

## Programación Concurrente 2025

### Cuestionario guía - Clases Teóricas 3, 4 y 5

- 1- a) Explique la semántica de un semáforo. Diferencie el caso de los semáforos generales y los binarios.  
b) Indique los posibles valores finales de  $x$  en el siguiente programa (**justifique claramente su respuesta**):  

```
int x = 4; sem s1 = 1, s2 = 0;  
co P(s1); x = x * x ; V(s1);  
    // P(s2); P(s1); x = x * 3; V(s1);  
    // P(s1); x = x - 2; V(s2); V(s1);  
oc
```
- 2- Desarrolle utilizando semáforos una solución centralizada al problema de los filósofos, con un administrador único de los tenedores, y posiciones libres para los filósofos (es decir, cada filósofo puede comer en cualquier posición siempre que tenga los dos tenedores correspondientes).
- 3- Describa la técnica de *Passing the Baton*. ¿Cuál es su utilidad en la resolución de problemas mediante semáforos?
- 4- Modifique las soluciones de Lectores-Escritores con semáforos de modo de no permitir más de 10 lectores simultáneos en la BD y además que no se admita el ingreso a más lectores cuando hay escritores esperando.
- 5- Describa el funcionamiento de los monitores como herramienta de sincronización.
- 6- ¿Qué diferencias existen entre las disciplinas de señalización “Signal and wait” y “Signal and continue”?
- 7- ¿En qué consiste la técnica de *Passing the Condition* y cuál es su utilidad en la resolución de problemas con monitores? ¿Qué relación encuentra entre *passing the condition* y *passing the baton*?
- 8- Desarrolle utilizando monitores una solución centralizada al problema de los filósofos, con un administrador único de los tenedores, y posiciones libres para los filósofos (es decir, cada filósofo puede comer en cualquier posición siempre que tenga los dos tenedores correspondientes).
- 9- Sea la siguiente solución propuesta al problema de asignación de recursos:  

```
monitor SJA {  
    bool libre = true;  
    cond turno;  
    procedure request(int tiempo) {  
        if (not libre) wait(turno, tiempo);  
        libre = false;  
    }  
    procedure release() {  
        libre = true;  
        signal(turno);  
    }  
}
```

  - a) Funciona correctamente con disciplina de señalización Signal and Continue?
  - b) Funciona correctamente con disciplina de señalización Signal and Wait?Justifique claramente sus respuestas.
- 10- Modifique la solución anterior para el caso de no contar con una instrucción wait con prioridad.

- 11- Modifique utilizando monitores las soluciones de Lectores-Escritores visto en la teoría, de modo de no permitir más de 10 lectores simultáneos en la BD, y además que no se admita el ingreso a más lectores cuando hay escritores esperando para ingresar.
- 12- Resuelva con monitores el siguiente problema: Tres clases de procesos comparten el acceso a una lista enlazada: searchers, inserters y deleters. Los searchers sólo examinan la lista, y por lo tanto pueden ejecutar concurrentemente unos con otros. Los inserters agregan nuevos ítems al final de la lista; las inserciones deben ser mutuamente exclusivas para evitar insertar dos ítems casi al mismo tiempo. Sin embargo, un insert puede hacerse en paralelo con uno o más searches. Por último, los deleters remueven ítems de cualquier lugar de la lista. A lo sumo un deleter puede acceder la lista a la vez, y el borrado también debe ser mutuamente exclusivo con searches e inserciones.
- 13- El problema del “Puente de una sola vía” (*One-Lane Bridge*): autos que provienen del Norte y del Sur llegan a un puente con una sola vía. Los autos en la misma dirección pueden atravesar el puente al mismo tiempo, pero no puede haber autos en distintas direcciones sobre el puente.
- a) Desarrolle una solución al problema, modelizando los autos como procesos y sincronizando con un monitor (no es necesario que la solución sea *fair* ni dar preferencia a ningún tipo de auto).
  - b) Modifique la solución para asegurar *fairness* (**Pista:** los autos podrían obtener turnos)
- 14- Indicar las características generales de Pthreads. Explicar cómo se maneja la sincronización por exclusión mutua y por condición en Pthreads. Indicar la relación entre ambas. Explicar cómo se pueden simular los monitores en Pthreads