Preguntas parciales sistemas operativos.xlsx

Sistemas Operativos (Universidad Nacional de La Matanza)

Modulo: introduccion a los Sistemas Operativos

En un sistema monoprocesador es posible que se genere en un mismo instante dos interrupciones por software.

FALSE

Un solo proceso se ejecuta en un instante

En un sistema monoprocesador es posible que se generen en un mismo instante dos interrupciones de hw.

TRUE

Dos modulos de io que terminan de efectuar sus operaciones y necesitan de notificar al procesador. Otro ejemplo la coincidencia de clock con el retorno de una IO.

En que modo de funcionamiento debe estar el procesador, en caso de ser dual, para que puedan ejecutarse instrucciones privilegiadas

Modo Kernel/Privelegiado.

El primer procesador de la familia Intel que trajo incorporado el modo dual de operaciones fue el 80286

TRUE

Recordar que los anteriores lo carecían y por eso era que en DOS se podría llegar a escribir en área de so desde un usr por ejemplo.

Cuando se inicia un computador en Cold satrt el kernel es el primer proceso del so en ser cargado en memoria. Sin embargo, esto no sucede cuando se inicia en warm start.

FALSO.

En cold start, el Sistema Operativo inicialmente se encuentra almacenado en la memoria secundaria. Al encender la máquina se carga y ejecuta un pedazo de código que se encuentra en la Memoria Central(ROM), el cual carga el BIOS (Basic Input Output Sistem), y este a su vez carga el Nucleo o Kernel del Sistema Operativo en Memoria Central (RAM). Inicializa la máquina y carga todos los programas de aplicación y software necesario para permitir ejecutar programas o comandos del Shell. La rutina de inicialización del sistema operativo es la primera que se ejecuta. Existen dos tipos de verificación de recursos, la primera es realizada por el BIOS, y la segunda es realizada por la rutina de inicialización del sistema operativo. En Warm start, no se ejecutan las rutinas de vrificación de recursos (residentes en la BIOS o Kernel), sino que el proceso arranca directamente de la ejecución del proceso de inicialización del sistema operativo. (El kernel no es un proceso)

La interrupcion con mayor prioridad de todas se clasifica como HW externa.

FALSE

NMI

Cual es la diferencia entre una instruccion en modo usuario y una en modo kernel

Basicamente:

Quien la efectua-> Kernel ->SO Usuario->Proceso Usuario

Si es de ejec atómica en su secuencia-> atómica->kernel (SALVO NMI), interrumpible->USR.

Su estado ante la PSW. Kernel-> Pone 1 en modo en la PSW Usr -> Pone 0 en la psw en modo.

El procesador se entera de que hay una interrupción de HW externa ya que se le activa el FLIH.

TRUE

Los Procesadores que soportan modo dual en general son más seguros que los que no.

TRUE

Permiten la existencia de un set de instrucciones (Privelegiadas-> SO, Comunes -> Usr)

Explique la diferencia entre Context switch y Process Switch

Cambio de contexto es producto de la existencia de una interrupción mientras que el process switch es un cambio de proceso

La instrucción Halt (Detención) se ejecuta en modo usuario.

FALSE

Modo Kernel, porque afecta a otro proceso y por ende requiere intervención del SO.

Cuando se presiona enter en una interfaz tipo Cli se produce una interrupción de software.

Falso

Interrupcion de hardware

Cuando se presiona ENTER en una interface tipo CLI se produce una interrupción de software

FALSE

Interrupcion de hardware externa. Externa porque no proviene del procesador. Se activa el FLIH

Un sistema con un solo procesador no puede implementar multiprogramación

FALSE

La Multiprogramacion de sistemas monoprocesador implica tener un cojunto de procesos activos residentes en MC.Se multiplexean los recursos para dar la impresión de que se estan ejecutando concurrentemente.

Si tenemos una computadora con dos procesadores, en esta computadora nunca podrán producirse dos interrupciones de hardware internas al mismo tiempo

FALSE

Ejemplo: P1 en CPU1 y P2 en CPU2 P1 TRAP div 0 y P2 TRAP div 0

Siempre que hay un cambio de contexto de ejecución se cambia el estado del modo dual de operaciones.

FALSE

Ejemplo: supongamos que se esta ejecutando el sistema operativo atendiendo una interrupcion, y llega una interrupcion no enmascarable que debe ser atendida instantaneamente, por lo tanto habria un cambio de contexto debido a la ejecucion de la nueva rai pero no hay necesidad de cabiar de modo ya que el modo actual seria kernel.

Las interrupciones por fin de quantum son internas al procesador, el bit de FLIH no se enciende.

FALSE

Las interrupciones de clock son de hardware externo y por ende activan el FLIH

La interrupción con mayor prioridad de todas se clasifica como Hardware externa.

FALSE

Las interrupciones se clasifican según la prioridad en NMI o enmascarables. Luego se clasifican según el origen en ISW IHWE IHWI

¿En qué modo de funcionamiento debe estar el procesador, en caso de ser dual, para que pueda ejecutarse instrucciones privilegiadas?

Modo Kernel

Explique en que consiste un context switch ¿En que difiere de un process switch?

Context switch: es salvar el estado actual del procesador.

Process switch: es cambiar el proceso que tiene asignado el procesador. Obiamente implica de un context switch previamente.

Los compiladores permiten convertir un programa en proceso.

FALSE

Los compiladores tranforma un codigo fuente en un codigo objeto.

¿Que son las interrupciones? Explique porque algunos autores utilizan el término "interrupción de software" y que significado tiene. Mencione además que es el vector de interrupciones.

La interrupcion es la detencion de una ejecucion por la necesidad de reconocer un evento. Las interupciones de software, son interripciones generadas como llamados desde los procesos usuarios en esta categoria tenemos las syscall. El vector de interrupciones es una tabla de punteros a RAI.

Cuando un proceso necesita una operación con un módulo de E/S (imprimir) el proceso produce una interrupción, se salva su contexto y se pasa el control al tratamiento de interrupciones.

verdadero.

el proceso emite una syscall se produce el cw y se atiende la interrupcion.

El vector de interrupciones es una lista que contiene la dirección de comienzo de cada una de las rutinas de servicio de las interrupciones.

TRUE

En general, la comunicación entre las diversas capas de una arquitectura jerárquica es más rápida que en una arquitectura modular.

FALSE

La jerarquica solo admite la comunicación entre una capa y su inmediata inferior por este motivo aveces comunicarse con una capa implicaria pasar por algunas previas de las que no se requiere su participacion. Por Otro lado la modular todas los modulos estan relacionados.

La interrupción generada por la terminación de un evento de I/O de un dispositivo, generalmente es no enmascarable.

FALSE

generalmente es enmascarable.

Explique brevemente la técnica de spooling, sus ventajas y aplicaciones. Ejemplos.

Tipico de la cola de impresión, donde los procesos tienen la sensacion de usar simultaneamente el dispositivos. Los procesos emiten la operación de entrada y salida la cual es tranferida a un proceso demonio quien se encargar finalmente de la operación mediante buffer a la velocidad de los dispositivos sin necesidad de intervencion del proceso emisor de la operacion, el cual puede continuar su ejecucion mientras se efecuta la I/o.

En la estructura jerárquica resulta difícil la reconfiguración y actualización ya que cada módulo conoce los servicios brindados por el modulo inferior, pero no saben cómo se implementan.

FALSE

Teoricamente los modulos fueron diseñados de tal forma que no tendria que ser alterado su funcionamiento por cambios de otros modulos.

Las interrupciones de software nunca pueden ocurrir cuando el proceso que ejecuta es parte del SO de la máquina.

FALSE

En un ambiente multiprocesador, se podria dar el caso que durante la ejecucion del so en un procesador, en otro se este ejecutando un proceso que emita la ISW

¿Cuál es la diferencia entre una instrucción en modo usuario y una en modo kernel?

Quien la ejecuta: Kernel->SO, Usuario->Procesos de usuario

Durante el modo kernel la ejecucion de una secuencia de instrucciones es atomica (solo una interrupcion no enmascarable podria llegar a interrumpirlas), encambio la secuencia de instrucciones de usuario si pueden ser interrumpida.

Las instrucciones kernel tienen acceso ilimitado sobre los recursos, mientras que las de usuario se limitan al contexto del proceso invocante.

Revisar respuesta de pibe para mi esta mal, cuando hay una trap deberia ser atendida en modo kernel al igual que las syscall.

El primer procesador de la familia Intel que trajo incorporado el modo dual de operaciones fue el 80286

FALSE

Posterirores

Si tenemos una computadora con un solo procesador, en esta computadora nunca podran producirse dos interrupciones de HW al mismo tiempo

FALSE

Por ejemplo dos dispositivos trabando en paralelo terminan de efectuar la operación al mismo tiempo

¿En que modo de funcionamiento debe estar el procesador, en caso de ser dual, para que pueda deshabilitarse la lectura del FLIH?

Modo Kernel

Si un proceso está realizando cálculos (CPU Bound) este puede ser interrumpido temporalmente por una interrupción de software proveniente de un proceso de mayor prioridad.

