Sistemas Operativos
1° Parcial 1C2023 - TT - Resolución

<u>Aclaración</u>: La mayoría de las preguntas o ejercicios no tienen una única solución. Por lo tanto, si una solución particular no es similar a la expuesta aquí, no significa necesariamente que la misma sea incorrecta. Ante cualquier duda, consultar con el/la docente del curso.

Teoría

1. El ciclo de ejecución de una instrucción está formado por las etapas de Fetch (traer instrucción), Decode (decodificar y validar instrucción) y Execute (ejecutar instrucción). Luego como 4ta etapa se realiza la verificación de Interrupciones pendientes. Un ejemplo de una interrupción provista por la cpu (también conocida como interrupción de software) sería la generada por una división por cero. Un ejemplo de una generada de forma externa sería una de I/O (también conocida como interrupción de hardware)

2. Sí, sería posible que se crearan todas esas entidades. El resultado sería:
   Proceso 1 (Variable Global compartida entre todos los hilos de Proceso 1)
   \- KLT1: {ULT1, ULT_NUEVO]
   \- KLT_NUEVO
   Proceso 2 (hijo de Proceso 1)

3. Los planificadores que inciden sobre el grado de multiprogramación son el de largo plazo, admitiendo y gestionando la finalización de procesos en el sistema y el de mediano plazo, suspendiendo y restableciendo procesos. Es importante que el planificador de corto plazo posea el menor overhead posible ya que es el más frecuentemente se ejecuta.

4.
   a. Falso: Es necesario, además, que exista la posibilidad de concurrencia para que sea un problema.
   b. Falso: Los hilos de kernel también pueden sufrir condición de carrera si comparten recursos.

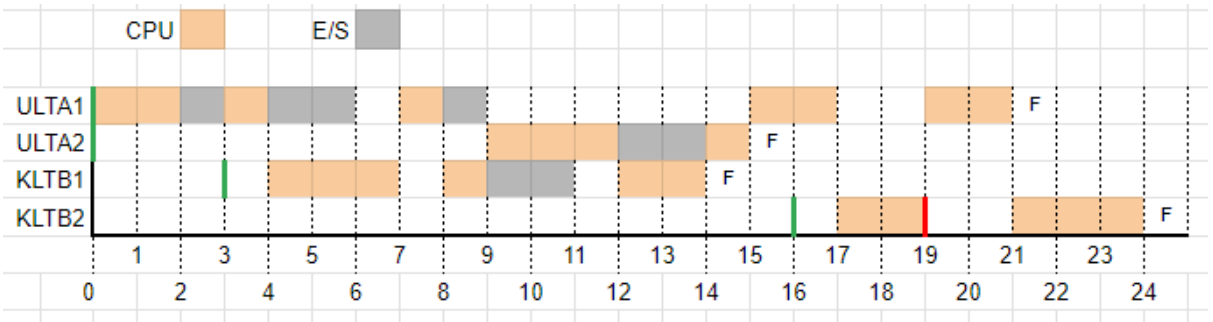
5. .Las cuatro condiciones son:
   1) Mutua exclusión: Provoca que los procesos se bloqueen.
   2) Retención y espera: Implica que existan recursos asignados a un proceso y que quede a la espera de otro proceso.
   3) Sin desalojo de recursos: El SO no desaloja los recursos asignados a los procesos.
   4) Espera Circular: Se produce cuando en un conjunto de procesos, cada proceso tiene asignado un recurso requerido por el siguiente proceso del grupo, y a su vez el último proceso tiene un recurso requerido por el primero.

Modo de prevenir (sólo se debe mencionar dos):
-Mutua exclusión: Permitir que los procesos accedan simultáneamente a una sección crítica. Depende del recurso y su uso.
-Retención y espera: Solicitar todos los recursos que se utilizar al comienzo de la ejecución o solicitar un recurso, utilizarlo y liberarlo antes de solicitar otro.
-Sin desalojo: Que el sistema operativo pueda desalojar recursos asignados a un proceso para asignarlos a otro. No puede utilizarse para cualquier recurso. Sólo para los que pueden volver a un estado anterior o guardar su estado.
-Espera circular: Asignar un orden a los recursos y asegurar que los procesos los soliciten sin alterar el orden.

