UTN – 1° Parcial	Sistemas Operativos	11/05/2019

Nombre y Apellido:..... Curso: .....

TEORÍA					PRÁCTICA			NOTA
1	2	3	4	5	1	2	3	

TEORÍA: Responda brevemente las siguientes preguntas. **Justifique**.

1. Compare los algoritmos HRRN, SJF con desalojo y sin desalojo en términos de criterio de selección, posible desalojo, penalización procesos cortos/largos, Overhead, y Starvation.
2. Describa brevemente los tipos de semáforos y qué problema resuelve cada uno. ¿Cómo solucionaría un problema de productor/consumidor con un buffer infinito utilizando semáforos?
3. En un sistema que tiene modos de ejecución para garantizar la protección, indique.
   a. ¿Cómo sería la forma correcta de realizar una operación sobre el HW?
   b. Indique una forma "incorrecta" y explique por qué no sería permitida.
4. V o F, justifique en ambos casos:
   a. En una situación de deadlock una buena solución suele ser matar a todos los procesos involucrados.
   b. Utilizando el algoritmo de evasión podría llegar a ocurrir un deadlock si es que todos los procesos piden lo máximo que podrían pedir.
5. Explique cómo el planificador de corto plazo podría llegar a generar una condición de carrera. Proponga una forma de solucionarla, sin soporte del SO, que pueda ser aplicado en entornos con múltiples procesadores.

PRÁCTICA: Resuelva los siguientes ejercicios **justificando** las conclusiones obtenidas.

Ejercicio 1

Un SO con un planificador a corto plazo que implementa un algoritmo Virtual Round Robin Q=3, ejecuta una serie de procesos que utilizan una biblioteca de hilos de usuario que planifica mediante un algoritmo SJF (sin desalojo).

		Llegada	CPU	I/O	CPU
KLT 1	**ULT 1**	3	7	-	-
KLT 2	**ULT 2**	1	3	3	3
KLT 3	**ULT 3.1**	0	1	2	2
	ULT 3.2	1	1	2	2

a) Realice el diagrama de GANTT correspondiente a la ejecución de los hilos
b) Indique, justificando su respuesta, a partir de qué instante cambiaría el diagrama si:
   i) El planificador de corto plazo fuese Round Robin (mismo quantum)
   ii) La biblioteca de hilos implementara la técnica de jacketing

Ejercicio 2

Peter necesita descifrar la cantidad total de recursos que tiene un sistema. Con sus habilidades de sistemas operativos logró averiguar que dicho sistema utiliza evasión de deadlocks y que las matrices de asignación y de necesidad en dicho momento son las siguientes:

NECESIDAD

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

ASIGNACIÓN

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

Indique:
a) ¿Cuál sería el mínimo vector de recursos totales posible en dicho sistema?
b) Luego Peter logra obtener más información para refinar su búsqueda: sabe que se realizan las siguientes peticiones en orden y ve que la tercera no es asignada inmediatamente : P1 1R1 – P4 1R2 – P4 1R3. ¿Con esta nueva información cambia su respuesta dada en a)?

Ejercicio 3

Peter suele ir al Super a hacer las compras. Al llegar, lo primero que hace es buscar un canasto para sus productos. Si hay uno disponible comenzará a usarlo, en caso contrario, deberá esperar hasta que otro cliente se retire y devuelva el suyo. El Super cuenta con un máximo de 20 canastos.
Una vez que agarra todos sus productos se dirige a la zona de los cajeros. Dentro del Super hay dos cajeros que atienden en simultáneo a los clientes siempre que haya alguien esperando para ser atendido. La atención de un cliente consiste en pasar los productos por la lectora de códigos, luego entregar el ticket y finalmente procesar el pago. Si el monto en su caja supera un valor, el cajero llevará la plata hasta la caja fuerte (compartida por ambos)
Teniendo en cuenta el pseudocódigo presentado, sincronice únicamente con semáforos.

Cliente (N instancias)	Cajero (2 instancias)

`dirigirse_al_super()`
`agarrar_canasto()`
`cargar_canasto()`
`esperar_en_cola()`
`id_cajero = recibir_llamado_cajero()`
`entregar_productos()`
`recibir_ticket()`
`pagar()`
	`while(1){`
	`  llamar_cliente()`
	`  pasar_productos_por_lectora()`
	`  entregar_ticket()`
	`  saldo_actual += recibir_dinero()`
	`  if(saldo_actual > 10000){`
	`  depositar_en_caja_fuerte(saldo_actual)`
	`  }`
	`}`

NOTAS:
- La función recibir_llamado_cajero() devuelve 0 o 1 dependiendo del cajero que realizó el llamado al cliente.
- El proceso "Cajero" cuenta con la función get_id() que devuelve 0 o 1 dependiendo del número de cajero.

Condiciones de aprobación: 3 preguntas correctamente respondidas y 1.5 ejercicios correctamente resueltos.