Sistema Operativos II

Prof. Saúl Zalimben

Trabajo de Práctico

Concurrencia

Indicaciones

- El trabajo puede ser realizado en grupos de hasta 2 personas, el nombre de los integrantes debe estar definido dentro del documento.
- Para evitar confusiones, todos los integrantes deben subir el trabajo.
- Debe subir el documento en la plataforma, en la tarea creada a dicho fin.
- El archivo deberá tener como nombre el número de documento del alumno, nombre y apellido [3564325_carlos_moreno].
- El formato de entrega es ZIP.

Criterios de Evaluación

1. Contenido
2. Uso correcto del idioma, ortografía, orden y pulcritud
3. Expresa sus ideas de forma clara y simple

Investigo y respondo las siguientes preguntas

Una situación cotidiana paralelizable

Los eventos en el mundo en que vivimos se producen de forma masivamente paralela, por lo cual, la concurrencia puede presentarse en muchos casos, en ámbitos no necesariamente ajenos al cómputo.

Por tanto, es de esperarse que –si existieran las primitivas de sincronización en el mundo real– podríamos aplicar los mecanismos de sincronización de los sistemas operativos.

Identificación y descripción del problema

Identificar un proceso de la vida real que presente características de concurrencia. Normalmente, cualquier proceso que presente eventos realizados por múltiples actores será suficiente.

Describan en un documento de texto la naturaleza del problema que eligieron:

- Describan la situación que moldearán

Un profesor de colegio enfrenta un desafío de gestión de consultas de estudiantes durante sesiones de investigación. El sistema inicial de atención era completamente aleatorio, generaba:

- Desorganización en las consultas.
- Posible frustración en estudiantes.
- Tiempos de espera indeterminados, aprovechar mal el horario de trabajo.
- Reprobar alumnos interesados en realizar las actividades.

- ¿Dónde pueden verse las consecuencias nocivas de la concurrencia? ¿Qué eventos pueden ocurrir que queramos controlar?

- Múltiples estudiantes intentando consultar simultáneamente.
- Posible inanición, estudiantes sin ser atendidos nunca.
- Conflictos en el orden de atención.
- Saturación de recursos (profesor).

- ¿Hay eventos concurrentes para los cuales el ordenamiento relativo no resulta importante?

- Llegada de estudiantes para consulta. Solicitudes simultáneas.
- Tiempo de consulta variable.

Implementación del modelo e introducción de mecanismos de sincronización

- Implementen, en el lenguaje de su preferencia, un programa que simule la situación descrita, necesariamente empleando hilos o procesos simultáneos (es criterio mínimo de aceptación del proyecto).
- Identifiquen qué puntos de sincronización son necesarios, y qué patrones requieren para ello. Naturalmente, estos deben ser implementados.

Documentación

El programa debe ir acompañado de documentación que incluya, además de lo mencionado en la primera sección (Identificación y descripción del problema):

- Descripción de los mecanismos de sincronización empleados

Semáforos:

- teacher_available: Controla acceso al profesor.
- student_ready: Indica estudiantes en espera.

Mutex:

- queue_mutex: garantiza acceso individual a la cola.

Eventos:

- simulation_complete: Control de finalización de simulación.

- Lógica de negocio

1. Estudiantes llegan y se encolan.
2. Profesor atiende en orden de llegada.
3. Tiempo de consulta es aleatorio, depende de cada caso presentado por el estudiante.
4. Sistema maneja los elementos en cola.

- Identificación del estado compartido (variables o estructuras globales)

Variables globales:

- consultation_queue: Cola de estudiantes.
- teacher_available: Estado del profesor.
- student_ready: Estudiantes esperando.

- Descripción algorítmica del avance de cada hilo/proceso
- Descripción de la interacción entre ellos (sea mediante los mecanismos de sincronización o de alguna otra manera)

Hilo de Estudiantes:

1. Crear instancia de estudiante.
2. Intentar entrar en cola.
3. Liberar semáforo si logra entrar

Hilo de profesor:

1. Esperar señal del estudiante.
2. Extraer estudiante de cola.
3. Simular consulta.
4. Liberar recurso.

- Descripción del entorno de desarrollo, suficiente para reproducir una ejecución exitosa
- ¿Qué lenguaje emplean? ¿Qué versión?
- ¿Qué bibliotecas más allá de las estándar del lenguaje?
- ¿Bajo qué sistema operativo / distribución lo desarrollaron y/o probaron?

Lenguaje: Python 3.12
IDE: Pycharm Community Edition
Sistema Operativo: Windows 11 Pro
Bibliotecas: estándares: threading, queue, dataclasses, logging.

