TRABAJO PRACTICO 5

EJERCICIO 5.1

En un entorno de multiprogramación, varios procesos pueden competir por un numero finito de recursos. Un proceso solicita un recurso; si el recurso no está disponible en ese momento, el proceso entra en un estado de espera. Puede suceder que la espera del proceso jamás cambie de estado, porque los recursos que solicito están bloqueados por otro proceso en espera. Esta situación se llama deadlock.

EJERCICIO 5.2

- **Exclusión mutua:** cada recurso está asignado a un único proceso de manera exclusiva.
- **Retención y espera:** los procesos que tienen, en un momento dado, recursos asignados con anterioridad, pueden solicitar nuevos recursos y esperar a que se le asignen sin liberar antes alguno de los recursos que ya tenía asignados.
- **No apropiación**: los recursos otorgados con anterioridad no pueden ser forzados a dejar un proceso. El proceso que los posee debe liberarlos en forma explícita. Ni siquiera el sistema operativo puede expropiárselo.
- **Espera circular**: debe existir una cadena circular de dos o más procesos, cada uno de los cuales espera un recurso poseído por el siguiente miembro de la cadena. Esta condición es una consecuencia potencial de las tres primeras, es decir, dado que se producen las tres primeras condiciones, puede ocurrir una secuencia de eventos que desemboque en un círculo vicioso de espera irresoluble

EJERCICIO 5.3

Prevención:

Para que se produzca deadlock, deben estar presentes las cuatro condiciones nombradas anteriormente. Evitando que al menos una de estas no ocurra, podemos prevenir la ocurrencia de deadlock

Evasión:

Para evitar deadlock se necesita información adicional sobre como los recursos son requeridos. El algoritmo de evasión de interbloqueos dinámicamente examina el estado de asignación de recursos para asegurar que no puede haber una condición de espera circular.

Detección:

Permite que el sistema entre en estado de deadlock. Emplea un algoritmo de detección y un esquema de recuperación

EJERCICIO 5.4

Prevención

- El interés se centra en condicionar un sistema para que elimine toda posibilidad de que éstos se produzcan.
- Los métodos pueden dar como resultado una pobre utilización de los recursos, aún así son ampliamente utilizados.

Evasión

- La meta es imponer condiciones menos estrictas que en la prevención, para intentar lograr una mejor utilización de los recursos.
- No precondiciona al sistema para que evite todas las posibilidades de que se produzca un bloqueo.
- Permiten la aparición del bloqueo, pero siempre que se produce una posibilidad de bloqueo, éste se esquiva.

EJERCICIO 5.5

Un sistema está en estado seguro solo si existe una secuencia de procesos P1,P2,..,Pn, donde para cada Pi, el recurso que Pi requiere puede ser otorgado por los recursos actualmente disponible, teniendo en cuenta los recursos utilizados por Pj, donde j<i.

EJERCICIO 5.6

A)

- **Exclusión mutua:** cada cruce puede ser atravesada por un auto a la vez
- **Retención y espera:** un vehículo que quiere atravesar un cruce, debe esperar que no haya otro vehículo cruzándolo

- **No apropiación**: La única forma de que un vehículo no se encuentre en un cruce, es avanzando. Solo avanzará si no hay otro vehículo adelante
- **Espera circular**: puede producirse que varios vehículos estén esperando para atravesar un cruce, pero el vehículo que está atravesándolo no puede para porque el vehículo de enfrente no puede avanzar, ya que está bloqueado por una secuencia de vehículos que se encuentran a la espera circularmente (un verso bárbaro)

B)

Un vehículo solo puede atravesar un cruce si está seguro que existe un espacio disponible luego del cruce.

EJERCICIO 5.7

No

EJERCICIO 5.8

A)

En tiempo t0 el sistema se encuentra en estado seguro (no está interbloqueado). La secuencia <P1,P0,P2> satisface la condición de seguridad, ya que al proceso P1 se le pueden asignar inmediatamente todos los recursos que requiere y luego liberarlos. Luego el proceso P0 puede obtener los recursos y liberarlos, y finalmente el proceso P2.

