# SISTEMAS OPERATIVOS: PREGUNTAS DE FINAL

## Comunicación entre Procesos con Pipes

1. Explique diferencias respecto de mecanismos de comunicación basados en memoria compartida.
2. ¿Qué es un pipe? ¿Cómo se utiliza (desde un punto de vista de programación)? Explique su funcionamiento.
3. Compare con memoria compartida y pasaje de mensajes.
4. Explique comunicación entre procesos utilizando pipes.
5. Detalle cómo se tratan los problemas de sincronización con estos.
6. Describa y explique su funcionamiento desde el punto de vista de las aplicaciones y del S.O.
7. Detalle cómo se tratan los problemas de sincronización con este tipo de mecanismo de comunicación (desde el punto de vista de las aplicaciones y del S.O.).

## Procesos/Threads

1. Gracias a la multiprogramación es posible que dos procesos se completen en un tiempo menor que si se ejecutaran de forma secuencial, incluso en una computadora con un solo procesador. Explique por qué esto es posible, incluso si el procesador sólo puede ejecutar una instrucción en cada instante.
2. ¿Puede ocurrir lo mismo que en la pregunta anterior pero con threads en lugar de procesos? Justifique.
3. Generalmente los S.O. no hacen nada para evitar ni prevenir la aparición de deadlocks. ¿Se puede hacer algo para que las aplicaciones multi-thread que uno escribe no posean deadlocks (sin modificar el núcleo del S.O., bibliotecas de threads ni el lenguaje de programación)?

## Concurrencia

1. ¿Cómo evitaría la aparición de deadlocks en los programas multi-thread que Ud. escribe?
2. Un procedimiento reentrante puede ser utilizado por varios threads/procesos concurrentemente de forma segura, es decir, sin generar race-conditions. Un procedimiento reentrante puede ser interrumpido en cualquier momento (ej. Por el planificador) y continuando más tarde sin ocasionar problemas. En otras palabras, pueden existir varios threads/procesos ejecutando el mismo procedimiento reentrante concurrentemente. Explique cómo programaría un procedimiento reentrante para maximizar la concurrencia.
3. Considera importante que los procedimientos dentro de una biblioteca de vinculación dinámica sean reentrantes? ¿Por qué? ¿Bajo qué condiciones esto puede ser importante?

## Threads y Programación Concurrente

1. ¿Para qué sirven y por qué son ventajosos?
2. ¿Qué es un race condition?
3. ¿Cómo evita su aparición?

## Sincronización

1. ¿Por qué generalmente se evitan las esperas activas (también llamadas ocupadas, busy waits o spinlocks) cuando se programan aplicaciones concurrentes?
2. ¿Cómo se evitan las esperas activas?
3. A pesar de que las esperas activas son indeseables en muchos casos, se utilizan frecuentemente dentro del S.O. ¿Por qué?.

## Administración de Memoria

1. Se dice que los archivos mapeados a memoria son más rápidos que los archivos accedidos de forma común (open, read, write, close, etc). ¿Considera esta afirmación verdadera o falsa? Explique.
2. Las tablas de páginas multinivel típicamente se organizan en dos niveles. ¿Por qué no más o menos? Opina que existe algún límite práctico o teórico para el número de niveles posibles?
3. ¿Existe alguna relación entre el principio de localidad y la pregunta anterior? Explique
4. Explique principio de localidad y su aplicación para memoria virtual. Tiene alguna relación con caching?
5. ¿Por qué es importante diferenciar entre direcciones lógicas y físicas?
6. ¿Cómo se logra la diferenciación entre direcciones lógicas y físicas?
7. ¿Tiene alguna ventaja? ¿Para la protección de memoria?
8. Explique qué es fragmentación interna y externa. De ejemplos de cada una de ellas. Menciona técnicas de administración de memoria y que tipo de fragmentación posee cada una.
9. ¿Puede ocurrir fragmentación en memoria RAM? En caso de que si, explique qué tipos de fragmentación puede ocurrir.
10. ¿Qué tipos de sistemas de administración de memoria sufren de fragmentación?
11. ¿Se puede eliminar la fragmentación? ¿Cómo?
12. ¿Qué es reubicación y para qué sirve?
13. ¿Qué es reubicación (memoria RAM), para qué sirve y cómo se implementa (2 alternativas)?
14. Explique servicios de administración de memoria que requieren de reubicación para sus implementaciones.
15. Menciones métodos de administración de memoria que requieren de reubicación para su implementación.
16. Describa y compare dos formas de implementar reubicación.

