-Explique por qué las llamadas al sistema deben realizar un cambio a modo kernel. Indique qué sucedería si la misma llamada se mantuviera en modo usuario durante toda su ejecución.

* Porque utilizan instrucciones que requieren que el bit de modo esté seteado en modo kernel, ya que acceden a hardware o bien a secciones de memoria que un proceso no puede (ni debe) acceder en modo usuario. Si la misma se mantuviera en modo usuario, se producirá un error (en forma de interrupción) a la hora de ejecutar alguna instrucción privilegiada

- Explique por qué un proceso suspendido no siempre está a la espera de un evento.

* Puede estar suspendido listo, lo cual significa que estuvo esperando un evento y mientras eso ocurría fue suspendido. Luego dicho evento se produjo, y por lo tanto el proceso quedó en estado suspendido/listo.

-Un Sistema Operativo bancario corre algunos grupos de procesos. A primera hora de la mañana, se corren procesos financieros que deben finalizar lo antes posible, ya que de sus resultados depende la compra de acciones y bonos durante el dia. Durante toda la jornada, se corren procesos que atienden “virtualmente” a los clientes, realizando pequeñas entradas-salidas ante cada pregunta que reciben. Explique qué algoritmo de planificación utilizaría, indicando configuración de colas, transiciones y al menos dos criterios para evaluar su funcionamiento.

* En principio alguna variante de colas multinivel. Idealmente, los procesos financieros deben estar en una cola de mayor prioridad, y las atenciones virtuales en una cola de menor. La primera cola probablemente utilice un FIFO, la segunda podría usar RR, para poder distribuir la ddatención de los clientes. No hay transiciones entre las colas, ya que la idea es separarlos. Dos criterios que podrían usarse para evaluar el funcionamiento son: Deadlines (sobre todo para la cola de mayor prioridad) y quizás tasa de procesamiento (para todos, cuantas más consultas se contesten, mejor será el algoritmo).

- Explique cómo funcionan los semáforos bloqueantes, indicando ventajas y desventajas. Explique  cómo implementa el Sistema Operativo la mutua exclusión sobre la “variable semáforo”.

* El semáforo bloqueante decrementa la variable semáforo con wait, y si el valor es < a cero bloquea al proceso. Lleva una cola de procesos bloqueados. La función signal la incrementa, y si corresponde, desbloquea un proceso de la cola. El sistema operativo puede implementar mutua exclusión, sobre el semáforo en sí, deshabilitando las interrupciones o bien con instrucciones de hw (por ej, test and set)

- Describa brevemente la estrategia de detección y recupero de deadlock (no es necesario describir  el    algoritmo de detección en sí mismo). Mencione al menos dos desventajas de aplicar dicha estrategia

* La estrategia consiste en no otorgar límite alguno en el inicio y solicitud de recursos de parte de los procesos. Agregado a eso, se ejecuta periódicamente un algoritmo de detección de deadlock sobre el estado actual, y en caso de encontrar alguno se intenta eliminar el mismo a través de alguna estrategia de recupero (finalización de proceso, expropiación de recurso, etc). Desventajas podrían ser el overhead que genera correr el algoritmo cada cierto periodo de tiempo, y también el hecho de que los procesos podrían verse finalizados abruptamente si fueron parte de un deadlock.

- Defina brevemente llamada al sistema (syscall). ¿Qué relación tiene con los modos de ejecución y las instrucciones privilegiadas?

* Herramienta utilizada por los procesos para solicitar servicios al sistema operativo que de otra forma le serían imposibles poder realizar, como el uso de recursos de hardware (leer disco por ejemplo) o de otra índole (crear un klt por ejemplo). Los procesos ejecutan en modo usuario, bajo el cual no se pueden ejecutar instrucciones privilegiadas. Al realizar una llamada al sistema, se cambia a modo kernel, y solamente en dicho modo se pueden ejecutar instrucciones privilegiadas.

-¿En qué situaciones convendría usar hilos de usuario en lugar de hilos de kernel? Mencione al menos dos atributos propios del TCB (Thread Control Block)

* Convendría usar ULT por sobre KLT cuando se requiera overhead al mínimo (debido a que no hay cambios de modo), o se requiera portabilidad del programa (dado que sobre bibliotecas estándares podria usarse en cualquier SO), o se requiera una planificación particular no provista por el sistema operativo. Dos atributos pueden ser el TID (thread id) y el contexto de ejecución (PC, flags, etc)

- Explique las implicancias de utilizar un planificador de corto plazo sin desalojo (non preemptive) en  sistemas operativos de tiempo compartido (o multitarea).