FALSE

Primero si el sistema es monoprocesador es imposible que suceda la interrupcion por que es el unico programa en ejecucion. Si es multiprocesamiento la interrupcion se atenderia en el mismo procesador que ejecuta el procesador llamante.

Preguntar: Que pasa si se tendria asignado un procesador puntual para la ejecucion del sistema operativo.

En un sistema multiprocesador multiprogramado, el SO puede atender dos o más interrupciones de hardware simultaneas.

TRUE

Si el sistema operativo puede copiarse en los dos procesadores es cierto

Preguntar: MultiProgramado solo aplica a monoprocesador como inidca el apunte.

Cada vez que un SO recibe una interrupción de HW se produce un context switch.

TRUE

Siempre se produciria un cambio de contexto, si se esta ejecutando un proceso usuario cuando se la recibe deberia provocarse el cambio de contexto, para salvar el estado actual del procesador. Si ya se esta atendiendo una interrupcion y llega otra no enmascarable abria un cambio de contexto para conservar el estado de la rutina de atencion de interrupcion corriente.

Preguntar ahora que pasa con esta situancion se esta atendiendo una no enmascarable y llega la enmascarable, cuando se termina de atender la no enmascarable se retorna el control a quien la produjo y por ende abria un cambio de contexto porque abria que atender a la enmascarable, o se atiende a la nueva sin producirse cambio de contexto

Las interrupciones de software siempre generan un process switch, un context switch y un cambio de modo de ejecución.

FALSE

Según los pibes es verdadero para mi es falso.

Si la syscall implica que el proceso se bloque si abria process switch, pero si la syscall no es bloqueante como podria ser un fork no necesariamente abria un process switch.

CONTEXT switch y cambio de modo siempre porque las interrupciones de software son invocadas desde el proceso usuario.

El kernel de un sistema operativo moderno no tiene PCB ya que el sistema operativo no necesita contar con la información de ese proceso.

FALSE

Osea falso por lo de moderno, pero en realidad los kernel de los sistemas operativos no tienen pcb porque no son procesos sino un conuunto de rutinas.

Cada vez que el SO recibe una interrupción de hardware, se produce un process switch.

FALSE

Primero si la interrupcion se la recive durante la ejecucion de una rai de una interrupcion de menor prioridad no abria process switch. Segundo depende de la planificacion si no es expropiativa se atenderia la interrupcion y luego seguiria la ejecucion de la rai actual.

```
En un sistema multiprocesador multiprogramado, el SO puede atender dos o mas interrupciones
     hardware simultaneamente
```

La comunicación entre módulos en un SO de microkernel genera menos overhead que en un SO jerárquico.
RRta --> [Verdadero] En una estructura microkernel, los servicios se efectúan mediante la técnica de message passing.
En cambio en una estructura en estratos, algunas comunicaciones son entorpecidas por la jerarquía.
Por ej.: si la administración de la CPU necesita un acceso a disco se encuentra en un nivel inferior al administrador de Entrada - Salida,
entonces debe pasar por los servicios de otros administradores que se encuentran en el medio.

n la estructura jerarquica resulta dificil la reconfiguracion y actualizacion ya que cada modulo conoce los
los servicion brindados por el modulo inferior, pero no saben como se implementan.
RRta--> [Falso] Se ha visto que los diseños jerárquicos son más fáciles de depurar, modificar y verificar. En los
diseños en que el núcleo está distribuido en varios niveles de jerarquía, elegir qué función colocar en
cada nivel requiere un análisis cuidadoso. En tales diseños, con frecuencia sólo se permite hacer
llamadas a funciones situadas jerárquicamente por debajo de quién hace la llamada; es decir, cada nivel
sólo puede llamar a las funciones que están colocadas en el nivel inmediato inferior.

Un sistema con un solo procesador no puede implementar multiprogramacion
RRta --> [Falso]La multiprogramación se refiere a varios programas activos, residentes en una memoria central ejecutándose sobre una CPU.
Lo que no se puede implementar con un solo procesador es el Multiprocesamiento (multiprocessing) que significa que se usan
varios procesadores para ejecutar los procesos.

Cuando un proceso del SO solicita un servicio a otro modulo del SO no se produce un system call ya que no hay cambio de contexto
de ejecucion.
RRta --> [Verdadero]
Los servicios que provee el S.O. a los programas usuarios cuando estos se ejecutan, son
denominados llamadas al sistema (System Calls o SYSCALL).

El concepto de proceso solo se aplica a los programas
de usuario, el código del sistema operativo es una entidad aparte que opera en modo privileg

Siempre que hay un cambio de contexto de ejecucion se cambia el estado del modo dual de operaciones
RRta --> [Consultar]
El cambio de contexto se producirá en caso de la ejecución de una instrucción privilegiada, una
llamada al Sistema Operativo o una interrupción, es decir, siempre que se requiera la atención de algún
servicio del Sistema Operativo, el procesador se pasa al modo kernel y el control
pasa al S.O.. Con tal fin, se salva el contexto del procesador y tiene lugar un cambio de contexto mediante
una rutina del S.O..

Las interrupciones de SW nunca pueden ocurrir cuando el proceso que ejecuta es parte del SO de la maquina.
RRta
Cada vez que un SO recibe una interrupcion de HW se produce un context switch
RRta --> [Consultar]
Cambio de contexto [Mod 2 - pag 22]

Cuando se acaba el tiempo de quantum de un proceso, el SO puede o no efectuar un context switch
RRta --> [Verdadero?]
Puede producirse un context switch por ejemplo en el caso de que haya uno o mas procesos en la cola de listos
y al finalizar el quantum del proceso en ejecucion, se desaloja dicho proceso del CPU y se le otorga el procesador
al primer proceso de la cola de listos.
Por otro lado, una situacion donde no se produce un context switch aunque se le acabe el quantum al proceso en ejecucion
es aquel caso donde, salvo el proceso nulo, el unico proceso en el sistema es el que se encuentra actualmente en ejecucion,
con lo cual una cuando se termina el quantum no se hace el context switch sino que el proceso ejecuta hasta acabar.

En un sistema monoprocesador multiprogramado, el SO puede atender dos o más interrupciones en forma simultanea
FALSE
un solo proceso esta en ejecucion en un sistema monoprogramado asi que no se realizarian dos o mas llamados en forma simultanea.
Mal la justif

preguntar una trap es una interrupcion que requiere modo kernel

Modulo 2: Procesos

Un proceso pasa a la cola de bloqueados cuando solicita I/O al sistema operativo

TRUE

El process control block (PCB) es creado por el planificador de largo plazo cuando el proceso llega al sistema y eliminado por el planificador de corto plazo cuando el proceso finaliza su ejecución.

FALSE

Ambas tareas son resueltas por el largo plazo.

El PCB de un proceso solo se modifica cuando el proceso pasa del estado de listo a ejecucion o viceversa

FALSE

Teniendo en cuenta que guarda info de estado por ejemplo en cada cambio de estado que sufriera seria modificado. Con cada Context switch se producirían cambios ya que abria que salvar la psw por ejemplo.

Cuando un proceso esta en estado READY aun no tiene su PCB creado, éste se crea al otorgarselo al procesador

FALSE

El pcb Es creado por el largo plazo y es encolado en estado de ready o suspend ready si esta disponible.

Un proceso hijo comparte parte del PCB con su proceso padre

FALSE

Siempre todo proceso tienen un PCB propio, en el caso puntual de los hijos heredan parte de info de su padre.

Cuando un proceso finaliza su ejecucion, toma control el planificador de corto plazo para destruir su PCB,generar estadisticas, entre otras tareas.

FALSE

El planificador de largo plazo destruye los pcb y libera los recursos y libera y limpia las áreas de memoria empleadas.

El dispatcher es el encargado de ordenar la cola de listos

FALSE

Trafic controler.

Explique el mecanismo de creación de un nuevo proceso.

Asignamos un PID. Asignamos un espacio de memoria. Inicializamos El PCB asignado. Se establecen las relaciones con otras estructuras de datos y se amplían y crean las necesarias estructuras de datos.

La creación de un nuevo proceso puede darse como resultado de la ejecución de un proceso, por parte del OS o por el inicio de sesión de un usuario por ejemplo.

El PCB es creado por el planificador de corto plazo cuando llega un proceso al sistema.

FALSE

Largo Plazo

Un proceso pasa a la cola de bloqueados cuando solicita I/O al sistema operativo

FALSE

Un proceso se bloquea al estar ejecutando en espera de un evento por ejemplo solicitar un recurso el cual no esta disponible o efectuar una io, o por decision de otro proceso. En un modelo de dos estados tomamos como bloqueado al estado de no ejecucion.

El PCB es creado por el planificador de corto plazo cuando el proceso llega al sistema.

FALSE

El planificador de largo plazo o de trabajos es quien crea los pcb ya que es el que recibe los trabajos, y asigna a la cola de listos cuando los recursos necesarios esten disponibles. El planificador de corto plazo se compone del despachador quien elige de la cola de listos el proceso a ejecutar, y del controlador de trafico quien es el que ordena la cola de listos. Tambien el cotro plazo es quien verifica las interrupciones y quien desaloja los procesos cuando maneja algoritmos expropiativos. Ambos planificadores son componentes del OS.

La creación del PCB o vector de estado no genera overhead.

FALSE

Es una tarea administrativa del so que no beneficia al usuario y que ocupa tiempo de procesador. Por lo tanto es puro overhead.

El proceso nulo ejecuta con la máxima prioridad pero se la asigna un quantum de tiempo cercano a cero.

FALSE

El proceso nulo existente en los sistemas multiprogramados, es el proceso que existe desde que en el inico del so se arma la cola de listos, y es el de menor prioridad, su funcion minimamente es ser el proceso que se ejecuta mientra que no haya otro proceso en la cola de listo. Aveces lleva acabo tareas administrativas. Permite que el planificador que tiene elevada prioridad no este constantemente verificando la cola de procesos y impidiendo la llegada de nuevos procesos.

Efectue al ciclo de vida de los procesos de 9 estados, mostrando que estados abarca cada uno de los niveles

de planificacion. Explique el por que del nombre de cada uno de los planificadores y sus principales diferencias

El PCB de un proceso solo se modifica cuando el proceso pasa del estado de listo a ejecucion o viceversa

Falso

el PCB contiene informacion administrativa que se edita en cada cambio de contexto.

	En un entorno de multiprogramacion y monoprocesamiento siempre en conveniente utilizar hilos a nivel de kernel (KLT), ya que estos pueden competir por el uso del procesador con el resto de los procesos
	FALSE
	Depende de las actividades de los hilos y de como fueron programados. Sabemos que una gran desventaja de los KLT es el overhead por cambio de contexto y que se planifican al igual que el resto de los procesos. El uso de ult convinados con jacketing y si el proceso no tiene muchas solicitudes de syscall podria lograr un mejor resultado, debido a que correr en el mismo proceso para cambiar entre hilos del mismo proceso no haria falta context switch
	Un proceso hijo comparte parte del PCB con su proceso padre
	FALSE
	Es un proceso indepndiente con su propio PCB.
	En un entorno de multiprogramacion y monoprocesamiento siempre en conveniente utilizar hilos a nivel de kernel (KLT), ya que estos pueden competir por el uso del procesador con el resto de los procesos
	En un ambiente de multiprogramacion con monoprocesamiento podemos decir que, a persar de que los semaforos de un determinado proceso NO son pedidos o liberados por ningun otro proceso, igual puede ser necesario implementarlos.
	Verdadero. Procesos concurrentes.
	Suponga que tenemos un sistema con 4 microprocesadores, y un proceso que posee 4 threads. Asumiendo que esos threads ejecutan uno en cada procesador, la pregunta es: sabiendo que no se trata de multiprogramacion(multiplexado de procesamiento), sino que se trata de procesadores independientes (ejcucion en paralelo), ¿haria falta sincronizar esos threads? ¿Por que?
	Siempre en cuando existan recursos compartidos seria necesario sincronizarlos.
	En Threads ULT el SO es el encargado de su administacion.
	Falso. Definición de un KLT. El ULT es tarea del compilador.
	Los threads ULT se pueden migrar de proceso.
	Falso. Pertenecen al proceso creador.
	Utilizar hilos ULT junto con la técnica de jacketing permite al proceso que los generó aumentar su capacidad de direccionamiento.
	Falso. El uso de jacketing se relaciona en postergar las syscall para que no se bloquee el proceso.
	Si se construye un sistema basado en hilos klt, podemos afirmar que las instrucciones de dicho sistema se ejecutaran mas rápido que si tuviéramos solución sin hilos.
	Primero dependerá de como se haya programado osea si ubiese aprovechado las características de multi hilos.
	Los hilos KLT pueden implementarse en un SO multiprocesador.
	TRUE
	Los procesos livianos poseen su propio espacio de direcciones por lo tanto para compartir informacion entre ellos deben recurrir a mecanismos especiales provistos por el SO
	FALSE
	siempre encuando tengan el mismo padre todos comparten el espacio de direcciones.
	Si tengo un sistema de tipo I/O Bound resulta conveniente programarlo con threads ULT.
	Falso.
	Una solicitud de I/O es por syscall generalmente bloqueante lo que implica en los ult bloquear todos sus hilos ya que bloquea al proceso
	Los hilos KLT pueden implementarse en un SO multiprocesador.
	TRUE
	De echo en sistemas multiprocesador los klt son eficientes porque pueden ser asignados a distintos procesadores.
	Si se construye un sistema basado en hilos KLT, podemos afirmar que las instrucciones de dicho sistema se ejecutaran más rápido que si tuviéramos una solución sin hilos.
	FALSE
	Depende del ambiente y del proceso.
	Ante que nada depende del programador que alla construido un programa que saque provechos del procesamiento paralelo. Despues depende de la arquitectura por ejemplo si el sistema cuenta con un unico procesador o el sistema operativo es monoprocesador aun trabajando sobre un multiprocesador, los hilos klt no sacarian mucha ventaja debido a que provocarian cambios de contextos que no se producirian con una implementacion ult.
	La administración de los threads KLT es responsabilidad del usuario.
	FALSE
	Los klt los administra el so, encambio la administracion de los ult son responsabilidad de la aplicación
	Implementar hilos a nivel kernel es especialmente conveniente en sistemas con múltiples procesadores.

	FALSE
	Depende del programador,del tipo de aplicación, de que el sistema sea multiprocesador. Es cierto que respecto a hilos ULT, los klt harian uso de el resto de los procesadores. Una desventaja de los thread klt vs los ult es el overhead de los cw de hilos
	La técnica de "Jacketing" mejora el rendimiento de los threads haciendo que ellos sean colaborativos en lugar de competitivos.
	FALSE
	Es una rutina con el objetivo de convertir syscall bloqueantes en no bloqueantes, se postergan las ejecucion de las syscall hasta que no bloque al proceso. Util en sistemas ULT.
	Suponga que tenemos un sistema con 4 microprocesadores y un proceso que posee 4 threads. Asumiendo que esos threads ejecutan 1 en cada procesador, la pregunta es: sabiendo que no se trata de multiprogramación (multiplexado del procesamiento), sino que se trata de procesadores independientes (ejecución en paralelo), ¿haría falta sincronizar esos threads? ¿Por qué?
	Siempre se tenga acceso a recursos en comun sera necesario sincronizar los procesos independientemente de si se ejecutan en paralelo o no. Si por ejemplo no se sincronizan y tienen una region critica en la cual se accede a una variable compartida se podria obtener lecturas o escrituras erroneas sobre la mismam terminando la ejecucion en condiciones de carrera.
	Un proceso hijo comparte parte del PCB con su proceso padre.
	FALSE
	Es un proceso nuevo con su propio PCB.

Modulo 3: Planificacion

El process control block (PCB) es creado por el planificador de largo plazo cuando el proceso llega al sistema y eliminado por el planificador de corto plazo cuando el proceso finaliza su ejecución.

FALSE

Ambas tareas son resueltas por el largo plazo.

El algoritmo de planificacion Round Robin mejorado degenera en FIFO cuando el quantum tiende a infinito

TRUE

Algunos algoritmos de planificacion non-preemptive pueden implementarse como preemptive

TRUE

La planificacion a Extra Largo Plazo no la realiza el sistema operativo y por ende no afecta los tiempos de respuesta de los procesos

Falso.

Si bien uno pensaría que por no ser ejecutado por el SO, sus parámetros si afectaran las decisiones tomadas por los otros planificadores.

el dispatcher es un programa que pertenece al planificador de largo plazo y se ocupa de ordenar la cola de nuevos.

FALSE

Pertenece al corto plazo y se encarga de poner en ejecución los procesos.

El proceso nulo es el de más alta prioridad que se ejecuta en cada cambio de contexto.

FALSE

Es el de menor prioridad y permanece bloqueado mientras exista al menos un proceso disponible para la ejecución.

Explique las dif y similitudes entre modelos de planificación de 5 y 7 estados.

El planificador de 7 estados se compone del 5 asi que hereda sus características.

Ambos poseen los estados: READY-RUNNING-BLOCK-NEW-END.

Por otro lado el planificador de 7 estados complementa con los estados SUSPEND READY y SUSPEND BLOCK utilizados en los sistemas con SWAPING.

en un SO con algoritmo de planificación sin sustitución los procesos solo dejan la CPU cuando finalizan.

FALSE

Pueden abandonarla por la ejecución de una syscall para por ejemplo efectuar una operación de I/O Si el algoritmo lo contempla.

El dispatcher se ocupa de la planificación a largo plazo.

FALSE

Dos motivos. El despachador actua en el corto plazo, segundo no existe planificación a largo plazo, es un planificador de largo plazo, el plazo hace referencia a su tiempo de ejecución.

Los algoritmos con sustitución son los menos usados actualmente porque el overhead que necesitan ralentizan al sistema.

FALSE

Si bien es cierto que son poseen overhead estos algoritmos, en contra parte aprovechan de una mejor manera el procesador.

Por ejemplo los sistemas de tiempo compartido emplean algoritmos premtive con Quantum de tiempo.

El proceso NULO se crea cada vez que la cola de listos está vacía y finaliza después de un quantum de ejecución.

FALSE

El nulo se crea desde el inicio del so cuando se crea la cola, y entra en ejecución cada vez que la cola de listo esta vacia. Finaliza en el cierre del sistema.

El planificador de mediano plazo se ocupa de pasar los procesos de listos a running

FALSE

Este planificador se encarga del swaping de procesos. Los estados que manejan son los inactivos (bloqueado suspendido y suspendido listo). El planificador de corto plazo es quien pasa los procesos de ready->running running->block block->ready

La planificación Extra Largo Plazo la realiza el sistema operativo en base a las políticas de prioridades de usuarios.

FALSE

El planificador de extra largo plazo no es parte del sistema operativo. Se basa en las politicas de funcionamiento del sistema, por ende tendra encuenta las prioridades de los usuarios. Utiliza procedimientos escritos.

El dispatcher se ocupa de ordenar la cola de listos en el planificador de Mediano plazo.

Falso.

El dispatcher junto al trafic controler pertenecen al planificador de corto plazo. El traficc controler es quien ordena la cola de listos, mientras que el dispatcher quien toma los proceso de la cola de listo y los pone a ejecutar, o quien le retire a un proceso la cpu.

Un quantum grande favorece la ejecución de los procesos con mucha entrada salida.

FALSE

Favorece a los procesos intencivos en uso de procesador, los intencivos en I/O desperdiciarian generalmente los Quantum ya que al solicitar la I/O se bloquearian.

En un algoritmo de planificación non-preemtive, si un proceso toma la CPU, el sistema operativo no es capaz de recuperarla hasta que el proceso decida abandonarla por sí solo.

TRUE

El proceso solo abandona por su iniciativa (finalizacion, I/O), el sistema operativo no puede intercambiarlo por otro proceso durante su ejecucion, y tampoco se atienden las interrupciones de clock

El planificador de extra largo plazo se encarga de controlar que los procesos tengan todos los recursos para pasar a la cola de listos

FALSE

Quien efectua dicha tarea es el planificador de largo plazo.

El dispatcher es el encargado de ordenar la cola de listos y de asignar un proceso al procesador y pertenece al planificador a corto plazo.

FALSE

El despachador solo otorfa de la cola de listos al primer proceso y lo pone en ejecucion. Quien ordena la cola de listos es el trafic controler. Ambos pertenecen al planificador de corto plazo.

La planificación a extra largo plazo es algo que actualmente no se realiza en los sistemas operativos modernos ya que no se le encuentran ventajas respecto a la performance del sistema.

FALSE

Es quien tiene encuenta lass politicas de funcionamiento del sistema desde el punto de vista de la organización, y por ende por ejemplo le otroga las prioridades a los trabjos de acuerdo a las necesidades de los usuarios.

Los procesos que se encuentran en la cola de listos no están ejecutándose por tal motivo es incorrecto decir que son procesos.

Falso.

Se encuentran en memoria central asociados a su stack y con el pcb creado pro el largo plazo por ende es un proceso no es necesario que se este ejecutando.

La creación de un proceso en un SO es siempre resultado de una interrupción de software.

TRUE

Los procesos son creados por el sistema operativo mediante una syscall (interrupcion de software)

Cuando un proceso finaliza su ejecución, toma control el planificador de corto plazo para destruir su PCB, generar estadísticas, entre otras tareas.

FALSE

El planificador de largo plazo es quien destruye los PCB.

Si se utiliza un algoritmo FCFS para el planificador de corto plazo, podemos decir que existe tan solo un proceso que puede ser expropiado por el sistema operativo.

Verdadero.

El unico que puede ser expropiado serial el proceso nulo.

Cuando un proceso está en estado READY aún no tiene su PCB creado, este se crea al otorgárselo al procesador.

FALSE

Si esta en cola de listo es porque ya tiene un pcb. Cuando son creados los procesos por el largo plazo son creados junto al pcb.

Explique las diferencias entre un algoritmo de planificación con prioridad fija y uno de prioridad variable. ¿Podría un algoritmo utilizar ambos tipos de prioridad?

Prioridad fija externa: suministrada fuera del sistema y su valor es cte durante la ejecucion.

Prioridad variable Interna: Varian durante la ejecucion. Asignadas por el OS

Si se pueden combinar por ejemplo: los procesos tienen una prioridad inicial que se suma a criterios globales para obtner la prioridad final

Explicar brevemente el modelo de 7 estados del ciclo de vida de un proceso.

El modelo de 7 estados es aquel destinado a los sistemas con memoria virtual que pueden aplicar swapping. Es una especializacion del de 5 estados activos (listo, ejecutando, bloqueado, nuevo, terminado) incorporandole los dos estados inactivos (suspendido bloqueado - suspendido listo).

Un proceso que se encuentra en estado "terminated" en un diagrama de 7 estados, se conserva para que pase nuevamente a "running"

FALSE

En ningun modelo un proceso terminado puede pasar a listo si a la inversa.

En un sistema que trabaja con el algoritmo de planificación HRRN, solo los procesos nuevos tendrán el ratio igual a 1.

Taza de respuesta = (Tiempo espera + Tiempo Ejecución) / Tiempo Ejecución

Si el proceso se ponen en ejecucion inmediatamente se crea sin hacerlo esperar (tiempo de espera = 0) su ratio seria 1

	Los algoritmos de planificación preemptive producen más overhead que los non-preemptive pero son más equitativos.
	Falso.
	Depende puntualmente de los algoritmos, muy por encima si se podria afirmar la cuestion de overhead, pero la parte de equitativos no es cierto basta con mirar el algoritmo srt que prioriza a los procesos cortos como contra ejemplo.
	En un sistema con algoritmo de planificación non-preemptive, la ejecución de las RAI se demora hasta que el proceso en ejecución decide abandonar el procesador.
	FALSE
	El hecho de ser non-preemptive implica que un proceso no puede ser desalojado por el sistema operativo para intercambiarlo por otro proceso. Los procesos solo abandonan el procesador voluntariamente (I/O / RECURSO, FINALIZACION, SYSCALL Bloqueante). Las RAI por otro lado si pueden ser atendidas porque no son procesos. Por ende se puede atender cualquiere interrupcion durante la ejecucion de un proceso.
	Un proceso que se encuentra en estado "bloqueado" puede pasar al estado "listo" luego de la ocurrencia de una interrupción de hardware externa o por una interrupción de software.
	TRUE
	tanto Una IHE que indicaria la finalizacion de un dispositivo de I/O, como tambien una interrupcion de software podria provocar este echo (Semaforo V()) si el planificador lo indica podria pasar de bloqueado a list
	El algoritmo de planificación Round Robin mejorado degenera en FIFO cuando el quantum tiene a infinito.
	TRUE
	Si El q es suficiente grande los procesos podrian terminar de ejecutar antes que la interrupcion de clock y como RR se organiza en una cola FIFO Circular terminaria comportandose simplemente como FIFO.
	Cuando acaba el tiempo de quantum de un proceso, el SO puede o no efectuar context swicth
	TRUE
	Cuando el unico proceso en la cola de listo sea el nulo, no abra cambio de contexto, en otra situacion dependera del planificador, podria por ejemplo tratarse de una convinacion de Q + prioridades.
	Un proceso pasa a la cola de bloqueados cuando solicita I/O al sistema operativo.
	TRUE
	Los planificadores mencionados en la catedra ante una solicitud de entrada y salida o recurso no dispoble se bloquean. Podria llegar a existir planificaciones poco eficiente que no bloqueen a un proceso durante la operación de una entrada y salida.
	El process control block (PCB) es creado por el planificador de largo plazo cuando el proceso llega al sistema y eliminado por el planificador de corto plazo cuando el proceso finaliza su ejecucion
	FALSE
	El planificador de largo plazo finaliza los procesos.
	Algunos algoritmos de planificación non-preemptive pueden implementarse como preemptive.
	TRUE
	Tipico de los algoritmos por prioridades como el SPN (np) y SRT (p).
	Todos los algoritmos de planificación de procesos preemptive pueden implementarse como algoritmos non-preemptive.
	FALSE
	Por ejemplo RR si se lo adaptara perderia la funcionalidad del algoritmo se terminaria convirtiendo en un FSFC
	La planificación a extra largo plazo no la realiza el Sistema Operativo y por ende no afecta a los tiempos de respuesta de los procesos.
	FALSE
	Definen prioridades estaticas que seran tenida encuenta durante la planificacion.
	Explique las diferentes formas de implementar la planificación por múltiples colas y el motivo para que existan varias. Compare las distintas formas indicando diferencias y similitude
	La idea de multiples colas es tener colas con distintas prioridades y iguales o diferentes algorimos de ogranizacion de la misma. Los proceso pueden pertencer a una misma cola durante toda su ejecucion coloas fijas o puede cambiar de colas con retro alimentacion en colas dinamicas con feeback.
	El algoritmo de planificación SJF se puede implementar para la cola de nuevos, pero no para la cola de listos.
	TRUE
	SJF es implementado por el largo plazo durante la creacion de procesos. Existe un algoritmo similar implementado en el planificador de corto plazo que es el SPN o el SRT basados en el SJF pero en escencia no es el mismo.
	El objetivo principal de la planificación es evitar la inanición de todos los procesos activos del sistema.
	TRUE
	El planificador deberia lograr la mayor eficiencia en el procesamiento.
	Un quantum grande favorece la ejecución de los procesos con mucha entrada salida.

