bytes por sector, lo dividimos entre 1 024 y nos da los KiB del disco: $8\,388\,608$ KiB, que equivalen a 8 GiB. Dividiendo entre 8 KiB por bloque lógico, obtenemos los bloques lógicos del disco: $1\,048\,576$ bloques. El primer bloque es el MBR, el segundo el superbloque, tenemos dedicados 16 384 bloques para nodos-i, que almacenan $2\,097\,152$ nodos-i. Esto quiere decir que cada uno de esos nodos-i tiene un bit en su mapa de bits, y por tanto necesitaremos: $2\,097\,152/(8 \times 8\,192) = 32$ bloques. Esto nos lleva a que la zona de mapa de bits de bloques de datos más la zona de bloques de datos tienen que ocupar $1\,048\,576 - 1 - 1 - 32 - 16\,384 = 1\,032\,158$ bloques. Si llamamos x al número de bloques lógicos ocupados por el mapa de bits de bloques de datos, podemos plantear la siguiente ecuación:

   $$x + (8 \times 8\,192) \cdot x = x + (65\,536) \cdot x = 1\,032\,158$$

   Despejando la x, obtenemos:

   $$x = \frac{1\,032\,158}{65\,537} = 15,75 = 16$$

   Por tanto el mapa de bits de bloques de datos ocupará 16 bloques y la zona de bloques de datos tendrá un tamaño de $1\,032\,158 - 16 = 1\,032\,142$ bloques.

   c) Si los bloques lógicos son de 8 KiB y una dirección de disco ocupa 4 bytes, cada bloque indirecto podrá almacenar 8192/4=2048 direcciones. Con este dato, el tamaño máximo teórico de un fichero con esta implementación de nodos-i sería aquel que hiciera que el fichero ocupara todo su árbol de metadatos, teniendo por tanto: $10 + 2\,048 + 2\,048^2 + 2\,048^3 = 8\,594\,130\,954$ bloques. Con bloques de 8 KiB, el tamaño máximo teórico sería 68 753 047 632 KiB, es decir, algo más de 64 TB.

   d) El tamaño de los metadatos (sin contar el nodo-i), sería el que ocupen los BSI, BDI y BTI del fichero anterior. Como tenemos 1 BSI, 1 BDI y 1 BTI completamente llenos, eso nos da un total de: $1 + 1 + 2\,048 + 1 + 2\,048 + 2\,048^2 = 4\,198\,403$ bloques de metadatos, que por 8 KiB cada uno, resulta en 33 587 224 KiB.

   e) Si el fichero tiene un tamaño de 33 628 160 bytes, su último byte será el 33 628 159. Si dividimos este número entre 8 192, tendremos el número de bloque que hay que leer, es decir, el $4\,104{,}99$, o lo que es lo mismo, el $4\,105$. Ahora podemos restar los 10 bloques del nodo-i y calcular en qué BSI se encuentra el número de bloque de disco correspondiente a ese bloque del fichero: $(4\,104 - 10)/2\,048 = 1{,}999 = 1$ empezando a contar desde 0. La entrada del BSI donde se encontrará el número de bloque de datos buscado, es $4\,094 \mod 2048 = 2046$. Como el BSI 0 es el que está independiente, lo descontamos restándole 1, lo que nos indica que nos encontramos en el BSI 0 a partir del BDI, es decir, el primer BSI del BDI independiente. Con esto ya podemos ver la secuencia de accesos: en primer lugar accedemos al BDI independiente, leemos el BSI de su entrada 0 y en su entrada 2046 se encontrará el bloque de datos buscado. En el desplazamiento 8191 de dicho bloque de datos está el byte que se desea leer.

2. La solución de cada apartado se muestra a continuación:

a) Como tenemos un registro de segmento de 16 bits, podremos tener hasta $2^{16} = 65536$ segmentos como máximo. Como el desplazamiento dentro de cada segmento es de 32 bits y el tamaño de página es de 16 KiB, necesitamos 14 bits de para el offset dentro de cada página, quedándonos 18 bits para el número de página virtual. Por tanto, la tabla de páginas de cada segmento tendrá 2^{18} entradas.

b) La siguiente tabla muestra:

- Los valores de Número de Página Virtual.
- Si se produce acierto o fallo de página.
- El contenido de los 4 marcos de página.
- El valor de los bits de control (R y M) para los 4 marcos de página.

Orden	Seg	DV	L/E	NPV	A/F	Marco 4 | Bits	Marco 5 | Bits	Marco 6 | Bits	Marco 7| Bits
1	0x01	0x00440123	E	0x110	F	1/0x110 | RM			
2	0x01	0x00662332	L	0x198	F	1/0x110 | RM	1/0x198 | R-		
3	0x02	0x00440434	E	0x110	F	1/0x110 | RM	1/0x198 | R-	2/0x110 | RM	
4	0x01	0x00441222	L	0x110	A	1/0x110 | RM	1/0x198 | R-	2/0x110 | RM	
5	0x00	0x00663332	L	0x198	F	1/0x110 | RM	1/0x198 | R-	2/0x110 | RM	0/0x198 | R-
6	0x02	0x00552443	E	0x154	F	1/0x110 | RM	2/0x154 | RM	2/0x110 | RM	0/0x198 | R-
Tic									
7	0x02	0x00440055	E	0x110	A	1/0x110 | RM	2/0x154 | RM	2/0x110 | RM	0/0x198 | R-
8	0x01	0x00551443	E	0x154	F	1/0x154 | RM	2/0x154 | RM	2/0x110 | RM	0/0x198 | R-
9	0x01	0x00661332	E	0x198	F	1/0x154 | RM	2/0x154 | RM	2/0x110 | RM	1/0x198 | RM

c) Los 4 marcos se encuentran a partir de la dirección física 0x00010000. Si ponemos esa dirección en binario, tendríamos 0000 0000 0000 0001 0000 0000 0000 0000. Como el offset es de 14 bits, el número de marco físico se obtendría tomando los primeros 18 bits, es decir 0000 0000 0000 0001 00, que en decimal sería un 4. Por tanto los marcos asignados serían los marcos físicos 4, 5, 6 y 7. La tabla de segmentos y las tablas de páginas quedarían así: