SISTEMAS OPERATIVOS (2011-12)
Grupo D
Ejercicios - 1

1. ¿Cuál de las siguientes combinaciones no es factible? Justifíquelo detalladamente.
   q) Procesamiento por lotes (batch) con multiprogramación.
   c) Tiempo compartido sin multiprogramación.
   d) Multiprogramación en un sistema monousuario.

2. ¿Qué debería hacer el planificador a corto plazo cuando es invocado pero no hay ningún proceso en la cola de ejecutables?

3. ¿Qué algoritmos de planificación quedan descartados para ser utilizados en sistemas de tiempo compartido?

4. La representación gráfica del cociente *[(tiempo_en_cola_ejecutables + tiempo_de_CPU) / tiempo_de_CPU]* frente a *tiempo_de_CPU* suele mostrar valores muy altos para ráfagas muy cortas en casi todos los algoritmos de asignación de CPU. ¿Por qué?

5. Sea un sistema multiprogramado que utiliza el algoritmo Por Turnos (*Round-Robin*).
Sea **S** el tiempo que tarda el despachador en cada cambio de contexto.
¿Cuál debe ser el valor de quantum **Q** para que el porcentaje de uso de la CPU por los procesos de usuario sea del 80%?

6. Sea un sistema multiprogramado que utiliza el algoritmo Por Turnos (*Round-Robin*).
Sea **S** el tiempo que tarda el despachador en cada cambio de contexto,
y **N** el número de procesos existente.
¿Cuál debe ser el valor de quantum **Q** para que se asegure que cada proceso "ve" la CPU al menos cada **T** segundos?

7. ¿Puede el procesador manejar una interrupción mientras esta ejecutando un proceso si la política de planificación que utilizamos es no apropiativa (sin desplazamiento)?

8. Suponga que es responsable de diseñar e implementar un sistema operativo que va a utilizar una política de planificación apropiativa (con desplazamiento) y que ya tenemos desarrollado el algoritmo de planificación sin desplazamiento ¿qué partes del sistema operativo habría que modificar para implementar la modalidad apropiativa y cuáles serían tales modificaciones?

9. En el algoritmo de planificación FCFS, la **penalización** (*(t + tº de espera) / t*), ¿es creciente, decreciente o constante respecto a t (tiempo de servicio de CPU requerido por un proceso)? Justifique su respuesta.

10. En la tabla siguiente se describen cinco procesos:

Proceso	Tiempo de creación	Tiempo de CPU
A	4	1
B	0	5
C	1	4
D	8	3
E	12	2

Si suponemos que tenemos un algoritmo de planificación que utiliza una política FIFO (primero en llegar, primero en ser servido), calcula:

a) Tiempo medio de respuesta
b) Tiempo medio de espera
c) La penalización, es decir, el cociente entre el tiempo de respuesta y el tiempo de CPU.

11. Utilizando los valores de la tabla del problema anterior, calcula los tiempos medios de espera y respuesta para los siguientes algoritmos:

a) Por Turnos con quantum q=1
b) Por Turnos con quantum q=4
c) El más corto primero (SJF). Suponga que se estima una ráfaga igual a la real.

12. Calcula el tiempo de espera medio para los procesos de la tabla utilizando el algoritmo: el primero más corto apropiativo (o primero el de tiempo restante menor, SRTF).

Proceso	Tiempo de creación	Tiempo de CPU
A	0	3
B	1	1
C	3	12
D	9	5
E	12	5

13. Utilizando la tabla del ejercicio anterior, dibuja el diagrama de ocupación de CPU para el caso de un sistema que utiliza un algoritmo de colas múltiples con realimentación con las siguientes colas:

Cola	Prioridad	Quantum
1	1	1
2	2	2
3	3	4

y suponiendo que:

(a) los procesos entran en la cola de mayor prioridad (menor valor numérico). Cada cola se gestiona mediante la política Por Turnos.
(b) la política de planificación entre colas es por prioridades no apropiativo.
(c) un proceso en la cola *i* pasa a la cola *i+1* si consume un quantum completo sin bloquearse.
(d) cuando un proceso llega a la cola de menor prioridad, permanece en ella hasta que finalice.

14. Consideremos los procesos cuyo comportamiento se recoge en la tabla siguiente

Proceso	Tiempo creación	Comportamiento						
		CPU	Bloqueo	CPU	Bloqueo	CPU	Bloqueo	CPU
A	0	1	2	1	2	1	-	-
B	1	1	1	1	2	1	-	-
C	2	2	1	2	1	1	1	1
D	4	4	-	-	-	-	-	-

Dibuja el diagrama de ocupación de la CPU para los siguientes algoritmos:

a) FIFO
b) Por Turnos (Round-Robin), con q=1
c) Prioridades, suponiendo que las prioridades son 3 para A y B, 2 para C, y 1 para D (mayor número = menor prioridad).
d) Primero el más corto, suponiendo que la estimación de cada ráfaga coincide con la duración de la ráfaga anterior. La estimación para la primera ráfaga es su valor real.

SISTEMAS OPERATIVOS (2011-12)
Grupo D
Ejercicios – 2

1. Considere un sistema con un espacio lógico de memoria de 128K páginas con 8 KB cada una, una memoria física de 64 MB y direccionamiento al nivel de byte. ¿Cuántos bits hay en la dirección lógica? ¿Y en la física?