	FALSE
	Favorece a los procesos intencivos en uso de procesador, los intencivos en I/O desperdiciarian generalmente los Quantum ya que al solicitar la I/O se bloquearian.

Modulo 4: Concurrencia

Mutua Exclusión Implica que un recurso pueda ser accedido simultáneamente por 2 o más procesos.

Falso. Justamente la condición de mutua exclusión asegura que un recurso crítico (1 punto de entrada) sea accedido por un solo proceso a la vez.

la Espera activa para ingresar a una región crítica es menos eficiente que la bloqueante.

verdadero. La espera activa implica que el proceso en ejecución desperdicie tiempo de procesamiento en consultar la disponibilidad de la región critica, mientras que en la bloqueante (sin espera actica) permite que el proceso sea bloqueado y intercambiado por otro proceso disponible.

Los mecanismos IPC se inventaron para poder transmitir informacion entre procesos alojados en distintos equipos

Verdadero, pueden aplicarse con ese objetivo pero nada impederia utilizarlo dentro del mismo equipo aunque con menos rendimiento talvez que compartir memoria si fuera posible

La mutua exclusion puede implementarse mediante el uso de las instrucciones especiales TAS y CAS solo en sistemas monoprocesadores

Falso. Puede aplicarse tanto en mono como en multi.

Los algoritmos de deteccion de deadlock solo funcionan si los recursos son de una sola entrada

FALSE

la comunicacion entre procesos solo es util en sistemas multiprocesamiento, ya que solo en este tipo de sistema pueden ejecutarse dos procesos simultaneamente.

Falso Concurrencia

¿Por que la cola de espera es un recurso critico?

Porque puede que se ejecuten poner /sacar simultáneamente sobre la cola

¿Cuales son los mecanismos provistos por el hardware para asegurar la mutuaexclusion?

TAS/CAS desabilitacion de interrupciones

Si hay un cambio de ejecución de threads siempre hay un CS pesado.

Falso. CW Liviano.

La mutua exclusión puede implementarse mediante la des habilitación de las interrupciones solo en los sistemas monoprocesador.

True.

Cuando cada proceso de un conjunto espera por un mensaje de otro miembro del grupo, y no existe un mensaje en tránsito, entonces ocurre un deadlock.

Verdadero.

Todos los procesos del conjunto T están bloqueados a la espera de mensajes de procesos perteneciente al conjunto y no existe mensajes en curso.

La inanición es una situación que se presenta cuando muchos procesos no pueden obtener muchos recursos individualmente.

Falso. Necesariamente no deben ser muchos. En una relación 1 a 1 podria darse inanacion. Osea el proceso quedara esperando la asignación de un recurso indefenidamente o por grande periodos de tiempo y nunca terminara de ejecutarse.

Existen deadlock si tengo 2 o mas procesos y 1 o mas recursos involucrados.

Verdadero si se cumple alguna de las condiciones de Coffman.

Si dos Instrucciones coinciden en sus conjuntos de entrada, entonces pueden ser ejecutadas en paralelo.

Falso. Justamente w1nw2=w1nr2=r1nw2 = vacio se debe cumplir.

Los semáforos son un excelente método de sincronización que se pueden usar tanto en ambientes mono como multiprocesador, pero no con threads.

fFalso puede implementarse con threads.

produce deadlock cuando un proceso no obtiene un recurso en un determinado tiempo.

Falso Starvation.

La implementación de mutua exclusión deshabilitando las interrupciones solo es factibles en sistema monoprocesador.

Verdadero.

para prevenir deadlocks basta con asegurar la no ocurrencia de algunas de las condiciones de Bernstein.

Falso. Deadlocks-> coffman. Concurrencia->Benesterin son 3

Un deadlock se recupera mediante la expropiación del procesador a uno de los procesos.

Falso, si bien podría llegarse a recuperar, en algunos casos n seria suficiente ejemplo.

Dado 3 procesos p1, p2, p3

P1 tiene asignado r1 y pide r2 y r3

P2 tiene asignado r2 y pide r1

P3 tiene asignado r3 y pide r1

Si expropiamos a P2 segueria existiendo deadlock entre p1 y p3 aunque desasparezca el que existía entre p1 y p2

Un deadlock se produce si un proceso retiene indefinidamente un recurso que otro proceso tiene.

Falso. Para empezar el proceso solicitante esta sufriendo inanacion, pero para que se produzca abrazo mortal el recurso que posee el proceso solicitante debería estar siendo solicitado por el que retiene el que solicita.

Dos procesos se pueden ejecutar concurrentemente si la intersección de sus conjunto de escritura son vacios.

Falso. Deben cumplirse las tres condiciones de bernestein.

En un sistema monoprocesador multiprogramado resulta mas eficiente sincronizar el acceso a un recurso, deshabilitar las interrupciones que el uso de semaforos

Falso. Semaforo seria mas eficiente por el echo que no limitarías el sistema deshabilitando las interrupciones. Dehabilitando interrupciones por ejemplo limitaríamos a planificaciones no expropiativas.

Los semaforos son una tecnica obsoleta, ya no se utilizan en los SO modernos. [Consultar]

Falso. Los semáforos pueden ser usados para diferentes propósitos, entre ellos:

- °Implementar cierres de exclusión mutua o locks
- °Permitir a un máximo de N threads (hilos) acceder a un recurso, inicializando el semáforo en N
- °Notificación. Inicializando el semáforo en 0 puede usarse para comunicación entre threads sobre la disponibilidad de un recurso.

la sincronización con semáforos es un mecanismo usado en sistemas distribuidos.

Falso. Implicaría compartir un espacio de memoria, lo cual no lo hace aplicable en un sistema distribuido.

La virtualización de dispositivos es una técnica que permite reducir deadlocks.

Falso. Solo estaríamos agregando disponibilidad de dispositivos pero eso no implica que sean todos necesarios.

```
Cuando se crea un proceso hijo puede compartir su codigo con su padre, si el codigo es reentrante.
```

Para que se cumpla un deadlock es necesario que se cumplan las 4 condiciones de Coffman.Por lo tanto, sería correcto afirmar que si evitamos solo una de ellas, un deadlock nunca ocurrirá

TRUE

Las cutro condiciones de coffman son

Mutua exclusion: solo un proceso puede acceder a la RC

Espera circular: un proceso espera el recurso asignado a otro que asu vez pide un recurso que tiene asociado el anterior. Se expande a mas de un proceso.

Esperar y retener: esperar recursos sin liberar los actuales

No expropiacion: la liberacion de recursos es responsabilidad del proceso.

Definir semáforo

Mecanismo de sincronizacion. Version con espera activa, y sin espera activa. Emplean una variable global que se incrementa mediante la primitiva P, se decrementa atraves de la primitiva V, y se inicializan con otra primitiva. Según la inicializacion de su variable se clasifican en mutex inicializan en 1 y sus posibles valores son 0(libre) y 1 (ocupado), contadores inicializados en 0 gralment y toman valores negativos positivo y 0, y binarios inicializados en 0 y toma valores 1 - 0

Definir deadlock

El abrazo mortal se da cuando 2 o mas procesos esperan por recursos que tienen asignados procesos del grupo y que esto asu vez piden el recurso de ellos. En comunicación es cuando los proceso miembros de la comunicación esperan por mensajes que no se transmiten

Explique tres métodos diferentes para recuperar el deadlock.

Abortar todos los proceso involucrados en el deadlock y volver a ponerlos en ejecucion probablemente se vuelva a dar la condicion

Ir abortando procesos del conjunto que forma el deadlock hasta que desaparece la condicion

Expropiarse del recurso de uno del grupo y pasarlo al otro solicitante

Establecer puntos de control para cuando se de el abrazo mortal volver al punto de control. Es posible recaer de nuevo el el abrazo mortal.

¿Cuál es la diferencia entre comunicación y sincronización entre procesos?

La sincronizacion es el ordenamiento expreso en el tiempo de un conjunto de operaciones. Mediante la misma se puede establecer que un proceso espere la ocurrencia de un determinado evento para continuar su ejecución.

La comunicación por otro lado esta destinado al intercambio de informacion en un grupo de proceso. Mediante aplicaciones de la misma se puede lograr sincronizar procesos

La comunicación entre módulos de un SO de microkernel genera menos overhead que un SO jerárquico.

TRUE

La comincacion en microkernel es mediante el intercambio de mensajes. En cambio en los so jerarquicos la comunicación es entre modulos adyacentes, lo que entorpece la comunicación entre dos modulos no adyacentes..

La mutua exclusión puede implementarse mediante la deshabilitacion de las interrupciones solo en sistemas monoprocesadores

TRUE

En sistemas multiprocesadores carece de sentido esta tecnica porque no garantiza que dos proceso en distintos procesadores soliciten entrar a la RC al mismo tiempo.

Las instrucciones TAS están implementadas por el sistema operativo.

FALSE

Son instrucciones propias del procesador. Set de instrucciones

¿Cuáles son los mecanismos provistos por el hardware para asegurar la mutua exclusión?

TAS: Compara el valor actual del registro si es 0 lo ponen en 1 y devuelve true, si esta en uno lo deja en uno y devuelve falso. Instrucción del set de instrucciones. Util en multiprocesamiento.