* El sistema operativo solo puede recuperar el control de la CPU cuando el proceso finaliza o se bloquea, por lo que un usuario/proceso podría nunca ejecutar si otro proceso nunca libera el procesador, ya sea por un error o porque deliberadamente no se bloquea nunca.

-Explique cómo funcionan los semáforos con espera activa, indicando ventajas y desventajas.

* Los mismos garantizan la mutua exclusión a través de alguna solución de software (algoritmo de peterson por ejemplo) o usando la instrucción de hardware (como test\_and\_set). Tienen la ventaja de sirven para sistemas multiprocesador sin generar overhead extra (como sí lo haría el deshabilitar las interrupciones) o no realizan cambios de contexto como si lo harian los bloqueantes. Tienen la desventaja de generar una pequeña carga de espera activa.

-En la estrategia de evasión de Deadlock mediante algoritmo del banquero, ¿Que significa estado  seguro?Podría el sistema quedar en estado inseguro ante alguna situación particular?

* Estado seguro significa que se tiene la certeza de que nunca ocurrirá un deadlock y que existe al menos una secuencia segura de finalización de los procesos. El sistema nunca podría quedar en estado inseguro, debido a que la estrategia consiste en no otorgar recursos que pudieran dejarlo en dicha situación.

- Indique en qué forma/s se realizan los cambios de modo (en ambos sentidos). Explique para qué son necesarios.

* Los cambios de modo se realizan modificando el bit de estado que contiene el psw. Si esta 0 esta en modo kernel. Si esta en 1 esta en modo usuario. Es necesario cambiar de modo cuando se produce una syscall ya que estas son las que llaman a las instrucciones privilegiadas y estas se pueden ejecutar a nivel kernel. De forma contraria, la gestión de los hilos usuario deben ser gestionados por una biblioteca y el sistema operativo no conoce de ellos.

-Relacione brevemente los siguientes conceptos: PCB - Proceso - Imagen - Memoria. Muestre un ejemplo simple en pseudocódigo y muestre cada sección en que parte de la imagen del proceso se encontraría.

* El PCB es el bloque de control del proceso, que almacena en el el pid, el estado del proceso, el program counter, registros de cpu, información de planificación , de gestión de memoria, contable, de E/S y punteros. Cuando el proceso se encuentra en memoria en la cola de ready, la imagen del proceso es un parte de código que contiene este mismo, una parte de datos que contiene las variables globales, una parte de stack que contiene variables locales, argumentos y retornos, y un parte de heap que contiene las estructuras dinámicas.

Int Global = 1;

Int main(int argc, char\* argv)

argv = malloc(10);

argc = 3

return global + 1;

- En un instante X ocurre una interrupción por n de quantum y otra por n de E/S. Explique qué ocurriría a bajo nivel, justicando qué proceso será elegido para ejecutar y por qué. Considere dicha  situación por un lado para RR y por otro para VRR.

* Cuando retorna el proceso a la cola de ready en RR, el proceso que realizo la E/S vuelve a la misma cola que la del quantum, mientras que en VRR el proceso que realiza una E/S se aloja en una cola con mayor prioridad que la de quantum, con un nuevo quantum que se calcula con el quantum original menos la ráfaga ejecutada anteriormente.

-V o F. Justque

a. Los semáforos mutex son la única técnica que se puede utilizar para garantizar mutua  exclusión de un recurso compartido.

* Falso, también se puede realizar impidiendo las interrupciones, o con las instrucciones atomicas test and set.

b. La situación en la cual el resultado de la ejecución de un conjunto de procesos depende del orden en el que los mismos se ejecutan es llamada starvation.

* Falso, esto mencionado anteriormente es condición de carrera. Starvation es que un proceso nunca ingresa a la cpu por falta de recursos o por una mala planificación.

-En un sistema en el que no se utiliza ni evasión ni prevención de deadlocks, indique cómo sería el algoritmo que utilizaría para detectar la existencia de deadlocks. Proponga dos opciones para  solucionarlos comentando los problemas y ventajas de cada una.