2. Sitúese en un sistema paginado con memoria virtual, en donde:
- la memoria real tiene un tamaño de 16 Mbytes
- una dirección virtual ocupa 32 bits, de los cuales los 22 de la izquierda constituyen el número de página, y los 10 de la derecha el desplazamiento dentro de la página.

Según lo anterior,
a) ¿Qué tamaño tiene cada página?
b) ¿Cuál es el tamaño del espacio de direccionamiento virtual?
c) ¿En cuántos marcos de página se divide la memoria física?
d) ¿Qué tamaño deberá tener el campo **Número de Marco** de la Tabla de Páginas?
e) Además de dicho campo, suponga que la Tabla de Páginas tiene los siguientes campos con los siguientes valores:
* Presencia: 1 bit (1= Presente en memoria fisica, 0= ausente)
* Modificación: 1 bit (1= Ha sufrido modificación desde que se cargó en memoria)
* Protección: 1 bit (1= Sólo se permite leer; 0= Cualquier tipo de acceso).
f) ¿Cuál es el tamaño de la Tabla de Páginas para un proceso cuyo espacio de memoria virtual es de 103K bytes?

3. Sea un sistema de memoria virtual paginada con direcciones lógicas de 32 bits que proporciona un espacio virtual de 220 páginas y con una memoria física de 32 Mbytes ¿cuánta memoria requiere en total un proceso que tenga 453Kbytes, incluida su tabla de páginas cuyas entradas son de 32 bits?

4. Un ordenador tiene 4 marcos de página. En la siguiente tabla se muestran: el tiempo de carga, el tiempo del último acceso y los bits R y M para cada página (los tiempos están en tics de reloj). Responda a las siguientes cuestiones justificando su respuesta.

Página	Tiempo de carga	Tiempo ultima Referencia	Bit de Referencia	Bit de Modificación
0	126	279	1	0
1	230	235	1	0
2	120	272	1	1
3	160	200	1	1

a) ¿ Qué página se sustituye si se usa el algoritmo FIFO?
b) ¿ Qué página se sustituye si se usa el algoritmo LRU?

5. ¿Depende el tamaño del conjunto de trabajo de un proceso directamente del tamaño del programa e jecutable asociado a él? Justifique su respuesta.

6. ¿Por qué una cache (o la TLB) que se accede con direcciones virtuales puede producir incoherencias y requiere que el sistema operativo la invalide en cada cambio de contexto y, en cambio, una que se accede con direcciones físicas no lo requiere?

7. Un ordenador proporciona un espacio de direccionamiento lógico (virtual) a cada proceso de 65.536 bytes de espacio dividido en páginas de 4096 bytes. Cierto programa tiene un tamaño de región de texto de 32768 bytes, un tamaño de región de datos de 16386 bytes y tamaño de región de pila de 15878. ¿Cabría este programa en el espacio de direcciones? (Una página no puede ser utilizada por regiones distintas). Si no es así, ¿cómo podríamos conseguirlo, dentro del esquema de paginación?

8. Analice qué puede ocurrir en un sistema que usa paginación por demanda si se recompila un programa mientras se está ejecutando. Proponga soluciones a los problemas que pueden surgir en esta situación.

9. Para cada uno de los siguientes campos de la tabla de páginas, se debe explicar si es la MMU o el sistema quién los lee y escribe (en éste último caso si se activa o desactiva), y en qué momentos:

a) Número de marco.
b) Bit de presencia
c) Bit de protección
d) Bit de modificación
e) Bit de referencia

10. Suponga que la tabla de páginas para el proceso actual se parece a la de la figura. Todos los números son decimales, la numeración comienza en todos los casos desde cero, y todas las direcciones de memoria son direcciones en bytes. El tamaño de página es de 1024 bytes.

Número de página virtual	Bit de validez o presencia	Bit de referencia	Bit de modificación	Número de marco de página
0	0	1	0	4
1	1	1	1	7
2	1	0	0	1
3	1	0	0	2
4	0	0	0	-
5	1	0	1	0

¿Qué direcciones físicas, si existen, corresponderán con cada una de las siguientes direcciones virtuales? (no intente manejar ninguna falta de página, si las hubiese)

a) 999
b) 2121
c) 5400

11. Sea la siguiente secuencia de números de página referenciados: 1,2,3,4,1,2,5,1,2,3,4,5

Calcula el número de faltas de página que se producen utilizando el algoritmo FIFO y considerando que el número de marcos de página de que disfruta nuestro proceso es de

a) 3 marcos
b) 4 marcos

¿Se corresponde esto con el comportamiento intuitivo de que disminuirá el número de faltas de página al aumentar el tamaño de memoria de que disfruta el proceso?

12. ¿Qué tipo de fragmentación se produce en un sistema de gestión de memoria virtual paginado? ¿Qué decisiones de diseño se pueden tomar para minimizar dicho problema, y cómo afectan estas decisiones al comportamiento del sistema?

13. Suponga que un proceso emite una dirección lógica igual a 2453 y que se utiliza la técnica de paginación, con páginas de 1024 palabras

a)Indique el par de valores (número de página, desplazamiento) que corresponde a dicha dirección.

b)¿Es posible que dicha dirección lógica se traduzca en la dirección física 9322? Razónelo.

