TEMA 2 – Procesos y Hebras

Relación de cuestiones y problemas

1. Cuestiones generales sobre procesos y asignación de CPU:
   - ¿Cuáles son los motivos que pueden llevar a la creación de un proceso?
   - ¿Es necesario que lo último que haga todo proceso antes de finalizar sea una llamada al sistema para finalizar de forma explícita, por ejemplo `exit()`?
   - Cuando un proceso pasa a estado "BLOQUEADO", ¿Quién se encarga de cambiar el valor de su estado en el descriptor de proceso o PCB?
   - ¿Qué debería hacer cualquier planificador a corto plazo cuando es invocado pero no hay ningún proceso en la cola de ejecutables?
   - ¿Qué algoritmos de planificación quedan descartados para ser utilizados en sistemas de tiempo compartido?

2. Cuestiones sobre el modelo de procesos extendido:
   - ¿Qué pasos debe llevar a cabo un SO para poder pasar un proceso de reciente creación de estado "NUEVO" a estado "LISTO"?
   - ¿Qué pasos debe llevar a cabo un SO para poder pasar un proceso ejecutándose en CPU a estado "FINALIZADO"?
   - Hemos explicado en clase que la función `context_switch()` realiza siempre dos funcionalidades y que además es necesario que el *kernel* la llame siempre cuando el proceso en ejecución pasa a estado "FINALIZADO" o "BLOQUEADO". ¿Qué funcionalidades debe realizar y en qué funciones del SO se llama a esta función?
   - Indique el motivo de la aparición de los estados "SUSPENDIDO-BLOQUEADO" y "SUSPENDIDO-LISTO" en el modelo de procesos extendido.

3. ¿Tiene sentido mantener ordenada por prioridades la cola de procesos bloqueados? Si lo tuviera, ¿en qué casos sería útil hacerlo? Piense en la cola de un planificador de E/S, por ejemplo el de HDD, y en la cola de bloqueados en espera del evento "Fin E/S HDD".

4. Explique las diferentes formas que tiene el kernel de ejecutarse en relación al contexto de un proceso y al modo de ejecución del procesador.

5. Responda a las siguientes cuestiones relacionadas con el concepto de hebra:
   - ¿Qué elementos de información es imprescindible que contenga una estructura de datos que permita gestionar hebras en un *kernel* de SO? Describa las estructuras `task_t` y la `thread_t`.
   - En una implementación de hebras con una biblioteca de usuario en la cual cada hebra de usuario tiene una correspondencia N:1 con una hebra kernel, ¿Qué ocurre con la tarea si se realiza una llamada al sistema bloqueante, por ejemplo `read()`?
   - ¿Qué ocurriría con la llamada al sistema `read()` con respecto a la tarea de la pregunta anterior si la correspondencia entre hebras usuario y hebras kernel fuese 1:1?

6. ¿Puede el procesador manejar una interrupción mientras está ejecutando un proceso sin hacer `context_switch()` si la política de planificación que utilizamos es no apropiativa? ¿Y si es apropiativa?

7. Suponga que es responsable de diseñar e implementar un SO que va a utilizar una política de planificación apropiativa (*preemptive*). Suponiendo que el sistema ya funciona perfectamente con multiprogramación pura y que tenemos implementada la función `Planif_CPU()`, ¿qué otras partes del SO habría que modificar para implementar tal sistema? Escriba el código que habría que incorporar a dichas partes para implementar apropiación (*preemption*).

8. Para cada una de las siguientes llamadas al sistema explique si su procesamiento por parte del SO requiere la invocación del planificador a corto plazo (`Planif_CPU()`):
   - Crear un proceso, `fork()`.
   - Abortar un proceso, es decir, terminarlo forzosamente, `abort()`.
   - Bloquear (suspender) un proceso, `read()` o `wait()`.
   - Desbloquear (reanudar) un proceso, `RSI` o `exit()` (complementarias a las del caso anterior).
   - Modificar la prioridad de un proceso.

9. En el algoritmo de planificación FCFS, el índice de penalización, (M+r)/r, ¿es creciente, decreciente o constante respecto a r (ráfaga de CPU: tiempo de servicio de CPU requerido por un proceso)? Justifique su respuesta.

10. Sea un sistema multiprogramado que utiliza el algoritmo Por Turnos (Round-Robin, RR). Sea S el tiempo que tarda el despachador en cada cambio de contexto. ¿Cuál debe ser el valor de quantum Q para que el porcentaje de uso de la CPU por los procesos de usuario sea del 80%?

11. Para la siguiente tabla que especifica una determinada configuración de procesos, tiempos de llegada a cola de listos y ráfagas de CPU; responda a las siguientes preguntas y analice los resultados:

Proceso	Tiempo de llegada	Ráfaga CPU
A	4	1
B	0	5
C	1	4
D	8	3
E	12	2

   - FCFS. Tiempo medio de respuesta, tiempo medio de espera y penalización.
   - SJF (ráfaga estimada coincide con ráfaga real). Tiempo medio de respuesta, tiempo medio de espera y penalización.
   - SRTF (ráfaga estimada coincide con ráfaga real). Tiempo medio de respuesta, tiempo medio de espera y penalización.
   - RR (q=1). Tiempo medio de respuesta, tiempo medio de espera y penalización.
   - RR (q=4). Tiempo medio de respuesta, tiempo medio de espera y penalización.

12. Utilizando los datos de la tabla del ejercicio anterior dibuje el diagrama de ocupación de CPU para el caso de un sistema que utiliza un algoritmo de colas múltiples con realimentación con las siguientes colas:

Cola	Prioridad	Quantum
1	1	1
2	2	2
3	3	4

Tenga en cuenta las siguientes suposiciones:

- Todos los procesos inicialmente entran en la cola de mayor prioridad (menor valor numérico).
- Cada cola se gestiona mediante la política RR y la política de planificación entre colas es por prioridades no apropiativa.
- Un proceso en la cola i pasa a la cola i+1 si consume un quantum completo sin bloquearse.
- Cuando un proceso llega a la cola de menor prioridad, permanece en ella hasta que finaliza.