## Memoria Virtual

1. ¿Qué es trashing y por qué se produce?
2. ¿Qué es la anomalía de Belady y como se evita?
3. Explique el modelo del conjunto de trabajo (working set) y su aplicación para memoria virtual. ¿Qué son?
4. Aplicación del modelo de conjunto de trabajo para memoria virtual y caché.
5. Las tablas de página de un S.O. pueden ser muy grandes y requerir mucho espacio en memoria, especialmente si la ocupación de la misma es baja:

* Describa una técnica para almacenar las tablas de forma más eficiente en espacio.
* Analice cómo influye esa técnica en el tiempo de acceso a memoria.

1. Los mecanismos de memoria virtual se basan en una suposición acerca de la forma en que los procesos hacen uso de la memoria con el fin de evitar impactar significamente en la performance de ejecución de dichos procesos:

* ¿Cúal es la suposición en la que se basan los mecanismos de memoria virtual?
* ¿Qué sucede si dicha suposición no se cumple la mayor parte del tiempo?
* Esa suposición también se aplica a las técnicas de caching. ¿Por qué?

1. Explique si las siguientes afirmaciones son verdaderas o falsas y justifique:

* La memoria virtual necesita de soporte de hardware para ser implementada eficientemente.
* Copy on write sirve para acceder con facilidad a archivos de gran tamaño.
* Los mecanismos de reubicación sólo son necesarios cuando se utiliza memoria virtual.

1. Copy on write (copia en escritura);

* ¿Qué es y cómo funciona?
* ¿Para qué se utiliza?
* ¿Cuál es su relación con la llamada al sistema fork?

1. Explique cómo influyen los mecanismos de administración de memoria basados en tablas de páginas y memoria virtual en el Tiempo Efectivo de Acceso a Memoria.

## Protección

1. ¿Cómo se logra protección de hardware?
2. ¿Cómo se logra protección entre procesos?
3. ¿Es posible lograr protección de hardware sin que el CPU provea algún soporte? ¿Cómo? Justifique.
4. ¿Qué es protección?
5. ¿Para qué sirve?
6. ¿Por qué es crucial en S.O. con multiprogramación?

## I/O

1. Generalmente las operaciones de I/O a nivel usuario se realizan de a bytes, mientras que los dispositivos sólo son capaces de leer/escribir de a bloques de tamaño fijo (por ej. De 4KB). ¿Cómo se manejan estas diferencias a nivel del sistema? ¿Por qué cree que los dispositivos de almacenamiento operan con bloques en vez de bytes?
2. Cuando varios procesos realizan I/O, a veces los S.O. atienden esos pedidos sin respetar el orden en que fueron realizados. ¿Por qué no se respeta el orden? ¿Eso puede traer problemas?
3. Por lo general en los lenguajes de programación los llamados al sistema que involucran I/O son del tipo bloqueantes (sincrónicas) y el S.O. oculta a los procesos y al programador la complejidad del manejo de esperas que impone el asincronismo en la interacción con los dispositivos. Sin embargo, recientemente algunos lenguajes de programación entre los que se destaca JavaScript con NodeJS adoptaron un modelo de I/O asincrónica:

* ¿Por qué cree que se hace esto?
* ¿Qué implicancias tiene esto en la programación de software respecto del modelo sincrónico?

## Sistemas de Archivos

1. ¿Qué significa que un sistema de archivos posea inconsistencias?
2. Describa al menos dos inconsistencias típicas y posibles métodos para solucionarlas de forma automática.
3. Describa RAID y al menos dos de sus variantes.
4. Explique el funcionamiento de los sistemas RAID.
5. ¿Considera a RAID un complemento o un reemplazo de los sistemas de archivos con log?
6. Compare RAID versus sistemas con logs (Journaling).
7. ¿Tiene sentido combinar RAID con sistemas de archivos con logs (Journaling)? Justifique.
8. Describa al menos dos inconsistencias que podrían ocurrir en sistemas de archivos FAT. Explique formas de detectar y arreglar esas inconsistencias.
9. Se desea agrandar o reducir el tamaño de un sistema de archivos. Explique cómo realizar eso en sistemas de archivos basados en FAT y en i-nodos.
10. ¿Qué es el formateo?
11. Al formatear un almacenamiento el espacio disponible resultante es menor que el que posee dicho almacenamiento sin formatear. ¿Por qué?