14. El tiempo medio de ejecución de una instrucción en un procesador es de 30 nsg. Tras diversas medidas se ha comprobado que:

a) El 0.001% de las instrucciones producen falta de página.
b) El 30% de las ocasiones en que se produce la falta de página, la página que hay que sustituir está "sucia".
c) La velocidad de transferencia al dispositivo de disco es de 2MB/sg. El tamaño de cada página es de 4 KB.

Calcule el tiempo efectivo de una instrucción (el tº que tarda en ejecutarse).

15. Suponga que tenemos 3 procesos ejecutándose concurrentemente en un determinado instante y que todas sus páginas deben estar en memoria principal. El sistema operativo utiliza un sistema de memoria con paginación. Se dispone de una memoria física de 131072 bytes (128K). Sabemos que nuestros procesos al ser ejecutados tienen los siguientes parámetros:

Proceso	código	pila	datos
A	20480	14288	10240
B	16384	8200	8192
C	18432	13288	9216

Los datos indican el tamaño en bytes de cada uno de los segmentos que forman parte de la imagen del proceso. Sabiendo que una página no puede contener partes de dos segmentos diferentes (pila, código datos), hemos de determinar el tamaño de página que debería utilizar nuestro sistema y se barajan dos opciones: páginas de 4096 bytes (4K) o páginas de 512 bytes (1/2K). Se pide:

o

a) ¿Cuál sería la opción más apropiada, 4096 bytes o 512 bytes?. Justifica totalmente la respuesta mostrando todos los cálculos que has necesitado para llegar a dicha conclusión.
b) ¿Cuál es el formato de cada entrada de la Tabla de Páginas con el tamaño de página elegido? Justifica el tamaño de los campos con direcciones. Puedes añadir los bits que consideres necesarios para el buen funcionamiento del sistema indicando para que van a ser utilizados.
c) ¿Cuántas Tablas de Páginas habrá en este sistema?¿Cuántas entradas hay en cada tabla de páginas (filas)?

16. En la gestión de memoria en un sistema paginado, ¿qué estructura/s de datos necesitará mantener el Sistema Operativo para administrar el espacio libre?

17. Situándonos en un sistema paginado, donde cada proceso tiene asignado un número fijo de marcos de páginas. Supongamos la siguiente situación: existe un proceso con 7 páginas y tiene asignados 5 marcos de página. Indica el contenido de la memoria después de cada referencia a una página si como algoritmo de sustitución de página utilizamos el LRU (la página no referenciada hace más tiempo). La secuencia de referencias es la indicada en la figura.

Referencias	2	1	3	4	1	5	6	4	5	7	4	2
Marcos de página												

¿Cuantas faltas de página se producen? _____

18. Supongamos que tenemos un proceso ejecutándose en un sistema paginado, con gestión de memoria basada en el algoritmo de sustitución **frecuencia de faltas de página**. El proceso tiene 5 páginas (0, 1, 2, 3, 4). Represente el contenido de la memoria real para ese proceso (es decir, indique que páginas tiene cargadas en cada momento) y cuándo se produce una falta de página. Suponga que, inicialmente, está cargada la página 2, el resto de páginas están en memoria secundaria y que no hay restricciones en cuanto al número de marcos de página disponibles. La cadena de referencias a página es: 0 3 1 1 1 3 4 4 2 2 4 0 0 0 0 3 y el parámetro es τ=3.

19. Describa el funcionamiento del algoritmo de sustitución basado en la **frecuencia de faltas de página**, con los siguientes datos: 4 marcos de página, en t = 0 la memoria contiene a la página 2. El tamaño de la ventana es τ= 3 y se produce la secuencia de referencias de páginas, 1 4 2 2 2 4 5 5 3 3 5 1 1 1 1 4

2															

20. Describa el funcionamiento del algoritmo de sustitución global basado en el **algoritmo basado en el modelo del conjunto de trabajo**, con los siguientes datos: 4 marcos de página, en t= 0 la memoria contiene a la página 2 que se referenció en dicho instante de tiempo. El tamaño de la ventana es τ= 3 y se produce la secuencia de referencias de páginas, 1 4 4 4 2 4 1 1 3 3 5 5 5 5 1 4

2															

21. Una computadora con memoria virtual paginada tiene un bit U por página virtual, que se pone automáticamente a 1 cuando se realiza un acceso a la página. Existe una instrucción ***limpiar_U (dir_base_tabla)*** que permite poner a 0 el conjunto de los bits U de todas las páginas de la tabla de páginas cuya dirección de comienzo pasamos como argumento. Explica cómo puede utilizarse este mecanismo para la implementación de un algoritmo de sustitución basado en el modelo del conjunto de trabajo.

22. Un Sistema Operativo con memoria virtual paginada tiene el mecanismo ***fijar_página(np)*** cuyo efecto es proteger contra la sustitución al marco de página en que se ubica la página virtual **np**. El mecanismo ***des_fijar (np)*** suprime esta protección.

a) ¿Qué estructura/s de datos son necesarias para la realización de estos mecanismos?
b) ¿En qué caso puede ser de utilidad estas primitivas?
c) ¿Qué riesgos presentan y qué restricciones deben aportarse a su empleo?

23. Implemente la política de sustitución global basada en la medida de la tasa de faltas de página de un proceso; es decir, dé respuestas a las siguientes cuestiones:

a) ¿Qué parte del Sistema Operativo deberá tomar parte?
b) ¿Cuándo entra en ejecución dicho módulo del S.O.?
c) ¿Qué estructuras de datos será necesario mantener?
d) ¿Qué decisiones podría adoptar?