- Ejemplos o pantallazos de una ejecución exitosa

2024-12-02 13:42:12,305 - Student-1 - Estudiante 1 entra a la cola de espera

2024-12-02 13:42:12,306 - Professor - Profesor atiende a estudiante 1
2024-12-02 13:42:12,420 - Student-2 - Estudiante 2 entra a la cola de espera
2024-12-02 13:42:12,489 - Student-3 - Estudiante 3 entra a la cola de espera
2024-12-02 13:42:12,598 - Student-4 - Estudiante 4 entra a la cola de espera
2024-12-02 13:42:12,663 - Professor - Consulta con estudiante 1 finalizada
2024-12-02 13:42:12,663 - Professor - Profesor atiende a estudiante 2
2024-12-02 13:42:12,777 - Student-5 - Estudiante 5 entra a la cola de espera
2024-12-02 13:42:12,863 - Student-6 - Estudiante 6 entra a la cola de espera
2024-12-02 13:42:12,951 - Student-7 - Estudiante 7 entra a la cola de espera
2024-12-02 13:42:13,437 - Professor - Consulta con estudiante 2 finalizada
2024-12-02 13:42:13,437 - Professor - Profesor atiende a estudiante 3
2024-12-02 13:42:13,437 - Student-8 - Estudiante 8 entra a la cola de espera
2024-12-02 13:42:13,728 - Student-9 - Estudiante 9 no pudo entrar - Cola llena
2024-12-02 13:42:13,819 - Professor - Consulta con estudiante 3 finalizada
2024-12-02 13:42:13,819 - Professor - Profesor atiende a estudiante 4
2024-12-02 13:42:13,820 - Student-10 - Estudiante 10 entra a la cola de espera
2024-12-02 13:42:14,433 - Professor - Consulta con estudiante 4 finalizada
2024-12-02 13:42:14,433 - Professor - Profesor atiende a estudiante 5
2024-12-02 13:42:15,058 - Professor - Consulta con estudiante 5 finalizada
2024-12-02 13:42:15,058 - Professor - Profesor atiende a estudiante 6
2024-12-02 13:42:15,772 - Professor - Consulta con estudiante 6 finalizada
2024-12-02 13:42:15,772 - Professor - Profesor atiende a estudiante 7
2024-12-02 13:42:16,431 - Professor - Consulta con estudiante 7 finalizada
2024-12-02 13:42:16,431 - Professor - Profesor atiende a estudiante 8
2024-12-02 13:42:16,970 - Professor - Consulta con estudiante 8 finalizada
2024-12-02 13:42:16,970 - Professor - Profesor atiende a estudiante 10
2024-12-02 13:42:17,308 - Professor - Consulta con estudiante 10 finalizada

Ejemplos. (No utilizar porque les saco puntos)

- Reserva de boletos de vuelo de avión
- Reserva de habitaciones de un hotel
- Actualización de stock
- Ventas de ticket de conciertos

Entrega

- Código fuente: Programa que resuelve el problema planteado
- Documentación (PDF)

Evaluación

La siguiente rúbrica presenta los diferentes puntos que serán evaluados en el trabajo práctico.

	Excelente (100%)	**Bueno (75%)**	**Suficiente (50%)**	**Insuficiente (0%)**
Requisitos				
Cumplimiento	Los requisitos planteados se cubren al 100%, incluso exceden lo solicitado	Los requisitos se cumplen satisfactoriamente	El proyecto se aproxima a los requisitos, sin llegar a cumplirlos por completo	El proyecto no tiene relación con lo solicitado
Proyecto				
Creatividad	El problema abordado es novedoso, o la manera en que aborda el planteamiento es original		El planteamiento de la situación resulta directo, su resolución es casi obvia,	
Documentación				

Documentación expresa	La documentación incluye a los nombres de los participantes, describe suficientemente la situación a implementar, la lógica que implementa (cómo lo resuelve), y presenta ejemplos de invocación	Falta uno de los elementos requeridos	Faltan dos o más de los elementos requeridos	No se entregó documentación expresa
Entorno y dependencias	Detalla el entorno para el cual el programa fue escrito, detallando según sea pertinente lenguaje (incluyendo la implementación y versión), principales módulos o bibliotecas que deben ser instalados (indicando sus respectivas versiones), y demás instrucciones pertinentes		Indica los principales componentes requeridos para la construcción y ejecución del proyecto, pero omite detalles importantes que dificultan su exitosa ejecución.	

Comentarios	El código está comentado donde hace falta, no repite información obvia. Los comentarios ayudan a comprender la lógica, no detalles específicos de la implementación	El código está comentado donde hace falta, pero los comentarios son excesivos.	Hay algunos comentarios en el programa, pero falta mucho para que ayuden a su comprensión.	No hay comentarios.
Entrega				
Código válido	Al ejecutar las instrucciones documentadas, el código puede ejecutarse exitosamente al primer intento.	Las instrucciones que forman parte de la documentación tienen que adecuarse para poder ejecutar el código, o hay errores menores que corregir para que funcione.	No está documentado cómo ejecutar el código, o hay errores mayores que corregir para poder ejecutarlo.	Resultó imposible probar la ejecución.
Legibilidad				
Estructura	El código está bien organizado y emplea un estilo de indentación de forma consistente.	El código está mayormente ordenado; hay inconsistencias menores.	Falta claridad en los bloques por no emplear indentación o hacerlo de forma	

			absolutamente inconsistente.	
Nomenclatura	Los nombres de los símbolos (variables, funciones, métodos, clases) son claros y acorde a su función; los principales elementos están documentados expresamente.	Los nombres de los símbolos mencionados son claros y acorde a su función, aunque no estén documentados.	Los nombres de los símbolos no son claros, pero su uso y significado forma parte de la documentación.	Cuesta trabajo seguir la lógica; los símbolos empleados no tienen nombres significativos, y su función no está documentada.