## Fragmentación en Sistemas de Archivos

1. ¿Por qué se produce?
2. ¿Qué problemas ocasiona?
3. ¿Cómo se evita?
4. ¿Cómo se reduce sin utilizar desfragmentadores?

## Bibliotecas de Vinculación Dinámica

1. ¿Qué son y cómo funcionan?
2. Ventajas y desventajas respecto de las bibliotecas de vinculación estática.
3. En una biblioteca de vinculación dinámica algunos de sus procedimientos mantienen estado de variables dentro de la biblioteca. La biblioteca se prueba con único proceso/thread miles de veces sin problemas, pero cuando se usa con más de un proceso/thread dichos procedimientos fallan.

* ¿Qué sucede y por qué?
* ¿Cómo solucionaría el problema sin reducir la concurrencia (es decir sin utilizar mecanismos de exclusión mutua)?

1. Suponga que varios procesos comparten una única copia de una biblioteca de vinculación dinámica. ¿Qué problemas podrían ocurrir? ¿Cómo los solucionaría?
2. Las bibliotecas de vinculación dinámica (también llamadas DLL o shared objects) tienen varias ventajas. Sin embargo pueden ocasionar problemas de dependencias conocido como DLL-hell. Una solución que proponen algunos sistemas consiste en incluir las bibliotecas en una carpeta junto con el ejecutable. Discuta los pros y contras de esas formas de utilizar bibliotecas de vinculación dinámicas.

## Planificación de CPU

1. Explique qué es un planificador de CPU con desalojo.
2. Describa 2 algoritmos de planificación de CPU con desalojo.
3. ¿Este tipo de planificador posee alguna ventaja en sistemas interactivos? Explique.
4. ¿Este tipo de planificador posee alguna ventaja en sistemas de tiempo real? Explique.
5. ¿Cómo influye el planificador CPU con desalojo en la programación de aplicaciones multithread respecto de un planificador sin desalojo?
6. En sistemas de planificación de CPU sin desalojo tales como Windows 3 (existe windows 3??) usualmente se lograba la ejecución de varios procesos de forma cooperativa. Explique dicha noción y menciones sus implicancias.

## Organización del S.O.

1. ¿Qué es un microkernel?
2. ¿Por qué se dice que los sistemas basados en microkernel son generalmente más seguro y robustos?
3. Compare microkernel con alguna otra organización del S.O. que conozca.

## Virtualización (Máquina Virtual)

1. ¿Qué es y para qué sirve?
2. Ventajas y desventajas.
3. En sistemas de máquina virtual uno de los principales desafíos es lograr buen desempeño. ¿Cuáles considera son los aspectos críticos que enfrentan este tipo de sistemas para lograr buena performance?

## Almacenamiento secundario

1. ¿Por qué usualmente se organiza en particiones?
2. ¿Por qué se utilizan sistemas de archivos en vez de almacenar directamente en el disco sin un sistema de archivos? Mencione ventajas y desventajas.
3. Explique dos mecanismos a nivel sistema de archivos para soportar tolerancia a fallas y recuperación.

## Modo Dual

1. Explique qué es el modo dual del CPU.
2. ¿Para qué lo utilizan los S.O.?
3. ¿Es posible lograr protección sin modo dual?
4. Los antiguos procesadores tales como el Intel 8086 no soportaban modo dual. Esto tenía un gran impacto en los S.O. que se podían implementar, por lo que la mayor parte de ellos no soportaban múltiples usuarios. Discuta algunos de los problemas que podrían surgir al tratar de construir un S.O. multi-usuario en esos procesadores.

* ¿Cuáles son las dos diferencias principales entre las operaciones de semáforos wait/signal (up/down o P/V) y las operaciones wait/signal de variables de condición en monitores?
* Explique las diferencias entre semáforos y monitores/variables de condición.