Trabajo Práctico Nº 4: Deadlocks y programación concurrente

1. ¿Qué se entiende por deadlock? Enumere las condiciones para que ocurra. De un ejemplo de deadlock fuera de los sistemas computacionales y analícelo/relaciónelo con las condiciones anteriores.

2. Responda las siguientes preguntas:

   (a) Una computadora tiene 6 drives de cinta, con n procesos compitiendo por ellos. Cada proceso puede necesitar dos drives. ¿Para qué valor de n el sistema está libre de deadlocks? Relacione el caso para n+1 con las condiciones de deadlock.

   (b) Considere un sistema con 4 recursos del mismo tipo compartidos por tres procesos, cada uno de los cuales requiere a lo sumo dos recursos. ¿En qué estado se encuentra el sistema? ¿Por qué?

3. En un sistema bancario existen cientos de procesos idénticos trabajando como sigue: cada proceso lee una línea de entrada especificando una suma de dinero, la cuenta para ser acreditada y la cuenta para ser debitada. Luego bloquea ambas cuentas y transfiere el dinero, liberando el bloqueo cuando termina. ¿Puede existir deadlock? ¿Por qué?. En caso afirmativo, describa posibles esquemas que eviten el mismo.

4. Considere un sistema en el siguiente estado:

	Asignación				Max				Disponible			
	A	*B*	*C*	*D*	*A*	*B*	*C*	*D*	*A*	*B*	*C*	*D*
P_0	0	0	1	2	0	0	1	2	1	5	2	0
P_1	1	0	0	0	1	7	5	0				
P_2	1	3	5	3	2	3	5	6				
P_3	0	6	3	2	0	6	5	2				
P_4	0	0	1	4	0	6	5	6				

   Conteste las siguientes preguntas utilizando el algoritmo del banquero:

   (a) ¿Cuál es el contenido de la matriz Need?

   (b) ¿El sistema está en un estado seguro?

   (c) Si llega un requerimiento de P1 para (0, 4, 2, 0) ¿puede ser otorgado inmediatamente?

5. Escriba un semáforo contador con una operación *wait()* que:

   (a) Permita ejecución concurrente a N procesos, bloqueando temporalmente el resto.

   (b) Bloquee hasta un máximo de N procesos, sin bloquear los N+1 siguientes. La operación debe retornar un booleano asociado a si la espera pudo iniciarse o no en cada caso.

6. Una barbería posee una sala de espera con n sillas y una sala para el barbero. Si no hay clientes esperando, el barbero duerme. Si un cliente entra a la barbería y todas las sillas están ocupadas, el cliente se retira. Si el barbero está ocupado, pero hay sillas, el cliente se sienta en una de las sillas libres. Si el barbero está dormido, el cliente lo despierta. Escriba un programa para coordinar al barbero y los clientes.

7. Una base de datos posee un conjunto de registros que son accedidos por n procesos. Varios procesos pueden leer en forma simultánea el mismo registro. Sin embargo, si un proceso está escribiendo sobre un registro, los demás no pueden leerlo hasta tanto no se complete la operación de escritura. Implemente el problema utilizando semáforos.

8. Escriba un corrector ortográfico que permita realizar las siguientes operaciones concurrentemente:

(a) Cargar una o más palabras.

(b) Eliminar una o más palabras.

(c) Verificar un conjunto de palabras, resultando en una lista de las que no están en el diccionario.

Tenga en cuenta dos variantes para el problema, donde la operación c) no asegura *consistencia temporal* y donde sí lo hace. Asegurar consistencia, por ejemplo, es evitar la situación en la cual un proceso obtiene, antes de terminar de procesar el conjunto de palabras, que ciertas palabras no están en el diccionario, pero a su vez otro proceso luego las agrega.

9. El Supermercado Baratija posee una canasta de productos en oferta. Con el fin de atraer al público, la canasta es modificada, agregando y eliminando productos cuando:

   (a) Un producto haya estado en oferta por más de una semana.

   (b) Se ha llegado al stock mínimo del producto.

   (c) Al Gerente de Ventas le parece adecuado.

   En las horas picos normalmente hay N cajas atendiendo en paralelo. Cuando un producto pasa por la caja el sistema verifica si el mismo está en la canasta. De ser así, aplica un 10% de descuento y luego actualiza el stock. Diariamente llegan al supermercado diferentes distribuidores trayendo pedidos de productos. Resolver maximizando la concurrencia entre cajas y evitando deadlocks.

10. Blancanieves y los siete enanitos viven en una casa donde sólo existen cuatro sillas que los enanitos utilizan para comer. Cuando un enanito vuelve de trabajar comprueba si hay una silla libre para sentarse. Si existe una silla libre, entonces indica a Blancanieves que ya está sentado y espera pacientemente que le sirva la comida. Cuando le ha servido, Blancanieves le indica que puede empezar a comer. El enanito come y cuando acaba, deja la silla libre y vuelve a trabajar. Por su parte Blancanieves, cuando no tiene ningún enanito pendiente de servirle, se va a pasear con su amigo el Príncipe.

    Escriba un programa que ejecute las funciones descritas. Debe existir un thread para Blancanieves y un thread para cada enanito. La solución debe estar libre de deadlocks. La solución debe maximizar la concurrencia de los thread. Argumente por qué la solución está libre de deadlocks.

11. Considere un club de fabricantes de vino con 8 miembros y un almacén con los ingredientes (jugo y levadura) al que puede acceder sólo una persona por vez. Para que uno de los miembros pueda hacer vino necesita utilizar 2 jarras, 1 unidad para fermentación/descanso de vino, jugo de fruta azucarado y levadura de vinificación. El proceso de la mezcla inicial requiere una estación de mezcla, el proceso de fermentación/descanso requiere 4 semanas para producir el vino una vez que los tres ingredientes se hayan mezclado correctamente. La segunda jarra se necesita solamente en el final del proceso para decantar el vino de la levadura muerta. El almacén contiene 2 estaciones de mezcla, 6 jarras (de 10 litros), 7 unidades para fermentación/descanso, 15 envases (de 5 litros) de jugo de fruta azucarado, y 20 paquetes de levadura de vino (para 10 litros de vino cada uno). Una vez que un miembro haya terminado su vino, todos prueban antes que el miembro comience a hacer más vino. Además, en el club existe un administrador que se encarga de reponer los ingredientes del almacén.

    Escriba un programa que simule a los miembros del club del vino considerando que cada miembro debe ser representado por un thread, la solución debe estar libre de deadlocks y debe que maximizar la concurrencia entre miembros del club. Justifique por qué el programa está libre de deadlocks.