* Se corre un algoritmo en intervalos regulares en busca de deadlocks, (actua como el algoritmo del banquero pero se fija que pueda realizar todas las peticiones en ese momento, no es una simulación, se esta haciendo en el momento), en caso de encontrar deadlock los trata. La forma de tratarlo es o matar a todos los procesos que me generen deadlock pero en este caso estaría perdiendo a muchos procesos, o voy desalojando de a un proceso para obtener sus recursos, pero esto presenta la desventaja de que se puede elegir siempre al mismo proceso como victima y sufre starvation.

- Explique por qué los semáforos necesitan, como parte de su implementación, utilizar llamadas al sistema

* Utilizan syscalls porque sus operaciones atomicas wait() and signal() utilizan syscalls. Si tienen espera activa para modificar la variable compartida usan testAndSet y swap y estas llaman syscalls. Si son semáforos bloqueantes block y wake up llaman syscalls tambien.
  + ¿Por qué un estado inseguro podría o no generar deadlock? Muestre una situación en las cual a partir  de un estado inseguro (resultado de la simulación de una asignación) ocurriría deadlock y otra en la que no. Asuma que luego de determinar que el estado es inseguro, no se utiliza más el algoritmo del banquero.

En el algoritmo del banquero se analiza el peor de los casos de asignación de recursos y si podría generar algún deadlock, el estado es inseguro, pero si los procesos se ejecutan en un orden distinto podría no generarse ese dealock.

- ¿Por qué y para qué se utiliza el cálculo de la media exponencial? ¿En qué tipo de algoritmos de  planicación sería útil usarlo? Analice los componentes que intervienen en dicho cálculo.

* Se utiliza para estimar, mediante la ráfaga anterior ejecutada y la estimación anterior, cuanto va a tomar ejecutar la ráfaga siguiente y elegir que proceso ejecutar. Además de los componentes mencionados anteriormente, hay un coeficiente de estimación, que según el valor que se le de, actua priorizando mayormente las ráfagas anteriores, o en otro caso las estimaciones anteriores. Seria útil en el algoritmo SJF y por prioridades, ya que se decide mediente un numero cual va a ser el próximo proceso a ejecutar.

- ¿Qué consideraciones tendría a la hora de usar semáforos si se utilizan KLTs o ULTs? Indique al menos 2

* En caso de usar KLTs, no tengo que realizar un cambio de modo ya que los semáforos utlizian syscalls para llamar a instucciones privilegiadas que se ejecutan a nivel kernel. En ULTs si se bloquea uno se bloquean todos, en ult no tenes cambio contexto.

- Explique el concepto de inanición(dando ejemplos) aplicado a:

* Starvation es que un proceso nunca ingresa a la cpu por falta de recursos o por una mala planificación.

Sincronización. Cuando se realiza un wait y nunca se realiza un sigan con un semáforo = 0

Planicación. En el algoritmo SJF, cuando un proceso o hilo ejecuta una ráfaga muy grande

Deadlock. Se produce una espera circular, y los procesos nunca obtienen los recursos que necesitan para ejecutar.

- ¿Cual es la diferencia entre una system call y una función “wrapper” con el mismo propósito? ¿Cuándo  usaría cada una?

* Una syscall es una llamada a sistema que puede ser diferente en cada sistema operativo, mientras que una función wrapper es una función que oculta los detalles algorítmicos de como realiza esta operación y no depende de las instrucciones de un sistema operativo en particular, por lo tanto son portables y mas declarativas. En el caso de fopen utiliza una función wrapper ya que no me interesa como lo hace en cada sistema, si no que me abra el archivo. En cambio, en una interrupción por entrada salida si ultilizaria una syscall.

- Explique porqué es importante encontrar el valor adecuado del quantum en el algoritmo RR. ¿Cómo se administra el quantum en VRR?

* Es importante encontrar un valor adecuado ya que si el valor es muy corto se tienen que hacer cambios de contexto todo el tiempo y esto produce overhead, y si el valor es muy grande es como utilizar un algoritmo FIFO. En VRR el quantum se administra de igual manera en una cola de quantum, pero cuando un proceso entra en E/S se aloja en una cola prioritaria con un un nuevo quantum que se basa en la resta entre el quatum original y la ultima ráfaga ejecutada.

- ¿Podría un proceso quedar en starvation luego de llamar a la función wait(semáforo)? Por otro lado,  ¿Sería posible implementar semáforos en una biblioteca de usuario, sin que intervenga el sistema  operativo?

