Conteste verdadero o falso y justifique su respuesta:

1. Cuando se presiona *Enter* en una interfase tipo CLI se produce una interrupción de software.   
     
   Falso. Las interrupciones de software ocurren cuando el programa en ejecución realiza una syscall, por ejemplo, para realizar una petición de E/S.   
     
   Al presionar enter se produce sí o sí una interrupción de hardware externo, ya que el teclado es un dispositivo periférico, y todas las entradas que se realicen con el teclado *deben* ser leídas y atendidas.
2. Si se construye un sistema basado en hilos KLT, podemos afirmar que las instrucciones de dicho sistema se ejecutarán más rápido que si tuviéramos una solución sin hilos.   
     
   Falso. Las instrucciones no se ejecutarán más rápido. La rapidez de ejecución de una instrucción es la misma, no depende de si se implementan hilos o no.   
     
   Lo que sí, si el sistema cuenta con multiprocesamiento, dos hilos KLT podrían estar siendo ejecutados en paralelo, pudiendo así tener dos hilos del mismo proceso avanzando en sus instrucciones al mismo tiempo, es decir, se estaría duplicando la cantidad de progreso. Es por esto que una aplicación que bien diseñada con hilos KLT en un sistema multiprocesador puede ser más rápida y eficiente que una sola aplicación que cuente con un solo hilo de ejecución.
3. El PCB es creado por el planificador de corto plazo cuando el proceso llega al sistema.   
     
   Falso. El PCB es creado por el planificador de largo plazo, ya que este es el planificador encargado de crear al proceso y, por lo tanto, de crear el PCB correspondiente al mismo. La creación del PCB debe hacerse en el momento en que un proceso/trabajo nuevo llega al sistema, y no en un momento posterior, como lo haría el planificador a corto plazo.   
     
   Además, no es parte del planificador de corto plazo crear PCBs, sino decidir qué proceso se ejecutará en este momento.
4. Los hilos KLT pueden implementarse en un SO multiprocesador.  
     
   Verdadero. Los hilos KLT pueden implementarse en sistemas multiprocesador o monoprocesador. El problema de implementarlos en un sistema monoprocesador es que se perdería la ventaja que tienen los hilos KLT (capacidad de ejecutar dos o más hilos en paralelo, uno en cada procesador), pero esto no significa que la implementación de los mismos esté restringida a SO que soporten multiprocesamiento.
5. Cada vez que el sistema operativo recibe una interrupción de hardware, se produce un process switch.  
     
   Falso. No, no se genera un process switch cada vez que se recibe una interrupción de hardware externo. Lo que sí se produce sin lugar a dudas es un context switch, ya que debe guardarse todo el estado actual de ejecución del proceso que tenía alojado el procesador, para poder luego retomar su ejecución luego de haber atendido la interrupción.  
     
   De todas formas, esto depende de cómo esté desarrollado el algoritmo de planificación a corto plazo. Se podría implementar un algoritmo que, cada vez que se reciba una IHE, el mismo haga un cambio de proceso. ¿Sería eficiente? No sé.
6. En cierta empresa se dispone de un equipo con un único microprocesador y sobre él ejecuta un software fuertemente orientado al cálculo numérico. Este software demora varios minutos en realizar cálculos y recién al finalizar muestra un informe por pantalla. ¿Qué técnica de programación considera que es la más adecuada para este tipo? ¿Trabajaría con hilos (ULT/KLT), con procesos pesados, o simplemente con código secuencial sin hacer uso de la multiprogramación? Considere que el objetivo es optimizar el tiempo total de respuesta de este sistema.   
     
   Depende de si los cálculos no dependen del resultado anterior. Es decir, si el programa tiene una depende continuamente de la resolución del anterior cálculo para proseguir, y si ese mismo anterior cálculo NO puede ser dividido en cálculos menores…   
     
   Pero en caso de que los cálculos puedan dividirse y el programa no dependa tanto de los resultados anteriores, una buena manera de agilizar el proceso sería utilizando hilos. Al estar nuestro sistema limitado por un único microprocesador, utilizar hilos KLT no sería una buena idea, ya que no poseeríamos la ventaja del paralelismo, pero sí tendríamos las desventajas (cada hilo KLT es como un proceso pesado, lo que implica realizar un context switch, lo que implica overhead…).   
     
   Considero que la técnica de programación más adecuada sería la implementación de hilos ULT que trabajen en modo equipo de trabajo, ya que esta forma de trabajo resulta más eficiente para el cálculo numérico.
7. Explique en qué consiste un context switch. ¿En qué difiere con un process switch?  
     
   Un context switch consiste en guardar el contexto de ejecución de un proceso para poder cambiarlo por otro contexto de ejecución diferente. Es decir, se guardan los datos de ejecución del proceso actual (registros del procesador, program counter, etc.) y se recupera/busca el contexto de ejecución del proceso o rutina que utilizará el procesador a continuación.  
     
   La diferencia entre un context switch y un process switch es que el process switch involucra siempre un intercambio de procesos. Sin embargo, no siempre que hay un context switch hay un intercambio de procesos.   
     
   Por ejemplo, puede producirse una interrupción de hardware externa mientras se ejecuta un proceso. Debe realizarse un context switch, para poder ejecutar la rutina de atención de la interrupción. Pero esto no significa que el proceso que estaba ejecutándose sea cambiado por otro. Cuando finalice la rutina de atención, el proceso seguirá con su ejecución con normalidad.