CAS: Los procesos tienen asignados un numero distinto de 1, la instrucción CAS intercambia el valor de un registro inicializado en 0. Solo podran entrar a la rc los proceso que encuentre en 0 a la rc. El proceso que logre entrar a la rc al salir ejecutara otra ves la cas volviendo a quedar en 0 el valor del registro. Instruccion del set de instrucciones. Util en multiprocesamiento

Deshabilitar interrupciones: Util solo en sistemas monoprocesador se deshabilita la llegada de interrupciones asegurando que el proceso no sea interrumpido en RC

Los semáforos son una técnica obsoleta, ya no se utilizan en los SO modernos.

Falso. Los semáforos pueden ser usados para diferentes propósitos, entre ellos:

- °Implementar cierres de exclusión mutua o locks
- °Permitir a un máximo de N threads (hilos) acceder a un recurso, inicializando el semáforo en N
- °Notificación. Inicializando el semáforo en 0 puede usarse para comunicación entre threads sobre la disponibilidad de un recurso.

Si dos instrucciones coinciden en sus conjuntos de entrada, entonces no pueden ser ejecutados en paralelo.

TRUE

Bernestein define dos conjuntos para una sentencia uno de escritura (variables alteradas durante una llamada) y otro de lectura (variables referenciadas pero no modificadas en una llamada). Mientras que los cruzes de los conjuntos de dos sentencias (Salvo R1 y R2) den vacio podrian ser ejecutadas concurrentemente.

Mediante la comunicación sincrónica pueden sincronizarse procesos, mientras que con la comunicación asincrónica y semisincronica esto no es posible.

FALSE

Sincronica y semisincronica son utiles para la sincronizacion, asincronica no sirve por el echo de no se bloquea nigun participante. Recordar definicion de sincronizacion requiere la espera de un proceso para continuar su ejecucion.

Dado un proceso P, con una region critica para el recurso critico A, y una region critica para el recurso critico B Y un proceso Q con una unica region critica para los recursos criticos A y B. Podria producirse deadlock entre ambos¿?

La mutua exclusión puede implementarse mediante el uso de las instrucciones especiales TAS y CAS solo en sistemas monoprocesador.

FALSE

Son las dos implementaciones de hardware de espera activa que son compatibles con multiprocesamiento

Explique cuál es la ventaja de usar threads ULT en un esquema de multiprocesamiento.

Poco overhead, se puede tener un algoritmo propio de planificacion para el proceso.

la única diferencia entre la comunicación directa e indirecta es que existe un almacenamiento temporario en una de ellas, mientras que en la otra no.

FALSE

Primero la directa tambien tiene almacenamiento temporario como amortiguador de diferencia de velocidades. Por otro lado la manera en que se establecen los link cambian, y como se referencian los emisores y receptones en la comunicación.

Los procesos comparten recursos que deben estar sincronizados necesariamente, sino podrían generarse race conditions.

FALSE

SI no se sincronizan se produce condicion de carrera.

Los semáforos son un excelente método de sincronización que se puede usar tanto en ambientes mono como multiprocesador.

TRUE

Siempre que se pueda compartir memoria entre los procesos es aplicable.

La comunicación entre procesos solo es útil en sistemas multiprocesamiento, ya que solo en este tipo de sistema pueden ejecutarse dos procesos simultáneamente.

FALSE

No solo sirve para el multiprocesamiento, tambien sirve para procesos concurrentes en monoprocesador.

Dos procesos pueden sincronizarse mediante la comunicación utilizando comunicación asincronica

FALSE

Justamente en la difinicion de sincronizacion esta esperar la ocurrencia de un evento para continuar la ejecucion. En el caso de asincronica no hay bloqueos.

La inanición es una situación que solo se presenta cuando muchos procesos no pueden obtener muchos recursos individualmente.

FALSE

Inanacion se da cuando almenos un recurso no puede completar su ejecicon debido a la carencia de un recurso.

Los algoritmos de sincronización con espera activa permiten un mayor aprovechamiento de la CPU, ya que mantienen ocupada un mayor porcentaje de tiempo.

FALSE

Al revez los procesos desperdician su tiempo esperando ocupando el procesador.

Los mecanismos IPC se inventaron para poder transmitir informacion entre procesos alojados en distintos equipos

FALSE

Permite la comunicación entre procesos independientemente de la ubicación

	La mutua exclusion puede implementarse mediante el uso de las instrucciones especiales TAS y CAS solo en sistemas monoprocesadores
	FALSE
	Sirve tanto para monoprocesamiento como para multiprocesamiento
	Suponga un sistema que implementa multiprogramacion en un entorno de multiprocesamiento donde se ejecutan procesos largos CPU bound, procesos largos I/O bound y procesos cortos CPU bound. Si quisiera favorecer la ejecucion de los procesos I/O bound cual de los siguientes esquemas eligiria (justifique su respuesta):
	a) Planificador de trabajos SJF, planificacion de procesos JPF e hilos ULT
	b) Planificador de trabajos FIFO, planificacion de procesos RR con Q grande e hilos KLT
	c) Planificador de trabajos FIFO, planificacion de procesos RR con Q pequeño y procesos pesados
	d) Planificador de trabajos FIFO, planificacion de procesos RR con Q pequeño e hilos KLT
	Descarto opcion b t C porque hacen uso de KLT lo que implican overhead por los CW, ademas b por usar Q grandes que no son aprovechados por los I/O. No menciona aplicar tecnica de jacketing por lo cual no me inclinaria por la opcion a ya que se bloquiaria la mayoria de las veces por ser I/O bound.
	TAS y CAS son instrucciones provistas por el SO para garantizar la mutua exclusión en sistemas monoprocesador como multiprocesador.
	Verdadero.
	Los procesos concurrentes son necesarios de sincronizar unos con otros.
	TRUE
	Primero por la race condition. Tambien deben sincronizarse por el uso de recursos.

Modulo 5: Memoria

la Técnica de paginación segmentada con memoria virtual es una de las más usadas en la actualidad para administrar la memoria.
Falso. Paginación segmentada no es aplicable (En si estarían limitados los tamaños de los segmentos por el tamaño de los frame). Segmentación paginada es lo correcto donde lo que se paginada son los segmentos.

El algoritmo "Optimo" para el remplazo de páginas es el mas utilizado.
Falso. Es un algoritmo de aplicación teorica, debido a su alto costo de procesamiento al necesitar ver el futuro.

El tamaño de página en un sistema es importante a efectos de la fragmentación.
Verdadero. El tamaño de la página influye en la fragmentación interna.

En un sistema con N particiones Fijas de memoria puedo ejecutar hasta 2 procesos usuarios más que N.
Falso. Particiones fijas por definición es una técnica sin swapping, en la cual se asigna una partición por proceso.

Con sustitución global el proceso solo es afectado por su propio paginado.
Falso. Lo presentado es característico de la sustitución Local.

El sistema de paginación segmentada es más eficiente y simple que el de segmentación paginada.
Falso por que paginación segmentada no existe.

técnica de paginación, el tamaño de páginas es uniforme y lo define el compilador.
Falso. Tamaño uniforme. Los compiladores definen los tamaños de los segmentos en segmentación.

El tamaño de pagina en un sistema paginado es importante a efector de la fragmentación externa.
Falso. En los sistemas de pagina no existen fragmentación externa.

el algoritmo de reemplazo de páginas óptimo es el que mejor tasa de fallos de página tiene.
Verdadero.

Los tiempos de búsqueda en una TLB dependen de la cantidad de registros que contenga.
Falso. Se accede simultáneamente a todos sus registros.

La estrategia de asignación de memoria best fit es mas rápida que first fit.
Falso.

Aumentar la cantidad de frames otorgados a un proceso al utilizar LRU genera un aumento en la tasa de fallos de página de ese proceso.
Falso. Fifo Anomalia Belady.

Las TLB son un soporte de HW que mejora la velocidad de los cache
Falso, permite un acceso rápido a los datos en memoria si tener que acceder a memoria para consultar las tablas de mapeo.

El uso de paginación segmentada es eficiente si el tamaño de pagina es grande.
Falso. No existe paginación segmentada.

El uso de page directory en sistemas de bus de direcciones de 16 bit mejora significativamente la performance.
Depende de los tamaños de la MPT en tal caso aplicar multinivel en paginación podríamos descargar espacio ocupado en memoria y tener mas paginas para un proceso si disponemos de suficiente mv.

Resulta útil la aplicación de compactación sobre la mv en paginación bajo demanda.
El costo en overhead para compactar bloques de corta vida útil no tendría sentido.

En un sistema con paginación bajo demanda, una pagina que se modifico durante la ejecicon es la cadidata mas probable de ser victima.
Falso depende del alogoritmo de sustitución que se emple.

Las tecnias de administración con swapping son mas veloces que las técnicas sin swaping porque permiten incrementar el nivel de multiprogramación del sistema.
Falso. Si bien el nivel de multiprogramación aumenta, como costo tiene el overhead que implica acceder a los distintos niveles de memorias y tablas necesarias, o de los algoritmos de colocación y remplazo de áreas para hacer el intercambio.