24. Disponemos de un ordenador que cuenta con las siguientes características: tiene una memoria RAM de 4KBytes, permite usar memoria virtual paginada, las páginas son de 1KBytes de tamaño y las direcciones virtuales son de 16 bits. El primer marco de página (marco 0) se usa únicamente por el Kernel y los demás marcos están disponibles para su uso por los procesos que se ejecutan en el sistema. Supongamos que tenemos sólo dos procesos, P1 y P2, y que utilizan las siguientes direcciones de memoria virtual y en el siguiente orden:

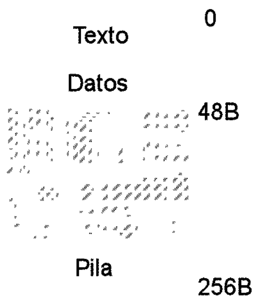
Proceso	**Direcciones virtuales**
P1	0-99
P2	0-500
P1	100-500
P2	501-1500
P1	3500-3700
P2	1501-2100
P1	501-600

a) ¿Cuántos marcos de página tiene la memoria RAM de este ordenador?
b) ¿Cuántos bits necesitamos para identificar los marcos de página?
c) Describe los fallos de página que tendrán lugar para cada intervalo de ejecución de los procesos, si la política de sustitución de páginas utilizada es LRU. Suponga que se dicho algoritmo es de asignación variable y sustitución global.

25. Estamos trabajando con un sistema operativo que emplea una gestión de memoria paginada sin memoria virtual. Cada página tiene un tamaño de 2.048 bytes. La memoria física disponible para los procesos es de 8 MBytes. Suponga que primero llega un proceso que necesita 31.566 posiciones de memoria (o bytes) y, después, llega otro proceso que consume 18.432 posiciones cuando se carga en memoria. Se pide:

a) ¿Qué fragmentación interna provoca cada proceso?
b) ¿Qué fragmentación externa provoca cada proceso?

26. Suponga un sistema que utiliza paginación a dos niveles. Las direcciones son de 8 bits con la siguiente estructura: 2 bits en la tabla de páginas de primer nivel, 2 bits en la tabla de páginas de segundo nivel y 4 bits para el desplazamiento). El espacio de direccionamiento virtual de un proceso tiene la estructura del dibujo. Represente gráficamente las tablas de páginas y sus contenidos, suponiendo que cada entrada de la tabla de páginas ocupa 8 bits y que todas las páginas están cargadas en memoria principal (elige tú mismo la ubicación en memoria principal de dichas páginas,


0
Texto
Datos
48B
Pila
256B

SISTEMAS OPERATIVOS (2011-12)
Grupo D
Ejercicios – 3

(Nota: Se han incluido algunas soluciones o comentarios sobre su resolución).

1. Sea un Sistema Operativo que sólo soporta un directorio (es decir, todos los archivos existentes estarán al mismo nivel), pero permite que los nombres de archivo sean de longitud variable. Apoyándonos únicamente en los servicios proporcionados por este Sistema Operativo, deseamos construir una "utilidad" que "simule" un sistema jerárquico de archivos. ¿Es esto posible? ¿Cómo?

A continuación representamos dos cajas: la capa software que implementa la estructura jerárquica sobre la caja que representa al sistema operativo original.

capa de software que implementa estructura jerárquica de directorios

<n1>.<n2> : significa que n2 es hijo de n1

Crear directorio n1: solo crear archivo n1

Establecer/ver directorio de trabajo: guardar/ver variable directorio-de-trabajo

Cualquier alusión <lit1> que no comience por / será convertido en directorio-de-trabajo.<lit1>

Si comienza por / se le quita / y se deja tal cual

Ver hijos de n1: ver archivos que comiencen por n1. y extraer el literal hasta el siguiente .

S.O. que solo soporta un directorio:

Ver archivos existentes
Crear, borrar archivo
Copia archivo en otro
Renombrar archivo

2. En la siguiente figura se representa una tabla FAT. Al borde de sus entradas se ha escrito, como ayuda de referencia, el número correspondiente al bloque en cuestión. También se ha representado la entrada de cierto directorio. Como simplificación del ejemplo, suponemos que en cada entrada del directorio se almacena: Nombre de archivo/directorio, el tipo (F=archivo, D=directorio), la fecha de creación y el número del bloque inicial.

Tenga en cuenta que:
- el tamaño de bloque es de 512 bytes
- el asterisco indica último bloque
- todo lo que está en blanco en la figura está libre.

Rellene la figura para representar lo siguiente:

a) Creación del archivo DATOS1 con fecha 1-3-90, y tamaño de 10 bytes.
b) Creación del archivo DATOS2 con fecha 2-3-90, y tamaño 1200 bytes.
c) El archivo DATOS aumenta de tamaño, necesitando 2 bloques más.
d) Creación del directorio D, con fecha 3-3-90, y tamaño 1 bloque.
e) Creación del archivo CARTAS con fecha 13-3-90 y tamaño 2 kBytes.