* Podría quedar en starvation si nunca se realiza el signal del semáforo si es bloqueante, o si no es bloqueante siempre va a estar en espera activa. No seria posible implementarlos sin que intervenga el sistema operativo ya que los semáforos utilizan operaciones atomicas como wait y signal que utilizan syscalls que llaman a testAndset que se ejecutan en modo kernel.

- Explique en qué consiste la técnica de prevención de Deadlock. Indique tres maneras posibles de  aplicarla.  Indique ventajas y desventajas respecto a la evasión de deadlock

* Se busca prevenir alguna de las 4 condiciones antes de que se ejecuten los procesos para que no se genere deadlock.

-Espera circular: una función donde asigno un valor a cada recurso y no se puede pedir un valor que sea mayor al que ya pedí.

-Sin desalojo: para evitarlo si un proceso pide un recurso que no le puedo dar le desalojo todos los recursos que tiene (no es nada efectiva).

-Retención y espera: que el proceso te pida el máximo y si no se lo puedo dar no le doy nada.

Ventajas, ningún proceso va a entrar en deadlock pues las condiciones están solucionadas de antemano. Desventaja, es ineficiente, produce mucho overhead.

- Explique los modos de ejecución de una CPU y cómo, a partir de estos, se implementa la seguridad

* La CPU puede ejecutar en modo kernel que es el modo en el cual se realizan las llamadas a sistema que ejecutan las instrucciones privilegiadas del sistema operativo, mientras que en modo usuario son ejecutadas los procesos y las instrucciones a nivel usuario que estas son manejadas por una biblioteca. En cuestión de seguridad. Que el usuario no pueda acceder a lo que no tiene permiso, por ej el hw.

- Explique la utilidad de los PCBs en un entorno que soporta KLTs

* Los KLTs pueden cambiar de contexto, tiene que guardar todo en el pcb del estado del KLT. Los pcbs permiten cambiar de contexto e ir ejecutando procesos en paralelo. TCB cada hilo tiene el suyo propio, para almacenar los datos que no comparte cada hilo.

-V o F Justique. En las soluciones de hardware para la Exclusión Mutua…

a. Deshabilitar las interrupciones: eficiente en sistemas de multiprocesamiento pero su uso requiere espera activa.

* Falso, es ineficiente en sistemas de multiprocesamiento pq tenes que desahibilitar en todo los procesadores la interrup

b. Instrucciones especiales: al ser atómicas evitan la espera activa y condiciones de carrera.

* Falso, ya que emplean espera activa.

- Explique la utilidad del planicador de largo plazo. Indique qué estados del proceso están   relacionados con este planicador y qué función cumplen

* El planificador a largo plazo es el encargado de seleccionar que procesos pasaran de la cola de nuevos a la cola de listos, cargándolos en memoria para su próxima ejecución. En la cola de nuevos se ponen todos los procesos que van ingresando al sistema y se van a querer ejecutar en algún momento, mientra que en la cola de ready se encuentran los procesos que el planificaodr de la cpu puede elegir para su próxima ejecución.

- Explique qué sucedería si un proceso ejecuta una instrucción privilegiada. ¿Cuál sería la forma correcta de hacer esto?

* Si un proceso ejecuta una instrucción privilegiada aborta porque esta en modo usuario. se debería llamar a una syscall, guardar el pcb y el psw del proceso, cambiar de modo a modo kernel y ejecutar la instruccion.

-Explique al menos tres problemas que tiene el algoritmo SJF, indicando aluna posible solución para cada uno.

* Starvation, un proceso nunca se ejecuta por tener una ráfaga muy alta. Se soluciona agregando un response ratio.

Viene un proceso con mucha prioridad pero no se ejecuta porque tiene una ráfaga grande. Se utiliza un algoritmo por prioridades.

No puedo saber cuanto va a durar la primer ráfaga. Se soluciona con la formula de e stimacion

- Compare los conceptos de Proceso, KLT y ULT en base a los siguientescriterios: Seguridad, Multiprocesamiento, portabilidad.

* En cuestión de seguridad los mas seguros son los procesos ya que un proceso con otro no comparte datos los cuales no puede acceder otro proceso, en cuestión de multiprocesamiento los indicados son los KLTs ya que pueden ejecutarse varios a la vez, y si uno de estos se bloque, otro puede continuar con su ejecución, y por ultimo en portabilidad los indicados son los ULTs ya que estos no son conocidos por el sistema operativo si no por la biblioteca, la cual cambiando de sistema operativo esta misma va a funcionar igualmente.

