UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

Sistemas Operativos

Grupo 6

Tarea 1

Ejercicios de Sincronización

Integrantes:

Romero Pizano Christian Gustavo

Avila Reyes Iker

Semestre 2025-2

Maestro Gunnar Eyal Wolf Iszaevich

27 de marzo del 2025

Documentación

1. Problema

Escogimos el ejercicio “Cruce de río” que simula el cruce seguro de desarrolladores (*hackers* y *serfs*) en una balsa para asistir a una reunión con las siguientes reglas:

- Hay solo una balsa
- En la balsa tienen que ingresar 4 personas (ya sean hackers o serfs), no más no menos
- Las posibles opciones en las que pueden cruzar el río en la balsa son 4 hackers, 4 serfs o 2 hackers y 2 serfs, puesto que en otro caso podrían pelear entre ellos

Por lo tanto buscamos coordinar el cruce evitando conflictos entre desarrolladores y condiciones de carrera.

2. Lenguaje y Entorno de Desarrollo

Escogimos el lenguaje de programación de Java debido a que nos encontramos familiarizados con el lenguaje, además de su soporte nativo para la concurrencia con java.util.concurrent.Sempahore para el uso de semáforos y el manejo de hilos con AtomicInteger.

El código fue desarrollado en Netbeans debido a que ya estábamos familiarizados con el mismo, por lo que únicamente necesitabamos tener el JDK instalado.

3. Estrategia de Sincronización

El mecanismo en el que se basa nuestro código es el uso de semáforos. Implementamos un mutex para controlar el acceso a los contadores compartidos (para checar la cantidad de hackers y serfs en la balsa), garantizando que solo un hilo modifique los contadores a la vez (exclusión mutua) para así cuidar la sección crítica.

Donde nuestro ejemplo consiste en una barrera que si no se llega a cumplir alguna de las condiciones establecidas dentro de las reglas, el desarrollador debe esperar en su semáforo (tal y como lo vimos en los ejemplos presentados en clase).
Una vez que un grupo de desarrolladores logren cruzar, se liberan los permisos para notificar a los desarrolladores restantes y así puedan subir a la balsa

Dentro del código podemos encontrar que en la mayoría de las salidas, vamos a encontrarnos el caso de 2 hackers y 2 serfs. Esto se debe a la lógica de la verificación donde

el orden de las condiciones y la acumulación rápida de los hilos disminuyen la probabilidad de encontrar un caso donde pase un grupo de puros hackers o un grupo de puros serfs. También, se puede llegar a forzar un cruce al quedar pocas personas con el fin de que todos lleguen a la reunión.

Dicho esto consideramos que logramos resolver un problema de sincronización con el uso de semáforos, garantizando condiciones seguras y eficientes,