Nombre	tipo	fecha	nº bloque
DATOS	F	08/02/90	3
DATOS1	F	01/03/90	0
DATOS2	F	02/03/90	1
D	D	03/03/90	5
CARTAS	F	13/03/90	6

FAT

0	*	10	
1	2	11	
2	4	12	
3	15	13	
4	*	14	
5	*	15	16
6	7	16	17
7	8	17	*
8	9	18	
9	*	19	

3. Si se pierde el primer puntero de la lista de espacio libre, ¿podría el Sistema Operativo reconstruirla? ¿Cómo?

Recorriendo todos los archivos del árbol de directorios conocemos los bloques ocupados.

4. El espacio libre en disco puede ser implementado usando una lista encadenada con agrupación o un mapa de bits. La dirección en disco requiere D bits. Sea un disco con B bloques, en que F están libres. ¿En qué condición la lista usa menos espacio que el mapa de bits?

D*F<B

Tamaño Lista = D * F bits
Tamaño Mapa = B bits

El Sistema Operativo borra el archivo como parte de las acciones de la llamada al sistema para cerrar el archivo.

5. Algunos SO proporcionan una llamada al sistema (RENAME) para dar un nombre nuevo a un archivo existente ¿Existe alguna diferencia entre utilizar esta llamada para renombrar un archivo y copiar el archivo a uno nuevo, con el nuevo nombre y destruyendo el antiguo?

Renombrar un archivo existente: escribir otro nombre en su entrada del directorio.

Crear un archivo nuevo involucra:
* buscar bloques libres y copiar en ellos el contenido del archivo antiguo
* crear una entrada nueva para el archivo nuevo
* borrar la entrada antigua.

6. Un i-nodo de UNIX tiene 10 direcciones de disco para los diez primeros bloques de datos, y tres direcciones más para realizar una indexación a uno, dos y tres niveles. Si cada bloque índice tiene 256 direcciones de bloques de disco, cuál es el tamaño del mayor archivo que puede ser manejado, suponiendo que 1 bloque de disco es de 1KByte?

Diez direcciones de bloques de datos
---> Capacidad de direccionamiento = 10* 1KB = **10KB**

Indexación a 1 nivel
---> Capacidad de direccionamiento =256 dir.de bloques * 1kB =
= **256KB**

Indexación a 2 niveles
---> Capacidad de direccionamiento = 256 dir.de bloques * 256kB que se pueden direccionar a 1 nivel = **65536KB**

Indexación a 3 niveles
---> Capacidad de direccionamiento = 256 dir.de bloques * 65536KB que se pueden direccionar a 2 nivel = **16777216KB**

Total: **10KB + 256KB + 65536KB** +**16777216KB = 17.247.250.432 Bytes**

7. Sobre conversión de direcciones lógicas dentro de un archivo a direcciones físicas de disco. Estamos utilizando la estrategia de indexación a tres niveles para asignar espacio en disco. Tenemos que el tamaño de bloque es igual a 512 bytes, y el tamaño de puntero es de 4 bytes. Se recibe la solicitud por parte de un proceso de usuario de leer el carácter número N de determinado archivo. Suponemos que ya hemos leído la entrada del directorio asociada a este archivo, es decir, tenemos en memoria los datos PRIMER-BLOQUE y TAMAÑO. Calcule la sucesión de direcciones de bloque que se leen hasta llegar al bloque de datos que posee el citado carácter.

a 3er nivel, 1 Bloque de índices contiene 128 punteros, con lo que temos una capacidad de almacenamiento de **64KB**

a 2º nivel, 1 bloque de índices contiene 128 punteros a bloques de 3er nivel, con lo que tenemos una capacidad de almacenamiento de 128 veces lo anterior: **8192 KB**
Cada puntero direcciona **64KB**

a 1er nivel, cada puntero direcciona **8192 KB**

NOS DIRIGIMOS AL BYTE NUMERO N,

A primer nivel ¿qué puntero elegimos? N/8192K-> C1, R1
Número de puntero = C1

Buscamos el byte numero R1
A segundo nivel, ¿qué puntero elegimos? R1/64K ->C2, R2
Número de puntero = C2
Buscamos el byte numero R2

A tercer nivel, ¿qué puntero elegimos? R2/512-> C3, R3
Número de puntero = C3
Desplazamiento dentro de ese bloque= R3

Bloque cuya dirección está contenida en el puntero numero C3, Desplazamiento = R3

MEMORIA REAL:

nº marco:	
0	Codigo
1	Datos 1
2	Datos 2
3	Pila 1
4	Pila 2
5	Pag 0
6	Pag 1
	Pag 2
	Pag 3
	.....
	.....
	Pag 14
	Pag 15
	.....

TABLA DE PAG.

		Desde	Hasta
0	0	0	15
1	1	16	31
2	2	32	47
3	No presente	48	63
4	No presente	64	79
5	No presente	80	95
....	.....	.....	.....
10	No presente	160	175
11	No presente	176	191
12	No presente	192	207
13	No presente	208	223
14	3	224	239
15	4	240	255

TRADUCCÓN DE LA DIRECCIÓN VIRTUAL 46=

0 0 1 0	1 1 1 0
p	Desplazamiento

Buscamos en la TP la entrada numero p=(10)base2 =2
Su valor es 2, éste es el número de marco.
Por tanto la dirección real es

..... 1 0	1 1 1 0
N.Marco	Desplazamiento

TRADUCCÓN DE LA DIRECCIÓN VIRTUAL 210= 11010010

1 1 0 1	0 0 1 0
p	Desplazamiento

Buscamos en la TP la entrada numero p=(1101)base2 =13
Su valor es "No Presente", por tanto generamos una falta de página

B) PROBLEMA DE MEMORIA CON Paginación a 2 niveles.