Una manera de reducir la fragmentación es ampliar la capacidad de memoria RAM de un equipo.
Falso. La fragmentación es producto de los métodos de asignación de espacio, y no depende de la cantidad de espacio.
El método de sustitución local produce menos fallos de página que el método global.
Falso. No necesariamente, debido a que dependerá de la situación en la cual se desarrolla. Generalmente un remplazo global debería ser mas productivo que un local. Remplazo global se puede remplazar cualquier pagina del espacio usuario de memoria, encambio en la local solo en la del proceso. Por lo tanto en global no solo depende del pro

La paginación multinivel es la mas lenta que la paginación pura o simple debido a que requiere mas acceso a las tablas MPT para encontrar los datos.
Verdadero.

En un So que administra la memoria atraves de particiones variables el grado de multiprogramación es siempre fijo y tiene un máximo conocido.
Falso. Dependera de los procesos que se vallan ubicando en memoria.

La tasa de fallo de pagina de un proceso es inversamente proporsional a la cantidad de frames asignados
Falso. Primero depende del algoritmo dado que por ejemplo algunos sufren del efecto belady)

Enumere las funciones del administrador de memoria:
Registro del estado de las áreas de memoria.
Mapeo de direcciones lógicas / físicas.
Tecnicas de asignación y recuperación.
Proteccion de zonas de memoria

La imagen de memoria del proceso se divide en porciones todas del mismo tamaño.
False depende del algoritmo de asignación. Por ejemplo en Segmentacion no serian todas del mismo tamaño.

Con la técnica de particiones variables el nivel de paginación esta limitado por el numero de particiones
Falso, corresponde a particiones fijas la afirmación. En particiones variables dependerá de las particiones libres que se iran generando apartir de que los procesos comienzan a ubicarse en las distintas áreas y vallan pidiendo su respectivo espacio.

Una tasa de fallo de páginas elevada indica que nuestro sistema tiene demasiados procesos en ejecución.
Falso. No necesariamente, supongamos un proceso que necesite de muchas paginas tantas que no queparian en memoria sin usar el área de swap, y el proceso constantemente salta de paginas de memoria a paginas en swap, en ese caso abría una tasa elevada de fallos de pagina.

Explique para que sirven los algoritmos de reemplazo de páginas. Indique además en que métodos de administración de memoria se utilizan.
Empleados en los sistemas de mv donde hay intercambio, se utilizan para intercambiar paginas entre dos niveles de memoria cuando ya no hay espacio disponible en la memoria
Ejemplo de estos tenemos paginacion bajo demanda y segmentacion con paginacion bajo demanda

las técnicas de administración de memoria con Swapping son más veloces que las técnicas sin Swapping porque permiten incrementar el nivel de multiprogramación del sistema.
FALSO. Si bien es cierto que aumentan el nivel de multiprogramacion, tenemos demasiado overhead por el echo de los algoritmos de selección y remplazo de pagina, para administrar los espacios de momorias. Por otro lado tenemos un lento acceso a la memoria secundaria.

La interrupcion de clock es externa enmascarable
TRUE

22-La reubicabilidad de la información solo es importante en sistemas que utilizan Swapping.
Falso.
La reubicabilidad de direcciones hace referencia a la correspondencia entre las direcciones logicas y fisicas por ende es utilizado tanto en sistemas que utilizan swapping como los que no

23-El método de sustitución local produce menos fallos de página que el método global.
FALSE
No necesariamente y de echo suele tener mas fallos de pagianas que los locales. Lo que suele suceder es que con local los procesos solo pueden sustituir sobre los frames que le fueron asignados
encambio con global se podria tomar frames de otros.

24-La paginación multinivel es más lenta que la paginación pura o simple debido a que
requiere más accesos a las tablas MPT para encontrar los datos.
TRUE

25-El algoritmo LFU mantiene un contador por cada página del número de veces que fue
accedida.
TRUE

26-La tasa de fallo de página de un proceso es inversamente proporcional a la cantidad de
frames asignado.
Falso. Primero depende del algoritmo dado que por ejemplo algunos sufren del efecto belady)

27-En un SO que administra la memoria a través de particiones variables el grado de
multiprogramación es siempre fijo y tiene un máximo conocido.
FALSE
Porque depende de como los procesos activos vallan ocupando la memoria

Enumero las funciones del administrador de memoria.
Proteccion
Reubicabilidad
Gestion de espacio libre y ocupado

Definir las direcciones físicas y las direcciones lógicas.
Direcciones logicas: pertenecen al espacio de direccionamiento logico, y son las direcciones emitidas por el proceso en ejecucion
Direcciones Fisicas: pertenecen al espacio de direccionamiento fisico y son las direcciones reales.

La imagen de memoria del proceso se divide en porciones, todas del mismo tamaño.
FALSE

Depende de la tecnica de asignacion
Ejemplo esto no sucede en la segmentacion

31- ¿Para qué se utiliza la memoria virtual?
Aumentar el nivel de multiprogramacion. Y eliminar la restriccion de que el tamño maximo de un proceso se corresponda al area libre.

32-Con sustitución local el proceso solo está afectado por su propio paginado.
TRUE

Con la técnica de particiones variables el nivel de multiprogramación está limitado por el número de particiones.
FALSE

Una tasa de fallo de página elevada indica que nuestro sistema tiene demasiados procesos en ejecución.
FALSE
Pensar en asignacion estatica pocos frames a un proceso que utiliza muchas paginas en el area de swap

39-Las técnicas de paginación evitan fragmentación interna.
FALSE
Externa

Modulo 6: I/O

En un sistema con procesador DMA, la administración de I/O la realiza directamente el CPU mediante rutinas a interrupciones.

FALSE

La idea de DMA es la mínima intervención del CPU durante las operaciones del CPU, por lo que solo es interrumpido por el DMA al finalizar la operación I/O. Por lo tanto el cpu solo interviene en el comienzo(Solicitud CPU->DMA) y finalización de la operación (DMA->CPU).

El algoritmo de ordenamiento de la cola de listo es el que mejor rendimiento produce en un disco de estado solido (SSD).

FALSE

Los tiempos de acceso a los discos de estado solido es constante para cualquier dirección que se requiera.

Los dispositivos externos se conectan directamente al modulo genérico de I/O ya que de esta manera oculta al OS la complejidad del HW.

TRUE

Es una de los motivos.

Los disco se formatean con interleave mayor a cero para lograr que la cabeza se mueva lo menos posible.

FALSE

Compensa los tiempos de transferencia.

Las operaciones I/O las controla directamente cada dispositivo.

FALSE

Falso. El modulo de I/O se encarga del control.

1-Utilizar el algoritmo de brazo de disco óptimo permite disminuir el tiempo de latencia promedio de todas las peticiones.

FALSE

Primero el tiempo que mejoran los algoritmos de brazo es el tiempo de busqueda y no el de latencia. Segundo No existe, o almenos la biblografia de la catedra no nombra un algoritmo optimo

5-El interleave óptimo permite mejorar el tiempo de acceso promedio de todas las peticiones.

FALSO. El tiempo de acceso = tl + tb + ttransf. Por lo que si las lecturas son secuenciales mejora pero si no lo es el tiempo de acceso tambien estaria afectado por el tb

El interleave optimo seria aquel que permitiera leer todos los sectores de una pista con el menor numero de vueltas.

6-El tiempo de búsqueda promedio puede mejorarse si se utiliza el algoritmo de brazo de disco adecuado.

TRUE

El tiempo de busqueda es el tiempo que tarda el brazo en moverse a la pista indicada

8-La velocidad de transferencia de datos del DMA depende de la velocidad del bus.

FALSE

EN realidad depende de la tecnica que se utiliza. De todas formas la velocidad del bus si afectaria pero a todas las tecnicas por igual.

9-En entrada/salida programada, el módulo de E/S no ejecuta ningún tipo de acción para alertar al CPU.

TRUE

10-El algoritmo de planificación de brazo de disco SSTF puede provocar starvation.

TRUE

11-La técnica de DMA transparente logra una velocidad de transferencia muy elevada.

Falso.

Solo emplea los ciclos en los que el procesador no usa el bus de sistema.

19-La técnica de E/S por DMA se utiliza para quitarle carga de trabajo al procesador ya que este no debe ocuparse de trasferir datos hacia memoria central.

TRUE

¿Para qué sirven los algoritmos de planificación de brazo de disco? ¿Que se gana teniendo un buen algoritmo?

Los algoritmos de planificacion de disco buscan minimizar el tiempo de busqueda (tiempo empleado en posicionarse las cabezas en el cilindro que contendra el sector que buscamos)

38-La técnica por robo de ciclos es la más rápida pero es la que más tiene activa al procesador.

FALSE

Transparente porque solo toma los ciclos del procesador en los que no utiliza el bus

El módulo de DMA debe ser capaz de trabajar con ambas velocidades, la del bus del sistema y la del dispositivo.

Modulo 7: File System

La Estructura de FAT / Bit Vector ocupa más espacio que la estructura de I-Nodo/ lista de sectores libres.

FALSE

Lista de sectores Libres: No Existe. Existe Lista de Bloques libres que emplea un puntero a la primera porción de bloques libres contiguos que a su vez posee un ptr a la próxima.

