UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

Materia

Sistemas Operativos

Profesor

Gunnar Eyal Wolf Iszaevich

Alumnos

Luna Quintero Diego Alejandro

Coronado Pérez Diego

Tarea 1

Problema del Cruce del Río

No. Cuenta

320225888

320252460

Grupo

8

Semestre

2026 – 1

Fecha de Entrega

16 de Octubre del 2025

Ciudad Universitaria, Cd. Mx., 2025

Problema del cruce del río

Problemática

El problema de la 'balsa de hackers y serfs' es un clásico ejemplo de sincronización de hilos en sistemas concurrentes. En una orilla hay dos tipos de personas —hackers y serfs— que desean cruzar un río en una balsa que sólo puede transportar cuatro personas a la vez.

Reglas:
1. La balsa puede cruzar solo si hay exactamente cuatro personas.
2. Las combinaciones válidas son: 4 hackers, 4 serfs o 2 hackers y 2 serfs.
3. Los demás deben esperar hasta formar una combinación válida.

Objetivo del Programa

El objetivo es simular la coordinación concurrente de los hackers y serfs para cruzar el río, controlando los accesos a la balsa mediante semáforos y evitando condiciones de carrera o bloqueos indefinidos.

Descripción General del Código

El programa está escrito en Python y utiliza el módulo threading para manejar los hilos. Cada hilo representa una persona (hacker o serf) que llega aleatoriamente y espera para subir a la balsa.

Variables globales:
- hackers, serfs: cuentan las personas esperando.

Semáforos:
- mutex: asegura acceso exclusivo a variables compartidas.
- balsa: sincroniza los viajes.

Función persona_llega(): simula la llegada de cada persona y decide cuándo la balsa puede zarpar.

Función main(): crea y lanza los hilos, simulando las llegadas aleatorias.

Conceptos de Concurrencia Utilizados

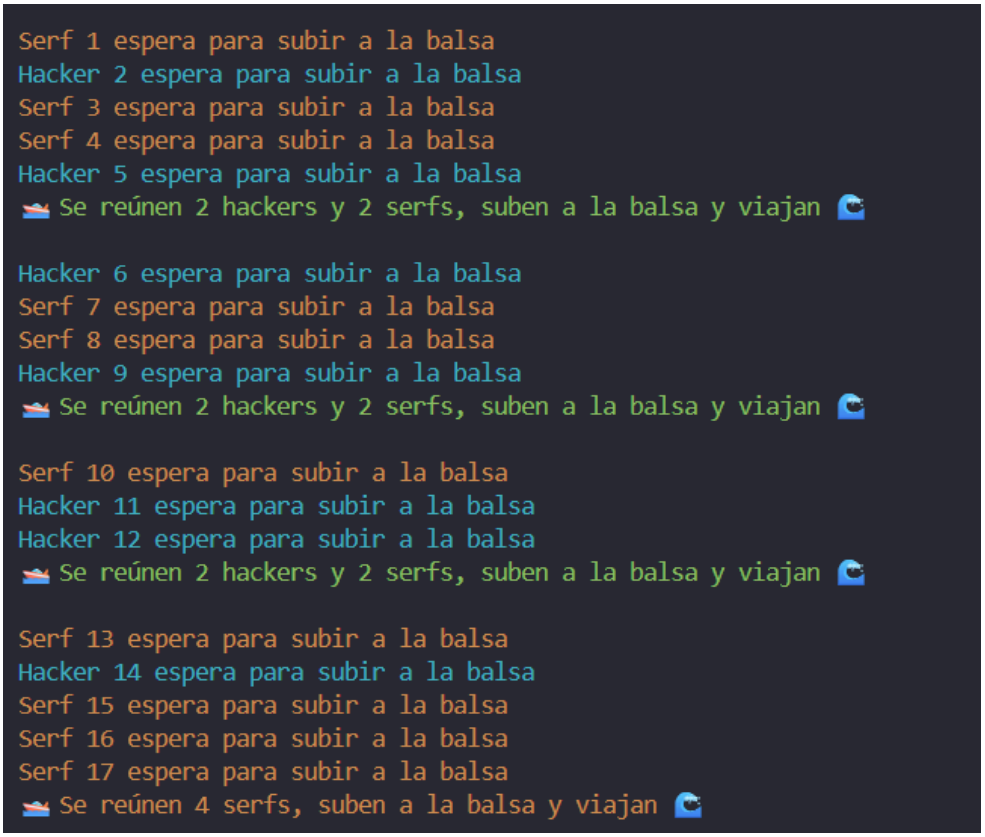
• Hilos (Threads): unidades de ejecución concurrentes.
• Semáforos: controlan el acceso a recursos compartidos.
• Exclusión mutua: evita condiciones de carrera.
• Sincronización: coordina el trabajo de los hilos.

Flujo de Ejecución del Programa

1. Llegan personas aleatorias (hackers o serfs).
2. Se anuncian y esperan.
3. Cuando hay una combinación válida, los 4 suben y viajan.

4. Los que no alcanzan grupo se retiran tras un timeout.
5. Al final se informa que los viajes han terminado.

Salida Ejemplo



Conclusión

Este ejercicio demuestra cómo resolver un problema de concurrencia utilizando semáforos y sincronización de hilos en Python. Permite comprender cómo coordinar múltiples procesos simultáneos y aplicar los principios de Sistemas Operativos y Programación Concurrente.