****

PROBLEMA DE MEMORIA CON PAGINACION A DOS NIVELES

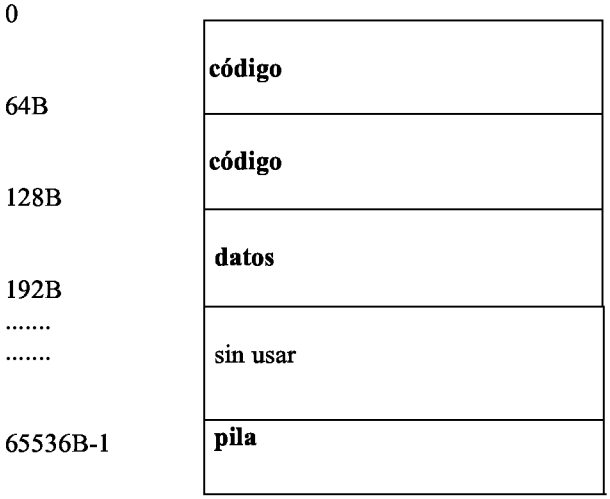
Suponga un sistema de gestión de memoria que utiliza paginación a dos niveles y que direcciona bytes. Las direcciones son de 16 bits y tienen la siguiente estructura: 5 bits para paginación a primer y segundo nivel respectivamente y 6 bits para desplazamiento. El espacio de direcciones virtuales de un proceso está constituido por: 128 bytes de código, 64 bytes de datos, la pila ocupa 64 bytes y está situada al final del espacio de direcciones, lo que deja un heap (espacio entre los segmentos de datos y pila para su crecimiento) de 65280 bytes. Representa gráficamente las tablas de páginas de primer y segundo nivel, suponiendo que cada entrada de las tablas de páginas ocupa 16 bits (para las entradas vacías, simplemente se puede indicar cuantas hay y su ubicación), e indica sus contenidos suponiendo que todas las páginas del proceso están cargadas en memoria principal a partir de la dirección 0 y son contiguas (en el orden código, datos y pila). Dada esta asignación de marcos a páginas, indica qué direcciones físicas se corresponden con las direcciones virtuales 222 y 65530.