FAT mínimamente se utiliza un puntero por bloque.

Bit Vector: utiliza un bit por bloque.

I-Nodo: depende de la implementación y independientemente se empleara un I-Nodo por archivo mínimamente.

Por su estructura, en un file system basado en FAT se puede acceder a los bloques de archivo en forma aleatoria.

FALSE

Porque lo que encontramos en la tabla de directorios es la ubicación del primer bloque, y luego cada bloque apunta a su próximo, por ende es imposible un acceso aleatorio.

Ejemplo: el archivo test ocupa los sectores 3, 5, 7.

Tabla de directorios (archivo, bloque inicio) = (test, 3)

Fat= b3->b5->b7-Null

Si quisiéramos acceder a un registro del bloque 5 si o si deberíamos pasar por el bloque 3.

La mejor forma de llevar la lista de bloques libres para un sistema de archivos donde se almacenan archivos de forma secuencial desfragmentada es la lista de bloques libres.

FALSE

Lista de bloques libres contigua seria mas eficiente porque es probable encontrar porciones contiguas.

Los FS que se manejan con el método de NTFS implementan la forma de llevar el espacio ocupado atraves de una tabla FAT.

FALSE

UTILIZA INDEXADO CON INODOS, PERO SI EL ARCHIVO ES PEQUEÑO PUEDE DIRECTAMENTE PONERLOS EN CATALOGO

La estructura de I-Nodos en un FILE SYSTEM tipo EXT3, se almacena en el área de archivos porque su Tamaño varía de acuerdo a la ocupación del disco.

FALSE

Los inodos se almacenan en la ilista y la ilista esta en catalogo y su tamaño es constante.

2-Una de las desventajas de la lista enlazada de bloques libres es el espacio utilizado por los punteros.

Falso en realidad se tiene un unico puntero para apuntar al primer bloque

El tamaño máximo de un archivo en un file system administrado por i-nodos ISAM depende del tamaño del bloque de disco.

Falso. Depende tambien del espacio que representen los punteros a guardar en los bloques de indereccion.

4-Con i-nodos ISAM, un archivo puede crecer tanto como bloques libres haya en el disco.

FALSO. Del tamaño de bloques y punteros utilizados.

7-La asignación contigua permite que el acceso directo a un bloque de un archivo sea más rápido.

12-Los inodos permiten administrar archivos de cualquier tamaño, siempre que haya lugar en el disco que los aloje.

Falso. Solo podran almacenarse archivos de acuerdo a la implementacion del inodo. En isam dependera del tamanio de los punteros y de los bloques.

13-El bit Vector ocupa menos espacio cuando el disco se encuentra poco ocupado.

FALSO

Porque el bit vector dedica un bit por cada bloque del disco el cual toma valor 1 para indicar que esta ocupado y 0 para indicar que esta libre

14-El tamaño máximo de un i-nodo depende del tamaño de bloque de disco.

FALSE

El tamaño de un inodo es fijo.

15-Con asignación contigua, el tamaño máximo de un archivo depende del tamaño del

bloque de disco.

FALSO

Dependera de la cantidad de bloques libre en el disco. Si no se compacta sera la mas grande, si se compacta sera el bloque libre encontrado.

16-La asignación enlazada permite tanto acceso secuencial como directo.

FALSE

Solo permite acceso secuencial recorriendo los punteros. Solo podremos acceder directamente al primes bloque de cada archivo

	17-Una manera de reducir la fragmentación es ampliar la capacidad de memoria RAM de un
	equipo.
	Falso.
	Compactacion o desfragmentacion en casos de fragmentacion externa.
	La fragmentacion interna no hay forma de solucionarla a no ser que se unse asignacion dinamica ajustando la porcion de memoria asignada a lo que se necesita.
	20-El concepto de File System es aplicable solamente a discos rígidos.
	FALSE
	se emplea en cualquier dispositivo de almacenaiento secundario
	37-El tamaño máximo de un archivo que utiliza asignación enlazada depende del tamaño del
	bloque de disco y del tamaño de los punteros.
	FALSE
	depende de la cantidad de bloques libres y el espacio desperdiciado en punteros
	Los accesos directos de MS Windows son analogos a los soft links de Unix

Modulo 8: Seguridad

La forma más utilizada para el almacenamiento de contraseñas es la encriptación con algoritmos reversibles.

FALSE

Sin un algoritmo es reversible implica una debilidad por lo tanto no es recomendable. Podría emplearse algoritmos no reversibles como RSA o implementaciones mediante hash.

La intercepción de un mensaje (sin modificarlo) por un intruso afecta a la disponibilidad del mismo.

FALSE

Falso. Al no modificarse y no haber interrupción lo único que se afecta es la privacidad.

El uso de mensajes encriptados y firmados con claves públicas y privadas garantizan solo la privacidad.

FALSE

Privacidad (Encriptado) + Autenticación (Firma).

Firma: Encriptados con la privada y des-encriptamos con la pública.

Ciframos: Encriptados con la pública y des-encriptamos con la privada.

Debemos hacer una composición de ambas funciones.

Cierto supermercado posee un sistema web sobre internet que permite a los clientes realizar sus compras, aunque no ofrece medios de pago on-line, es decir que una vez recibida la mercadería en su hogar, el cliente abona en efectivo o con tarjetas de compra/crédito vía un punto de venta con conexión 3G. Un empleado detectó por casualidad que el sistema web es sensible a ataques XSS, Sql Injection y ataques de tipo DoS (denegación de servicio). Es evidente que es necesario corregir el sitio web, pero ¿qué otras medidas sugeriría usted para mejorar la seguridad de éste servicio y del departamento de sistemas del supermercado en general?

Proteccion mediante un Firewall.

Un firewall sirve para proteger una red de ataques externos e internos.

FALSE

No puede proteger de ataque internos.

18-Los algoritmos de encripcion asimétricos utilizan dos claves, una para encriptar y otra para desencriptar.

TRUE

Con infraestructura de clave publica (PKI) y privada, se puede garantizar la confidencialidad, pero no la identidad.

FALSE

Depende en el orden en que se usen

Si usas Privada para des y publica para enc privacidad

Si usas Privada para enc y publica para desc identidad

Suponga que debe ofrecer asesoramiento sobre seguridad informatica a una PYME muy particular, pues se trata de una empresa que no posee conectividad de ningnun tipo a redes externas, ni siquiera a Internet. Es decir, que solo cuenta con una red local de PC's y servidores propios. ¿Recomendaria a esta empresa que utilice criptografia para el almacenamiento de datos sensibles o esto carece de sentido en un contexto asi? Siguiendo la logica ¿seria apropiado montar serivcios web seguros (https) o no seria necesario? Debe justificar su analisis.

Una solucion de almacenamiento que se comunica con FcOE se puede llevar acabo gracias a que el protocolo SCSI esta encapsulado, entre otros, en el protodolo TCP que le da confiabilidad a la conexion de los host (servidores) con el storage ya que tanto Ethernet como IP son ambos protocolos del mejor esfuerzo y en los mismos se pueden perder tramas o paquetes respectivamente

FALSE

Modulo: Virtualizacion

Los SO virtuales acceden directamente al HW solo cuando ejecutan instrucciones privilegiadas

Falso. Depende de varias cosas: Tipo de virtualizacion, tipo de hipervisor, arquitectura del procesador

un virtualizador tipo 1 (Baremetal) se instala sobre un sistema operativo de base.

Falso es sobre el HW que se instala.

El tipo de virtualizacion de un determinado hipervisor (tipo 1 o 2) depende de la instalacion con la que éste fue instalado.
No obstante, algunos hipervisores del mercado permiten el cambio de tipo (de tipo 1 a tipo 2 y viceversa)luego de ser instalado

FALSE

Creo que no se pueden porque el tipo dos se instala sobre un sistema operativo y el otro no

Modulo: storage

La configuración de RAID 5 es la más apta para sistemas transaccionales con tasa de escritura.

Si se debe asegurar las transacciones y un bajo costo el raid 5 es óptimo ahora si se prioriza la velocidad respecto a la seguridad con un raid 1+0 alcanzaríamos mejor performance.

La técnica de stripping aplicada en RAIDs consiste en escribir el mismo contenido en varios dispositivos simultáneamente.

Falso. Implicaria mirroring.

Un storage NAS es la configuración recomendada para usar como file server.

TRUE

Si porque permite limitar las consultas que se pueden hacer sobre los archivos. Ademas los comandos scsci correrian sobre el mismo servidor que tiene el poder. Por otro lado con los nas no necesitamos de hardware especial ni tratamiento del soft especial.

Al desarrollador de software de un sistema bancario le han pedido que desarrolle un modulo donde por cada trx que el cliente realiza sobre su cuenta bancaria la misma es escrita en una base de datos para fines de auditoria de seguridad. Luego de terminar el desarrollo del modulo y hacer las pruebas pertinentes, el desarrollador situa la base de datos que contiene dichas transacciones en un sistema de discos que trabaja en RAID 5 por que es el mejor nivel de RAID que se adapta a las caracteristicas de uso de la base de datos.

¿Que tipo de RAID usaria en un protocolo orientado a archivos? ¿Cual usaria en un protocolo orientado a caracteres?

No depende del protocolo el raid