Sistema Operativos II

Prof. Saúl Zalimben

Trabajo de Práctico

Concurrencia Clásica

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

Considerando los ejercicios clásicos de concurrencia. Implementar **una solución a uno de los siguientes problemas**. (The Little Book of Semaphores)

- Productores-Consumidores
- Lectores-Escritores
- Cena de los Filósofos
- Fumadores Compulsivos
- La cena de los caníbales

Entrega

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

Consideraciones

- Puede implementarlos en cualquier lenguaje de programación.
- Incluya un documento que contenga la siguiente información:
    - Descripción de los mecanismos de sincronización empleados.
    - Identificación del estado compartido (variables o estructuras globales).

Semáforos (estructuras de sincronización):

- agent_semaphore: Sincroniza al agente con los fumadores.
- smoker_semaphore: Un semáforo por cada fumador para despertar al correcto.
- mutex: exclusión mutua para acceso a recursos compartidos.

Variables compartidas (variables globales de sincronización):

- current_ingredients: Conjunto de ingredientes actuales.
- Ingredients: Lista de ingredientes posibles.

    - Lógica de negocio o línea de pensamiento para arreglar el problema.

El agente selecciona dos ingredientes al azar;
Un fumador (el que no tiene ninguno de esos dos ingredientes) puede fumar;
El fumador fuma y notifica al agente para continuar el ciclo;

Proceso:

1. Agente selecciona dos ingredientes al azar.
2. Agente despierta al fumador con el ingrediente faltante. Mientras todos están esperando a ser despertados.
3. Fumador verifica que puede fumar, fuma.
4. Fumador notifica al agente que ha fumado mientras el agente espera la confirmación.
5. Se repite el proceso.

    - Descripción algorítmica del avance de cada hilo/proceso (mínimo 3).

Obs: antes de iniciar la ejecución se hace una consulta al usuario de cuántas veces quiere simular este proceso.

HILO AGENTE

```
MIENTRAS verdadero:
  Bloquear mutex
  Seleccionar dos ingredientes aleatorios
  Establecer current_ingredients
```

Liberar mutex

Identificar ingrediente faltante
Despertar semáforo del fumador con ingrediente faltante

Esperar confirmación (acquire en agent_semaphore)
Pausar brevemente

HILO fumador cada uno con un ingrediente diferente, esperando que se active su semáforo para realizar proceso:
MIENTRAS verdadero:
Esperar en semáforo propio (esperando ser despertado)

Bloquear mutex
Validar que puede fumar
Limpiar ingredientes actuales
Desbloquear mutex

Simular tiempo de fumado
Notificar al agente que ha terminado

- Descripción de la interacción entre ellos (sea mediante los mecanismos de sincronización o de alguna otra manera)

Cada fumador tiene un semáforo individual
El agente libera el semáforo del fumador que puede fumar
Los fumadores esperan previamente en su semáforo

El mutex protege el acceso a current_ingredients, garantiza que solo un hilo modifique el estado compartido, previene condiciones de carrera.

El flujo de comunicación:
Agente selecciona ingredientes, agente despierta fumador específico, fumador fuma, fumador notifica al agente, ciclo se repite.

- Descripción del entorno de desarrollo, suficiente para reproducir una ejecución exitosa (Código fuente/GitHub, no enlace).

Python 3.12, Pycharm Community Edition.

- ¿Qué lenguaje emplean? ¿Qué versión?

Python 3.12

- ¿Qué bibliotecas más allá de las estándar del lenguaje?

Solo threading, que es estándar

- ¿Bajo qué sistema operativo / distribución lo desarrollaron y/o probaron? (Recomendable GBU/Linux)

Windows 12.

- Ejemplos o pantallazos de una ejecución exitosa

Ingrese el número de iteraciones para la simulación de fumadores: 5
Iteración 1 - Agente provee: ['tabaco', 'fósforos']
Fumador con papel puede fumar
Fumador con papel ha fumado
Iteración 2 - Agente provee: ['fósforos', 'tabaco']
Fumador con papel puede fumar
Fumador con papel ha fumado
Iteración 3 - Agente provee: ['fósforos', 'papel']
Fumador con tabaco puede fumar
Fumador con tabaco ha fumado
Iteración 4 - Agente provee: ['fósforos', 'papel']
Fumador con tabaco puede fumar
Fumador con tabaco ha fumado
Iteración 5 - Agente provee: ['papel', 'fósforos']
Fumador con tabaco puede fumar
Fumador con tabaco ha fumado