****

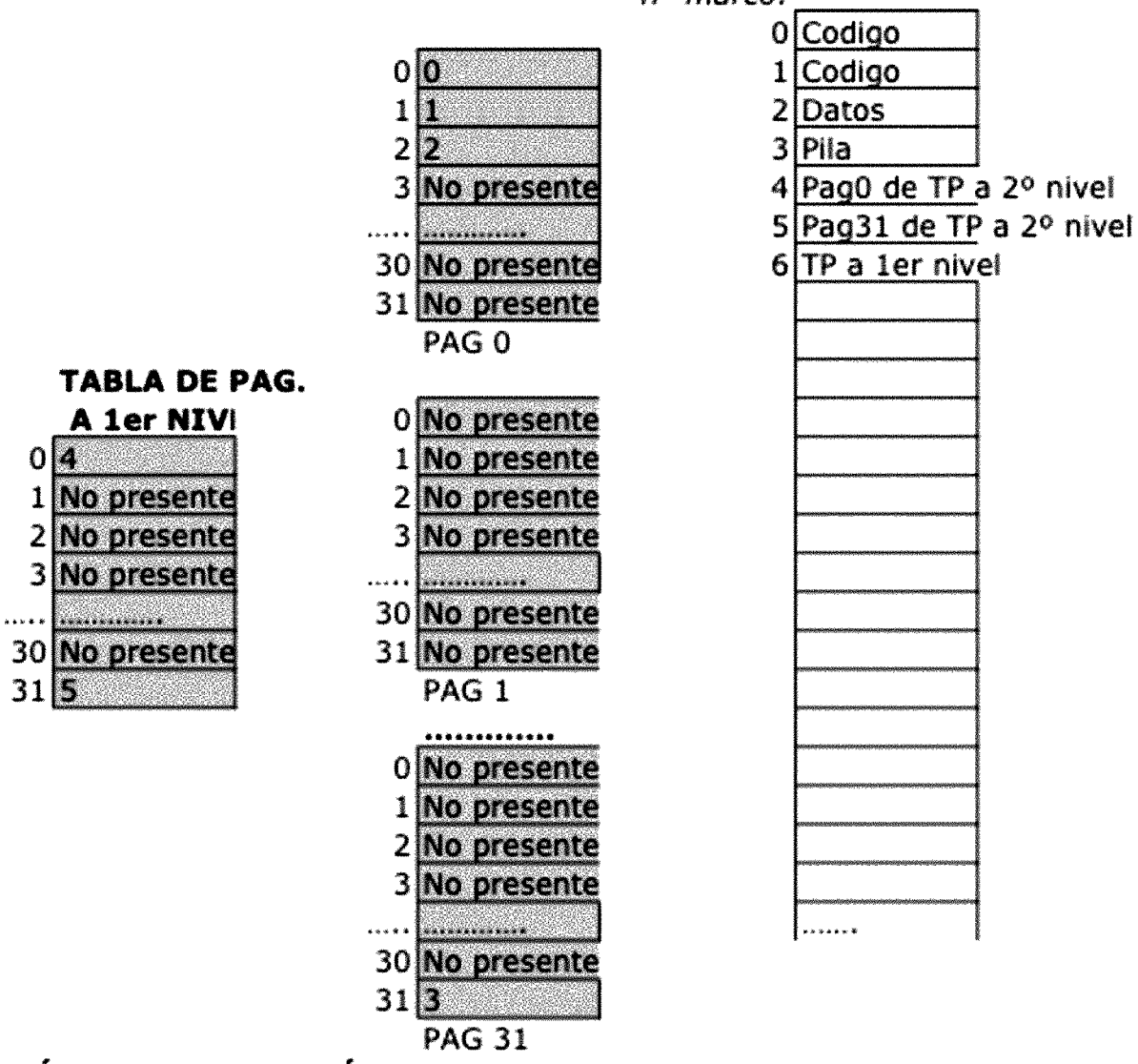
SOLUCION

1 dirección = 16 bits => hay 2**16 = 65536 direcciones
desplazamiento = 6 bits => tamaño de página = 2**6 = 64 B
1 entrada de TP ocupa 16 bits => en 1 pag caben 64B/2B= 32 entradas

ESPACIO DE DIRECCIONAMIENTO VIRTUAL DEL PROCESO



TABLAS DE PAG. A 2º NIVEL: **MEMORIA REAL:**



TRADUCCÓN DE LA DIRECCIÓN VIRTUAL 222=

0 0 0 0 0	0 0 0 1 1	0 1 1 1 1 0
p1	p2	Desplazamiento

Buscamos en la TP a 1er Nivel la entrada numero p1=0.
Su valor es 4, hemos de leer el marco numero 4 de la MP
Ahí está la Pag0 de TP a 2º nivel, nos dirigimos a ella.
Buscamos en la Pag0 de TP a 2º nivel la entrada numero p2 =(0 0 0 1 1)base2
Su valor es "No presente" por tanto generamos una falta de página

TRADUCCÓN DE LA DIRECCIÓN VIRTUAL 65530=

1 1 1 1 1	1 1 1 1 1	1 1 1 0 1 0
p1	p2	Desplazamiento

Buscamos en la TP a 1er Nivel la entrada numero p1=(1 1 1 1 1)base2 = 31
Su valor es 5, hemos de leer el marco numero 5 de la MP
Ahí está la Pag31 de TP a 2º nivel, nos dirigimos a ella.
Buscamos en la Pag31 de TP a 2º nivel la entrada numero p2 =(1 1 1 1 1)base2 = 31
Su valor es 3 = (0 0 0 1 1)base2: éste es el número de marco
Por tanto la dirección real es

0 0 0 1 1	1 1 1 0 1 0
N.Marco	Desplazamiento

C) REFLEXIONES VARIADAS SOBRE ASPECTOS DE IMPLEMENTACION DE UNIX

(Nota: Se han incluido algunas soluciones o comentarios sobre su resolución).

1. Cuando se abre el archivo /usr/ast/work/f, se necesitan varios accesos a disco. Calcule el número de accesos a disco requeridos (como máximo) bajo la suposición de que el i-nodo raíz ya se encuentra en memoria y que todos los directorios necesitan como máximo 1 bloque para almacenar los datos de sus archivos.

2. ¿El planificador de Unix favorece a los procesos limitados por E/S (cortos) frente a los procesos limitados por CPU (largos)? Explique cómo lo hace.

3. Supongamos que un proceso, **P1,** abre el archivo "**datos**" en modo **lectura/escritura** y otro proceso, **P2,** abre el mismo archivo y con el mismo modo, y a continuación crea un proceso hijo que abre el archivo "**/usr/pepe/doc**" en modo **lectura/escritura**. Represente toda la información relevante sobre el estado de las tablas de descriptores de archivos, tabla de archivos y tabla de i-nodos después de dichas operaciones.

4. Supongamos que se permite que dos o más búferes de la caché puedan contener el mismo bloque de disco simultáneamente ¿Qué problemas podrían presentarse? Ponga un ejemplo ¿Cómo lo resuelve el sistema?

5. Suponiendo una ejecución correcta de las siguientes órdenes en el sistema operativo Unix:

```
/home/jgarcia/prog > ls  -i        (* lista los archivos y sus números de i-nodos del directorio prog*)
      18020                fich1.c
      18071                fich2.c
      18001                pract1.c

/home/jgarcia/prog > cd ../tmp
/home/jgarcia/tmp > ln -s ../prog/pract1.c p1.c        (* crea un enlace simbólico *)
/home/jgarcia/tmp > ln ../prog/pract1.c p2.c           (* crea un enlace absoluto o duro*)
```

   represente gráficamente cómo y dónde quedaría reflejada y almacenada **toda** la información referente a la creación anterior de un enlace simbólico y absoluto ("hard") a un mismo archivo, **pract1.c**.

6. ¿Cuál es la razón de que el sistema Unix mantenga una ***Tabla de Regiones*** en vez de almacenar toda la información acerca de las regiones de un proceso en su ***Tabla de Regiones Por Proceso***?

Para poder compartir regiones

7. ¿Por qué se almacena la **prioridad** de un proceso en su entrada correspondiente de la ***Tabla de Procesos*** y no en su ***u-área***?

