

S.O.A.: Tareas Cortas

Lunes 24 de Febrero

I. TAREA 1: SINCRONIZACIÓN DE “threads”

Un problema clásico de Sistemas Operativos es la sincronización de *threads* usando semáforos. Suponga que tenemos una carretera de dos carriles, uno en cada sentido, que tiene en medio un puente de un solo carril de ancho. La Figura 1 muestra un diagrama, no a escala, de la situación. Suponga que el puente es arbitrariamente largo.



Figura 1

Los autos viajan de Este a Oeste y viceversa. Al llegar a la entrada del puente se detienen brevemente y únicamente entran al puente si no hay ningún auto en el puente, o si los que están adentro viajan en el mismo sentido que ellos.

Escriba un programa en modo texto que simule este problema. El programa recibe de **línea de comando** dos números que establecen cuántos autos vienen del Este y cuántos del Oeste. Los autos serán *threads* creados aleatoriamente con tiempos distribuidos exponencialmente entre una creación y la siguiente (escojan la media de esta distribución - podrían ser diferentes para cada sentido).

Se desplegarán mensajes apropiados para ilustrar **todo** lo que está pasando. No se permite el uso de *busy waiting*. El proyecto debe correr sobre Linux y debe usar *Pthreads*.

Esta tarea corta se desarrollará en los grupos de proyecto.

Enviar un .tgz a torresrojas.cursos@gmail.com antes de las **11:59pm del Lunes 24 de Febrero del 2020** cuyo nombre sea la concatenación de los apellidos del grupo con mayúsculas al inicio de cada uno (e.g., Torres-Venegas-Araya.tgz) que contenga el fuente, un makefile, un readme que explique como se usa su programa, especialmente cualquier opción extra incluida, y mencionando todo lo que NO funcione de su proyecto.

Identifique claramente su correo con el siguiente subject:

[SOA] Tarea Corta 1 - Apellido 1 - Apellido 2 - etc.

Lunes 20 de Abril

II. TAREA 2: RASTREADOR DE “System Calls”

Su programa tendrá la misión de poner a ejecutar a otro programa (digámosle *Prog*), pasarle los argumentos seleccionados por el usuario y rastrear todos los *system calls* utilizados por *Prog*.

La sintaxis de ejecución desde línea de comando es:

```
rastreador [opciones rastreador] Prog
[opciones de Prog]
```

Las [opciones rastreador] podrían no venir del todo o aparecer en cualquier orden o combinación válida.

En todo caso, al final de la ejecución de *Prog*, *rastreador* **siempre** desplegará en la salida estándar una tabla acumulativa que muestre todos los *System Calls* utilizados por *Prog*, así como el número de veces que fue utilizado cada uno.

Las [opciones de Prog] no serán analizadas ni consideradas por *rastreador*, sino que simplemente serán pasadas a *Prog* al iniciar su ejecución.

Las opciones válidas para *rastreador* son:

- **-v** desplegará un mensaje cada vez que detecte un System Call de *Prog*. Se debe desplegar la mayor cantidad posible de detalles respecto a cada System Call.
- **-V** será idéntico a la opción **-v**, pero hará una pausa hasta que el usuario presione cualquier tecla para continuar la ejecución de *Prog*.

Esta tarea corta se desarrollará en los grupos de proyecto.

Enviar un .tgz a torresrojas.cursos@gmail.com antes de las **11:59pm del Lunes 20 de Abril del 2020** cuyo nombre sea la concatenación de los apellidos del grupo con mayúsculas al inicio de cada uno (e.g., Torres-Venegas-Araya.tgz) que contenga el fuente, un makefile, un readme que explique como se usa su programa, especialmente cualquier opción extra incluida, y mencionando todo lo que NO funcione de su proyecto.

Identifique claramente su correo con el siguiente subject:

[SOA] Tarea Corta 2 - Apellido 1 - Apellido 2 - etc.

Lunes 18 de Mayo

III. TAREA 2: TIEMPO LÓGICO

Envíe un PDF a `torresrojas.cursos@gmail.com` con las respuestas a estas preguntas, antes de las **11:59pm del 18 de Mayo del 2020**.

Esta tarea es individual.

1. Demuestre matemáticamente la propiedad fuerte del reloj para relojes vectoriales
2. Dado un conjunto arbitrario de relojes vectoriales todos de las mismas dimensiones, podría haber uno o más relojes vectoriales que no pueden venir de la misma historia global que los otros. Dé al menos 2 ejemplos de conjuntos de relojes vectoriales con esta característica.
3. Investigue la técnica conocida como relojes plausibles. Explique su propósito, implementación y casos donde son utilizables.

Identifique claramente su correo con el siguiente subject:

[SOA] Tarea Corta 3 - Nombre Apellido