UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

INGENIERÍA EN COMPUTACIÓN

SISTEMAS OPERATIVOS

TAREA 01: SINCRONIZACIÓN

NOMBRE COMPLETO: Lechuga Castillo Shareny Ixchel,
Gonzalez Cuellar Pablo Arturo

Nº de Cuenta: 319004252, 319241013

GRUPO DE TEORÍA: 06

SEMESTRE 2025-1

FECHA DE ENTREGA LÍMITE: 22-10-2024

CALIFICACIÓN: _____

Informe: Sincronización - Alumnos y Asesor

1. Problema Resuelto:

El problema que decidimos resolver modela la interacción entre un profesor y varios estudiantes durante su horario de atención. El objetivo es que los estudiantes tomen turnos para hacer preguntas al profesor, mientras esperan sentados en las sillas disponibles dentro del cubículo. Además, cuando no hay estudiantes esperando, el profesor puede dormirse.

Planteamiento:

Un profesor de la facultad asesora a varios estudiantes, y estamos en su horario de atención.

Objetivo:

Modelar la interacción durante este horario de modo que la espera para todos sea lo más corta posible.

Reglas:

- El profesor tiene un número limitado de *x* sillas en su cubículo.

  - Cuando no hay alumnos que atender, las sillas sirven como sofá, y el profesor se acuesta a dormir la siesta.

- Los estudiantes pueden tocar a la puerta del cubículo en cualquier momento, pero no pueden entrar más de *x* alumnos a la vez.

- Para evitar confundir al profesor, solo un estudiante puede presentar su duda y esperar la respuesta al mismo tiempo:

  - Los demás estudiantes sentados deben esperar pacientemente su turno.

  - Cada estudiante puede hacer entre 1 y *y* preguntas, permitiendo que los demás también hagan preguntas mientras tanto.

2. Lenguaje y Entorno:

- Lenguaje: Python

- Entorno de Desarrollo: Python 3.7+ con soporte para hilos (threading).

- Librerías utilizadas:

  - `threading`: Para manejar los hilos (uno por cada estudiante y uno para el profesor).

  - `time`: Para simular los tiempos de espera entre preguntas.

- `random`: Para generar de forma aleatoria el número de preguntas que hace cada estudiante.

3. Instrucciones para ejecutar el programa:

1. Clona el repositorio o descarga el archivo `alumnos_y_asesor.py`.

2. Asegúrate de tener Python 3.7 o superior instalado.

3. Ejecuta el programa.

4. ¿Qué esperar del programa?

- El programa simulará la llegada de estudiantes al cubículo del profesor. Cada estudiante intentará sentarse en una de las sillas disponibles (limitadas por la variable `n_sillas`).

- Solo un estudiante podrá hacer preguntas al profesor a la vez. El número de preguntas que cada estudiante hace es aleatorio, entre 1 y 2.

- Al terminar de hacer sus preguntas, el estudiante dejará libre su silla para que otro pueda sentarse. Si no hay estudiantes esperando, el profesor se dormirá.

- Este ciclo se repetirá hasta que todos los estudiantes hayan sido atendidos.

5. Terminación del programa:

El programa seguirá ejecutándose indefinidamente, a menos que controles la cantidad de estudiantes o modifiques la duración de la simulación. Para detener el programa manualmente, puedes presionar Control+C en la terminal.

4. Estrategia de Sincronización:

- Semáforos: Se utiliza un semáforo para controlar el número de sillas disponibles en el cubículo. Solo un número limitado de estudiantes puede estar sentado al mismo tiempo, determinado por la variable `n_sillas`.

- Locks: Se usa un `lock` para garantizar que solo un estudiante pueda interactuar con el profesor a la vez. Esto previene que dos estudiantes hagan preguntas simultáneamente.

- Eventos: Se usa un evento para simular que el profesor duerme cuando no hay estudiantes esperando. El evento se activa cuando un estudiante llega y quiere hacer preguntas, despertando así al profesor.

5. Refinamientos Implementados:

Se han implementado los siguientes refinamientos en la solución:

1. Número máximo de preguntas: Se estableció un límite en la cantidad de preguntas que cada estudiante puede hacer al profesor. Esto mejora la simulación, controlando la interacción y reduciendo el tiempo de espera de los demás estudiantes.
2. Manejo de excepciones: Se añadieron capturas de excepciones para manejar posibles errores que puedan surgir en los hilos durante la ejecución, asegurando que el programa se comporte de manera más robusta y no falle en caso de errores inesperados.

6. Comentarios:

El programa funciona correctamente bajo los parámetros actuales. Sin embargo, se pueden hacer algunos ajustes para mejorar:

- Ajustar los tiempos de espera entre preguntas para hacer la simulación más realista.

- Implementar un sistema de prioridades, donde algunos estudiantes puedan recibir atención más rápido dependiendo de la urgencia de sus preguntas.