[Bach 2.2.2] Internamente e kernel referencia la variable estructura u para acceder a la u-area del proceso actual, y cuando otro proceso se ejecute, el kernel reasigna su espacio de direcciones virtuales de modo que la estructura u se refiera a la u-area del nuevo proceso. Esta implementación da al kernel una forma facil de acceder a los datos que se almacenan en la u-area (datos del proceso actual).

[Bach 6.3] Contexto a nivel de sistema: un componente de la parte estática el contexto a nivel de sistema es la u-area; contiene información de control del proceso que el kernel necesita tener únicamente cuando se ejecuta en el contexto de este proceso. Los

parámetros de control generales como la prioridad sol almacenados en la Tabla de Procesos porque debe ser accedidos fuera del contexto de este proceso.

[Stalling Esta distinción entre la Tabla de Procesos y la u-area refleja el hecho de que Unix siempre se ejecuta en el contexto de algún proceso, y gran parte del tiempo hará cosas alusivas a este proceso. Pero cuando tenga que hacer gestiones generales a todos los procesos como decisiones de elección de un proceso, necesitará acceder a informaciones de procesos que no son el actual, para ello usará la Tabla de Páginas que contiene la información de control accesible para el núcleo en todo momento, y todas sus entradas se mantienen en memoria principal.

8. Supongamos que un proceso va a ejecutar la llamada al sistema ***open*** la cual, a su vez, utiliza el algoritmo de más bajo nivel ***namei*** ¿Qué ocurre en el contexto del proceso a nivel de sistema antes, durante y después de la ejecución de la llamada (céntrese en las estructuras de datos relacionadas con dicho contexto)?

9. Respecto a la gestión de memoria: suponga que un proceso realiza una llamada al sistema ***fork*** creando un proceso hijo. Represente gráficamente como quedan las estructuras de datos relacionadas con ambos procesos (tablas de regiones por proceso, tabla de regiones, tablas de páginas y tabla de marcos de página).

10. ¿Cuál es el problema que se plantea cuando un proceso no realiza la llamada al sistema ***wait*** para cada uno de sus procesos hijos que han terminado su ejecución? ¿Qué efecto puede producir esto en el sistema?

11. ¿Qué información comparten un proceso y su hijo después de ejecutar el siguiente código? Justifique su respuesta e indique qué hace este trozo de código.

```
if   ( fork() != 0 )
       wait (&status)
else
       exec (B);
```

// usamos una llamada al sistema exec genérica con un único argumento, el nombre de un archivo ejecutable

12. Represente con un esquema gráfico la información almacenada en las estructuras (tablas de descriptores de archivos, tabla de archivos y tabla de i-nodos) que representan el estado real referente a los archivos utilizados por los siguientes procesos en el instante de tiempo en que: el *PROCESO HIJO DE A* ha ejecutado la instrucción marcada con *(1),* el *PADRE (PROCESO A)* ha ejecutado la *(2)*, y el *PROCESO B* ha ejecutado la *(3)*. Para ello suponga que son los únicos procesos ejecutándose en el sistema en ese momento.

```
/* PROCESO A */
main(){
int archivos[2], pid;
archivos[0]=open("/users/prog.c",O_RDONLY);
archivos[1]=open("/users/prog.c",O_RDWR);
if ((pid=fork())==-1)
        perror("Ha ocurrido un error en la creación del proceso);
else if (pid==0)
```

```
        execlp ("ls", "ls", NULL); /*(1)*/
else
close(archivos[0]); /*(2)*/
}


/* PROCESO B */
main(){
int archivos[2];
archivos[0]=open("/users/prog.c",O_RDONLY);
archivos[1]=open("/users/texto.txt",O_RDWR);
dup (archivos[0]);      /* (3) */
}
```

13. Para comprobar la consistencia de un sistema de archivos de Unix, el comprobador de consistencia (*fsck*) construye dos listas de contadores (cada contador mantiene información de un bloque de disco), la primera lista registra si el bloque está asignado a algún archivo y la segunda, si está libre. Según se muestra en la siguiente figura:

en uso:	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0
libres:	0	0	0	1	1	1	0	0	0	1	0	1	1	0	1

¿Existen errores? ¿Son serios estos errores? ¿por qué? ¿qué acciones correctivas sería necesario realizar sobre la información del sistema de archivos?

Hay incoherencia en aquellos casos en que un bloque está calificado como en uso y como libres simultáneamente. Es un error serio, pues podria provocar un deterioro en la información del bloque si se usa como libre estando ocupado. Su solución es fácil: recorrer el sistema de archivos comprobando si este bloque es aludido en alguno de los archivos.

14.Suponga el siguiente trozo de código ejecutado por un proceso:

```
int pid, fd;
...
fd = open ("arch_temporal", O_CREAT | O_RDWR, S_IRWXU);
unlink("arch_temporal");
pid = fork();
...
```

a) Dibuje las estructuras de datos del kernel respecto al Sistema de Archivos junto con sus principales contenidos.
b) Explique detalladamente qué ocurriría si, ahora, el proceso hijo intenta ejecutar una llamada al sistema **read** sobre el descriptor de archivo **fd**.

15.Suponga la siguiente instrucción: *ls | wc –l* y describa qué llamadas al sistema debería usar el shell para llevarla a cabo. Explique para qué usa cada una de ellas y relaciónelas con las estructuras de datos del Kernel.

Resuelto en prácticas en el ejemplo en que se implementa ls | sort