Universidad Nacional Autónoma de México

Facultad de Ingeniería

Sistemas Operativos

Tarea 1. Ejercicio de sincronización

Medina Villa Samuel: 3049538

Ávila Martínez Alonso: 320237988

Fecha de entrega: 15 de octubre de 2025

Profesor: Gunnar Eyal Wolf Iszaevich

Los alumnos y el asesor

Un profesor de la facultad asesora a varios estudiantes, y estamos en su horario de atención.
Modelar la interacción durante este horario de modo que la espera (para todos) sea tan corta como sea posible.

Reglas

- Un profesor tiene 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 alumnos pueden tocar a su puerta en cualquier momento, pero no pueden entrar más de x alumnos
- Para evitar confundir al profesor, sólo un alumno puede presentar su duda (y esperar a su respuesta) al mismo tiempo.

  Los demás alumnos sentados deben esperar pacientemente su turno.
  Cada alumno puede preguntar desde 1 y hasta y preguntas (permitiendo que los demás alumnos pregunten entre una y otra)

Lenguaje y entorno de desarrollo

El problema fue desarrollado en C++.
Para poder ejecutar el programa se necesita del compilador g++ y un entorno para poder ejecutarlo (editor de código).
Se debe instalar un conjunto de herramientas como MinGW o MSYS2 (nuestro caso).
Ya que tenemos nuestro compilador instalado y nuestro entorno, ejecutamos este comando de compilación:

g++ main.cpp -o alumnos-asesor -pthread

Para ejecutar el programa alumnos-asesor usamos el siguiente comando

./alumnos-asesor

Estrategia de sincronización

El programa implementa el patrón de productor-consumidor.
Los productores (alumnos): La función `ask` representa a los alumnos que después de un tiempo aleatorio genera un dato (una pregunta, encapsulada en la cola `questions`) y lo depositan para que sea procesado.
El consumidor (asesor): La función `answer` representa al asesor, su tarea es consumir el dato (atender la pregunta del alumno) sacándolo de la cola
El recurso compartido (fila): La cola `questions` es la fila de espera que almacena temporalmente las preguntas generadas por los alumnos hasta que el profesor las pueda tomar.

Para que esta fila funcione de forma segura y eficiente se emplean estos mecanismos:

- La variable `std::mutex mtx` actúa como un candado que protege el recurso compartido (fila). Antes de que el asesor o un alumno intente modificar la cola (`questions`) se debe de tomar el candado mediante `std::unique_lock.`
  Esto con el fin de garantizar que solo un hilo a la vez pueda manipular la cola.
- La variable `std::condition_variable cv` es el sistema de alarma y sueño del asesor y lo evita a desperdiciar tiempo verificando si hay preguntas.
  El sueño del asesor `cv.wait` revisa la cola y la encuentra vacía, no se queda en un bucle, en su lugar llama a `cv.wait ( )` lo que lo pone a dormir de inmediato y libera el candado `mtx` para que los alumnos puedan entrar. El asesor solo se despierta si hay una notificación y la cola no está vacía.

  Cada vez que un alumno añade una pregunta a la cola, inmediatamente llama a `cv.notify_one()` Esta es la alarma que despierta al asesor permitiendo tomar el candado, revisar el trabajo y comenzar a contestar la pregunta.

  En el programa se usa un mecanismo de conteo con la variable `current_students` para limitar el número de hilos de los alumnos que puedan estar activos a la vez (máximo 4) controlando el acceso a la clase.