- Indique las diferentes formas de lograr mutua exclusión por hardware .¿Cuándo utilizaría cada una?

* Las formas son deshabilitando las interrupciones, pero no son eficientes para sistemas de multiprocesamiento. Funciones basadas en instrucciones atomicas como test and set y swap, que tiene como desventaja que utilizan espera activa pero como ventaja que es un función que te da el sistema operativo.

- Explique por qué un set de procesos que sufre condiciones de carrera no podría entrar nunca en deadlock

* Para que haya condición de carrera todos los procesos deben utilizar un mismo único recurso, mientras que en deadlock se debe depender de como minimo dos recursos

-Describa los diferentes tipos de interrupciones y dé al menos 2 ejemplos de cada uno, relacionando cada ejemplo con alguna de las funciones enumeradas.

* Las interrupciones de HW y las excepciones son los dos tipos de interrupciones que tenemos. Ejemplos de interrupciones: fin de quantum (se usa en algoritmos de planificación, cuando ejecutan los procesos), fin de entrada/salida (se relaciona con algunos recursos, que informan que una operación finalizó, y son parte de la interfaz con el HW). Ejemplos de excepciones: división por cero (durante la ejecución de un proceso), acceso inválido (por ej, al querer escribir la imagen de otro proceso no podremos hacerlo, por cuestiones de seguridad).

-Escriba el pseudocódigo de dos hilos cooperativos, incluyendo (y señalando) dónde se referencia su stack, su sección de datos y su heap.

int GLOBAL = 1; ← Data

void main() { ← Stack

char\* msj = pedir\_mem(10); ← Heap

crear(hilo\_1, msj);

crear(hilo\_2, msj);

}

void hilo\_1(char\* msj) {

concat(msj, “hilo1”);

}

void hilo\_2(char\* msj) {

concat(msj, “hilo2”);

}

-Indique qué similitudes y diferencias hay entre el algoritmo del banquero y el de detección de deadlocks. ¿Por qué se dice que la técnica de evasión es “pesimista”?

* Ambos algoritmos consideran una matriz de recursos asignados y vectores de recursos totales y disponibles. Ambos también realizan una simulación de asignación de recursos, la diferencia radica que en el caso de detección, el resultado dice si en dicho momento el sistema se encuentra o no en deadlock mientras que con el algoritmo del banquero se determina si el sistema está en estado seguro. Otra diferencia es que el algoritmo del banquero utiliza las peticiones máximas, que pueden ocurrir o no, mientras el de detección sólo conoce peticiones que se realizan en un momento en particular.

La técnica de evasión es pesimista ya que considera el peor de los casos: ante cada petición, sólo la satisface inmediatamente si la misma deja al sistema en un estado que aún si todos los procesos piden lo máximo que podrían pedir, se pueden asignar correctamente los recursos (en una secuencia segura).

-En un web browser, ¿Qué ventajas traería usar Procesos por sobre Hilos? ¿Convendría usar un proceso por cada pestaña abierta o uno por cada dominio (ej: www.utn.so ) visitado?

* Los procesos son mejores en términos de estabilidad y seguridad. Para el caso del web browser, podemos usar un proceso por pestaña si nos enfocamos en la estabilidad (si se cuelga una pestaña no se cuelgan las otras) o bien un proceso por dominio si nos enfocamos en la seguridad (esperamos que www.utn.so/faq no tenga problemas de seguridad con www.utn.so/campus por ej).

-Explique detalladamente las operaciones implicadas en un process switch ocurrido al terminar el quantum

Cuando finaliza el quantum:

1. Se lanza una interrupción por fin de quantum a nivel de HW

2. Al finalizar la instrucción actual se chequean si existen interrupciones pendientes.

3. Se cambia de modo U -> K

4. Se guarda el PC y el PSW (estado del proceso actual)

5. Se determina a qué se debe la interrupción. Se identifica cuál es el interrupt handler adecuado y se carga su instrucción en el PC

6. Se guarda el resto del estado del proceso actual y se lo coloca al final de la cola ready (se modifica su estado de running -> ready)

7. El planificador de corto plazo elige al siguiente proceso a ejecutar

8. Se pasa al proceso a modo running

9. Setea el contexto del procesador con los valores del nuevo proceso. Dentro

de ellos el PSW ( eso hace pasaje de K -> U ) y el PC

10. La siguiente instrucción a ejecutar es la del nuevo proceso