UTN – 1° Parcial	Sistemas Operativos	26/09/2015

Nombre y Apellido:..... Curso: ….....

TEORÍA					PRÁCTICA			NOTA
1	2	3	4	5	1	2	3	

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

1. **Relacione los conceptos de: cambio de modo - cambio de contexto - cambio de proceso**
2. **Escriba el pseudo-código de dos hilos cooperativos, incluyendo (y señalando) dónde se referencia su stack, su sección de datos y su heap.**
3. **Indique qué similitudes y diferencias hay entre el algoritmo del banquero y el de detección de deadlocks. ¿Por qué se dice que la técnica de evasión es "pesimista"?**
4. **V o F. En un esquema en que se utiliza un algoritmo de planificación por prioridades, si dos procesos de diferente prioridad comparten un mismo recurso y utilizan mutex con espera activa se puede llegar a generar una situación en la que ninguno de los dos pueda ejecutar. Justifique**
5. **Explique cuál podría ser el problema de utilizar semáforos sin espera activa en procesos que utilizan ULTs.**

Bonus: Al programar en C, es posible "atrapar" algunas señales, asignándoles diferentes funciones (código). Cuando una señal es atrapada, el proceso frena su ejecución, corre el código asignado a la señal, y luego continúa la ejecución normal. Explique cómo podría esto producir un deadlock entre el proceso y sí mismo.

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

Ejercicio 1

Un Sistema Operativo, que posee una única instancia de cada recurso, detecta y corrige Deadlocks. En un instante, se encuentra en el estado que se muestra en las siguientes tablas:

a. Indique **utilizando únicamente el algoritmo de detección** qué procesos se encuentran o no en deadlock.
b. Realice el grafo de asignación de recursos. ¿Su respuesta cambia? ¿Qué problema presenta el algoritmo antes mencionado?

Recursos asignados

	R1	R2	R3
P1	1	0	0
P2	0	1	0
P3	0	0	1

Pedidos actuales

	R1	R2	R3
P1	0	1	0
P2	1	0	0
P3	1	1	0

Ejercicio 2

Teniendo los hilos K1, K2 y K3 listos para ejecutar, muestre su ejecución en un gráfico Gantt si el SO utiliza SJF con desalojo y la biblioteca de hilos de usuarios HRRN (sin desalojo).

KLT	ULT	CPU	IO	CPU	IO	CPU
K1	U1	3	-	-	-	-
	U2	2	1	1	-	-
	U3	1	1	1	-	-
K2	-	2	1	1	-	-
K3	-	2	2	3	1	2

Para el primer algoritmo se deberán calcular los estimados, teniendo como valores iniciales: Est k1=2, Est k2 =3 , Est k3 = 1.
$Est_{(n+1)} = \alpha\ R_n + (1-a)\ Est_n$
(con a = 0,5)

Nota1: Todos los ULTs llegaron en 0.
Nota2: Las IOs se realizan a través de la biblioteca de ULTs.

También responda (sin realizar la planificación nuevamente) qué hubiese cambiado si:

a. Se usa jacketing
b. Se agrega una CPU
c. Las IOs se realizan directamente al SO

Ejercicio 3

Un equipo de desarrollo programa y commitea constantemente. Cada commit es revisado, y aprobado por un programador. Peter es un programador excepcional, por lo que cada vez que programa, realiza 5 commits diferentes. Si hay más de 20 commits revisados, los deploya (los sube a un servidor web). Por último, un tester prueba los cambios subidos. Nunca debería haber más de 10 commits sin revisar.

Programador (4 instancias)	Peter (1 instancia)	Tester (1 instancia)

while(1) {
  if(commits_a_revisar > 0) {
  revisar_un_commit(1);
  commits_ok++;
  commits_a_revisar--;
  } else {
  programar();
  commitear(1);
  commits_a_revisar++;
  }
}
	while(1) {
	  if(commits_ok > 20) {
	  commits_ok = 0;
	  deployar();
	  } else {
	  super_programar();
	  commitear(5);
	  commits_a_revisar += 5;
	  }
	}
		while(1) {
		  testear_deploy();
		}

Sincronice el siguiente pseudo-código para que el equipo de desarrollo trabaje sin inanición ni deadlocks, y para que se cumpla el ciclo explicado.

Condiciones de aprobación: 3 preguntas correctamente respondidas y 1,5 ejercicios correctamente resueltos.