EJERCICIO 5.9

A)

Necesidad = Max – Asignados

Procesos	Asignacion				Máximo				Disponible				Necesidad			
	A	B	C	D	A	B	C	D	A	B	C	D	A	B	C	D
P1	0	0	1	2	0	0	1	2	1	5	2	0	0	0	0	0
P2	1	0	0	0	1	7	5	0					0	7	5	0
P3	1	3	5	4	2	3	5	6					1	0	0	2
P4	0	6	3	2	0	6	5	2					0	0	2	0
P5	0	0	1	4	0	6	5	6					0	6	4	2

B)

1. Inicio
   - Disponible = 1,5,2,0
   - Finalizados= F,F,F,F,F
   - Secuencia = []
2. Loop 1
   - Se puede ejecutar P1
   - Disponible = 1,5,3,2
   - Finalizados = T,F,F,F,F
   - Secuencia = [P1]
3. Loop 2
   - Se puede ejecutar P3
   - Disponible = 2,8,8,6
   - Finalizados = T,F,T,F,F
   - Secuencia = [P1, P3]
4. Loop 3
   - Se puede ejecutar P2
   - Disponible = 3,8,8,6
   - Finalizados = T,T,T,F,F

- Secuencia = [P1, P3, P2]

5. Loop 4
   - Se puede ejecutar P4
   - Disponible = 3,14,11,8
   - Finalizados = T,T,T,T,F
   - Secuencia = [P1, P3, P2, P4]
6. Loop 4
   - Se puede ejecutar P5
   - Disponible = 3,14,12,12
   - Finalizados = T,T,T,T,T
   - Secuencia = [P1, P3, P2, P4, P5]

Secuencia segura P1, P3, P2, P4, P5

C)

Si

EJERCICIO 5.10

A)

Procesos	Mantiene	Maximo	Necesita	Disponible
P0	45	70	25	25
P1	40	60	20	
P2	15	60	45	
P3	25	60	35	

1. Inicio
   - Disponible = 25
   - Finalizados = F,F,F;F
   - Secuencia = []
2. Loop 1
   - Se puede seleccionar P0
   - Disponible = 70
   - Finalizados = T,F,F,F
   - Secuencia = [P0]
3. Loop 2
   - Se puede seleccionar P1
   - Disponible = 110
   - Finalizados = T,T,F,F
   - Secuencia = [P0,P1]
4. Loop 3
   - Se puede seleccionar P2
   - Disponible = 125
   - Finalizado = T,T,T,F
   - Secuencia = [P0,P1,P2]
5. Loop 4
   - Se puede selccionar P3
   - Disponible = 150
   - Finalizado = T,T,T,T
   - Secuencia = [P0,P1,P2,P3]

Secuencia segura P0,P1,P2,P3

B)

Procesos	Mantiene	Maximo	Necesita	Disponible
P0	45	70	25	15
P1	40	60	20	
P2	15	60	45	
P3	35	60	25	

1. Inicio
    - Disponible = 25
    - Finalizados = F,F,F,F
    - Secuencia = []
2. Loop 1
    - Recursos innecesarios

EJERCICIO 5.11

Cuestiones que dificultarían la implementación del algoritmo del banquero en un sistema operativo real.

- Se debe conocer la máxima demanda de recursos por anticipado.
- La ejecución de los recursos no debe estar forzada por la sincronización.
- Se tiene un número fijo de procesos y recursos.
- Los procesos no finalizan mientras retengan recursos.
- Requiere que los procesos salden sus préstamos en un tiempo finito.

EJERCICIO 5.12

- Selección de una víctima – minimiza costos.
- Rollback – retorna a algún estado seguro, reinicia el proceso desde ese estado.
- Inanición – algunos procesos pueden ser elegidos siempre como víctimas, incluir un número de rollbacks en el factor de costo.