Práctica

1.

a)



En el instante 19, KLTB2 realiza YIELD y vuelve a la cola de listos.

b)
Interrupciones por finalización de E/S: 3, 6, 9, 11, 14.
Interrupción de Reloj: 7, 17

c) En caso que la biblioteca utilice la técnica de jacketing, KLTB1 ejecutaría en el instante 6. No lo hace en el instante 3 debido a que el proceso A arriba a Listos por fin de quantum.

2.

SLOTS_LISTA = 30, MUTEX = 1, ELEM_LISTA = 0, RESP = 0, TURNOA = 1, TURNOB = 0

Alumno (N instancias)	Validador (N instancias)	Ranking A (1 instancia)	Ranking B (1 instancia)

P = recibir_pregunta()
WAIT(SLOTS_LISTA)
WAIT(MUTEX)
responder(P, LISTA)
SIGNAL(MUTEX)
SIGNAL(ELEM_LISTA)
	while(true){
	WAIT(ELEM_LISTA)
	WAIT(MUTEX)
	R = retirar(LISTA)
	SIGNAL(MUTEX)
	validar(R)
	enviar_a_ranking()
	SIGNAL(RESP)
	}
		while(true){
		WAIT(TURNOA)
		WAIT(RESP) x 20
		armar_ranking()
		SIGNAL(TURNOB)
		}
			while(true){
			WAIT(TURNOB)
			WAIT(RESP) X 20
			armar_ranking()
			SIGNAL(TURNOA)
			}

3.

Peticiones máximas

	R1	R2	R3	R4
P1	2	2	2	1
P2	3	3	2	2
P3	2	1	2	2
P4	3	2	2	4
P5	3	1	1	2

Recursos asignados

	R1	R2	R3	R4
P1	2	1	1	1
P2	1	1	1	1
P3	2	0	1	2
P4	0	0	2	1
P5	1	1	1	0
	6	3	6	5

Recursos totales

R1	R2	R3	R4
7	5	8	6

Necesidad

	R1	R2	R3	R4
P1	0	1	1	0
P2	2	2	1	1
P3	0	1	1	0
P4	3	2	0	3
P5	2	0	0	2

Recursos Disponibles

R1	R2	R3	R4
1	2	2	1

a)

1. **P2**: (0, 1, 1, 0)

Disponibles	1	2	2	1
Pedido -	0	1	1	0
Nuevos disponibles	1	1	1	1
Ejecuto y finalizo P3	2	0	1	2
Nuevos disponibles	3	1	2	3
Ejecuto y finalizo P1	2	1	1	1
Nuevos disponibles	5	2	3	4

Puedo finalizar a todos -> Estado seguro -> asigno.

2. **P1**: (0, 1, 1, 0)

Disponibles	1	1	1	1
Pedido -	0	1	1	0
Nuevos disponibles	1	0	0	1
Ejecuto y finalizo P1	2	2	2	1
Nuevos disponibles	3	2	2	2
Ejecuto y finalizo P3	2	0	1	2
Nuevos disponibles	5	2	3	4

Puedo finalizar a todos -> Estado seguro -> asigno.

3. **P1**: (1, 0, 0, 0)

Disponibles	1	0	0	1
Pedido -	1	0	0	0
Nuevos disponibles	0	0	0	1
Ejecuto y finalizo P1	2	2	2	1
Nuevos disponibles	2	2	2	2
Ejecuto y finalizo P3	2	0	1	2
Nuevos disponibles	4	2	3	4

Puedo finalizar a todos -> Estado seguro -> asigno.

b) Cualquier pedido que sea válido y supere la matriz de disponibles (0,0,0,1) no podrá ser satisfecho inmediatamente por falta de recursos en el sistema.