

UNIVERSIDAD DE BUENOS AIRES - FACULTAD DE INGENIERÍA
75.08 / 95.03 - SISTEMAS OPERATIVOS - CÁTEDRA MÉNDEZ
1er PARCIAL

Nombre:	José Ignacio Castro Martinez				Nota:
Padron:	106957	Cuatrimestre:	1/2024	Fecha:	

GRILLA DE RESPUESTAS A PREGUNTAS

Responda las preguntas multiple choice utilizando la grilla de respuestas marcando con una X la respuesta correcta.
Las preguntas tienen una única respuesta correcta

Resolver los ejercicios en una hoja aparte.

	1	2	3	4	5	6	7	8	9	10
A										
B				X	X			X		X
C	X	X	X				X			
D										

1 - Pregunta: ¿Qué estructura de datos utiliza el CFS (Completely Fair Scheduler de Linux) para mantener el orden de ejecución de las tareas?

- a) Cola FIFO
- b) Árbol AVL (Adelson-Velsky and Landis)
- c) Árbol Rojo-Negro
- d) Lista doblemente enlazada

2 - Pregunta: ¿Qué estructura de datos en el VFS de Linux representa un archivo abierto?

- a) inode
- b) dentry
- c) file
- d) superblock

3 - Pregunta: En una implementación típica de malloc, ¿qué estructura de datos se utiliza comúnmente para administrar los bloques de memoria libre?

- a) Árbol AVL.
- b) Tabla hash.
- c) Lista enlazada.
- d) Pila (stack).

4 - Pregunta: Durante el proceso de ejecución de una llamada al sistema (system call) en un sistema operativo, ¿cuál es el orden correcto de los siguientes eventos y cuál es su propósito principal?

1. Cambio de modo de usuario a modo kernel.
2. Ejecución del manejador de la llamada al sistema en el kernel.
3. Validación de los parámetros de la llamada al sistema.
4. Retorno al modo de usuario con el resultado de la llamada al sistema.

- a) 1 -> 3 -> 2 -> 4; El propósito principal es asegurar que los parámetros sean válidos antes de ejecutar la operación del kernel.
- b) 1 -> 2 -> 3 -> 4; El propósito principal es permitir que el kernel ejecute la operación antes de verificar la validez de los parámetros
- c) 3 -> 1 -> 2 -> 4; El propósito principal es validar los parámetros en modo usuario antes de cambiar al modo kernel.
- d) 1 -> 2 -> 4 -> 3; El propósito principal es ejecutar la operación del kernel lo más rápido posible y validar los parámetros después.

5 - Pregunta: ¿Cuál de las siguientes afirmaciones describe mejor una condición de carrera (race condition) en el contexto de la programación concurrente?

- a) Ocurre cuando dos procesos intentan acceder a una variable global, pero el sistema operativo serializa los accesos para evitar problemas.
- b) Es una situación en la que el resultado del programa depende del orden no controlado de la ejecución de hilos o procesos.
- c) Se refiere a una técnica de optimización donde múltiples hilos compiten por el uso de la CPU para mejorar el rendimiento del sistema.
- d) Es un problema que solo ocurre en sistemas de tiempo real debido a la necesidad de cumplir con estrictos plazos de tiempo.

6 - Ejercicio: Describir en un esquema y detallar los componentes de un vsfs en un disco de 8192 bloques, cada bloque posee una longitud de 4096 bytes. Los inodos del disco tienen un tamaño de 128 bytes.

7 - Pregunta: ¿cuál de las siguientes afirmaciones describe mejor la diferencia entre un sistema operativo monolítico y un sistema operativo microkernel?

- a) Los sistemas monolíticos dividen el kernel en múltiples servicios pequeños, mientras que los microkernels tienen un kernel único grande.
- b) Los sistemas monolíticos implementan la mayoría de los servicios del sistema operativo en el espacio de usuario, mientras que los microkernels los implementan en el espacio del kernel.
- c) Los sistemas monolíticos ejecutan todos los componentes del sistema operativo en el espacio del kernel, mientras que los microkernels ejecutan la mayoría de los servicios del sistema operativo en el espacio de usuario.
- d) Los sistemas monolíticos son más adecuados para dispositivos embebidos, mientras que los microkernels son más adecuados para sistemas de escritorio y servidores.

8 - Pregunta: ¿Cuál de las siguientes afirmaciones describe mejor el propósito y funcionamiento de la Translation Lookaside Buffer (TLB) en un sistema de memoria virtual?

- a) La TLB almacena copias de datos de la memoria principal para reducir los tiempos de acceso, similar a una caché de CPU.
- b) La TLB es una caché especial en la unidad de gestión de memoria (MMU) que almacena las traducciones de direcciones virtuales a físicas más frecuentemente utilizadas, reduciendo la necesidad de acceder a las tablas de páginas en la memoria principal.
- c) La TLB gestiona la distribución de páginas de memoria entre múltiples procesos, asegurando que cada proceso reciba una cantidad justa de memoria física.
- d) La TLB es una tabla en la memoria principal que almacena información sobre la correspondencia entre páginas (de direcciones virtuales) y frames (de direcciones físicas).

9 - Ejercicio: Dado un espacio de direcciones virtuales con direcciones de 8 bits y páginas de 16 bytes, asume un array de 12 enteros (cada uno de 4 bytes) comenzando en la dirección virtual 100. Calcula el patrón de aciertos y fallos en la TLB cuando se accede a todos los elementos del array en un bucle. Asume que inicialmente, la TLB está vacía.

Pasos:

- Determina el VPN y el desplazamiento para cada elemento del array.
- Identifica si ocurre un acierto o un fallo en la TLB para cada acceso.

10 - Pregunta: En el contexto de la política de scheduling Multi-Level Feedback Queue (MLFQ) en sistemas operativos, ¿cuál de las siguientes afirmaciones describe correctamente cómo funciona MLFQ y cuál es una de sus ventajas principales?

- a) MLFQ asigna a cada proceso un nivel de prioridad fijo y lo ejecuta en ese nivel hasta que termina, proporcionando tiempos de respuesta predecibles para todos los procesos.
- b) MLFQ utiliza múltiples colas con diferentes niveles de prioridad y permite que los procesos se muevan entre las colas basándose en su comportamiento y tiempo de ejecución, adaptándose dinámicamente a las necesidades de los procesos.
- c) MLFQ asigna un quantum de tiempo fijo a cada proceso y rota los procesos en orden circular, proporcionando equidad en el tiempo de CPU.
- d) MLFQ asigna siempre el quantum de tiempo más largo disponible a los procesos interactivos, asegurando que estos procesos reciban más tiempo de CPU que los procesos en segundo plano.