Sistemas Operativos I - Parcial - 1/11/2018

- Nombre:
- DNI:
- Firma (**al entregar y ante un docente**):

Notas:

- Los alumnos que obtuvieron una calificación distinto de M (Mal) en el examen de scripting pueden obviar resolver el ejercicio 1).
- Se solicita realizar el examen con letra clara y legible.

1. Considere la puntuación obtenida por dos equipos en una competencia de arquería olímpica, que se encuentra almacenada en un archivo. Cada equipo consta de 2 integrantes. La competencia incluye 5 sets, y en cada set, cada participante suma puntos (entre 1 y 10) dependiendo del lugar en la diana en que arroja su flecha. Por ejemplo:

Set	Jugador1-Equipo1	Jugador2-Equipo1	Jugador1-Equipo2	Jugador2-Equipo2
1	10	10	9	7
2	9	10	10	9
3	8	10	10	10
4	10	9	8	9
5	10	9	10	10

   El equipo victorioso es aquel que logra conseguir más sets ganados en total. Un set es ganado por un equipo si la suma de los puntos de ambos jugadores supera a los puntos de los jugadores del otro equipo. Por ejemplo, el Equipo 1 ha ganado claramente el set #1.

   Codifique un script bash que reciba como parámetro la ruta al archivo de resultados e imprima por pantalla el equipo que resultó ganador, o bien "Empate".

2. Se tiene una red social representada por un grafo dirigido que indica cuándo una persona sigue a otra (y por ende quién es seguido por quién), como ilustra la Figura. Implemente un programa que dada una red social arbitraria calcule de forma concurrente el usuario que sigue más personas ("Jena" para el ejemplo), y el usuario más seguido ("Ralph" para el ejemplo). El cálculo debe hacerse utilizando N threads, con N dado por parámetro y sabiendo que $N < NroUsuariosRed$. Cada thread debe procesar por vez un usuario distinto de la red. Cuando un thread procesa un usuario, actualiza el usuario que más sigue o más seguido, si correspondiere. Si no hay más usuarios disponibles para procesar, cada thread debe finalizar. Existe un thread aparte que imprime el resultado cada vez que algún thread actualiza.

   Asuma que la red social está representada por una instancia de la clase *SocialNetwork* (ya implementada), y que tiene los siguientes métodos: *String[] getNetworkUserNames()*, *String[] getFollowersOf(String userName)*, y *String[] getFolloweesOf(String userName)*. Para el ejemplo, *getFollowersOf(Bob)* retorna [Alice, Jena] y *getFollowees(Bob)* retorna [Ralph].

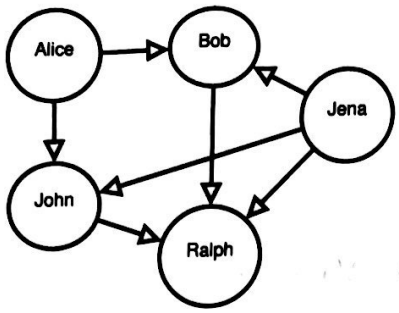

Figura 1: Ejemplo de red social

a) Programe una solución evitando *race conditions*, *busy waits*, *deadlocks*, y *maximizando concurrencia*. El código *debe estar comentado*, especialmente las estructuras concurrentes. Debe usar arreglos planos para las estructuras compartidas entre threads (no se permite Vector o listas sincronizadas).

b) Explique por qué su solución está libre de deadlocks, relacionando con las condiciones vistas en teoría de acuerdo a los procesos/hilos y recursos a partir de resolver el punto a).

3. Considere el siguiente conjunto de procesos:

Proceso	Tiempo de CPU requerido (en ms)	Tiempo de arribo (en ms)
P1	6	0
P2	1	1
P3	2	2
P4	1	3
P5	5	4

a) Grafique diagramas de Gantt ilustrando la ejecución utilizando los algoritmos SJF con desalojo y SJF sin desalojo.

b) Calcule el tiempo promedio de *turnaround* para los procesos de acuerdo a los dos algoritmos graficados.

4. ¿Es necesario recompilar o reescribir el código de las aplicaciones para aprovechar los esquemas de paginación de memoria provistos por el sistema operativo? Justifique su respuesta.

Puntaje individual (a completar por la cátedra)

1	2		3		4	Nota final
	a	b	a	b		