UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL MAR DEL PLATA
ARQUITECTURA Y SISTEMAS OPERATIVOS
1er Año – 2do Cuatrimestre
PROFESORES: G. GIMENEZ – G.ORELLANO - G. ACIERNO

Trabajo Práctico No 4: Concurrencia [Parte II]

1. Se tienen 3 hilos: H1, H2 y H3: El código del hilo Hi (i=1,2,3) es:

```
Hi()     /* Hilo Hi con i=1,2,3 */
{
      While (true)
          printf('Soy el hilo i \n");
}
```

Se desea mostrar en la salida lo siguiente:

Soy el hilo 1
Soy el hilo 2
Soy el hilo 3
Soy el hilo 1
Soy el hilo 2
……..

Se debe imprimir N veces cada hilo. Sincronizar mediante Semáforos. Implementar en máquina.

2. Tomando el ejercicio 4 del Práctico 2. Implemente un semáforo en la impresión en una línea de cada uno de los threads. Implementar en máquina.

3. Se tienen 3 procesos A, B y C. Implementar en máquina.
   a)Construya el código con semáforos de manera tal que la secuencia sea ABC, ABC,ABC,.....
   b)¿Y si quiero que la secuencia sea BCA, BCA, BCA,....?

4. ¿Tiene algún problema el siguiente programa? ¿Porque?

P1	**P2**
sem_wait(Sem1); **sem_wait(Sem2);** **// Sección Crítica //** **sem_post(Sem2);** **sem_post(Sem1);**	**sem_wait(Sem2);** **sem_wait(Sem1);** **// Sección Crítica //** **sem_post(Sem1);** **sem_post(Sem2);**

5. Sincronizar A y B de tal manera que (X es variable global). Implementar en máquina.
   a) Siempre el resultado de la ejecución sea 200 y 50.
   b) Siempre el resultado de la ejecución sea 50 y 200.

A	**B**
x=199; **x=x+1;** **Print(x);**	**x=500;** **x=X/10;** **Print(x);**

6. Sean tres semáforos: A y C inicializados a 0, y B inicializado a 1. Dados tres procesos que acceden a dichos semáforos de la siguiente forma. Implementar en máquina.

Proceso 1	**Proceso 2**	**Proceso 3**
Wait (A);	**Wait (B);**	**Wait(B);**
		Wait(C);
Post(C);	**Post(A);**	**Post(B);**
	Post(B);	

   Indicar una planificación de los mismos que les permitan terminar, y otra que conduzca a un interbloqueo.

7. **Productor-Consumidor**. El programa (pseudocódigo) describe dos procesos, productor y consumidor, ambos comparten un buffer de tamaño finito. La tarea del productor es generar un producto, almacenarlo y comenzar nuevamente; mientras que el consumidor toma (simultáneamente) productos uno a uno. El problema consiste en que el productor no añada más productos que la capacidad del buffer y que el consumidor no intente tomar un producto si el buffer está vacío.

int itemCount = 0; *//Variable Global*
const BUFFER_SIZE = 5; *//Tamaño del buffer*

Proceso Productor	**Proceso Consumidor**

void producer() {
  while (true) {
  item = produceItem();
 	
  if (itemCount == BUFFER_SIZE) {
  sleep();
  }
 	
  putItemIntoBuffer(item);
  itemCount = itemCount + 1;
 	
  if (itemCount == 1) {
  wakeup(consumer);
  }
  }
}
	void consumer() {
	  while (true) {
	
	  if (itemCount == 0) {
	  sleep();
	  }
	
	  item = removeItemFromBuffer();
	  itemCount = itemCount - 1;
	
	  if (itemCount == BUFFER_SIZE - 1) {
	  wakeup(producer);
	  }
	
	  consumeItem(item);
	  }
	}

Nota:

- *La función **wakeup** despierta al proceso pasado por parámetro y la función **sleep** duerme al proceso que lo llama.*
- *La función **produceItem()** produce un item.*
- *La función **putItemIntoBuffer(item)** inserta el item en el buffer.*
- *La función **removeItemFromBuffer()** remueve el item del buffer para consumir.*
- *La función **consumeItem(item)** consume el item pasado por parámetro.*

A) ¿Presenta algún problema el código anterior si se ejecutara de manera concurrente? Analice el código.

B) Si la respuesta es negativa, explique el porqué. Si la respuesta es positiva, muestre la secuencia donde se genera el problema (Trace del programa).

8. Dado el problema del Productor-Consumidor y los siguientes algoritmos:

Productor	Consumidor
REPETIR COMIENZO producir un elemento; WAIT (espacio libre); WAIT (manipulación del buffer); depositar un elemento en el buffer; SIGNAL (manipulación del buffer); SIGNAL (elemento disponible); FIN;	REPETIR COMIENZO WAIT (manipulación del buffer); WAIT (elemento disponible); sacar elemento del buffer; SIGNAL (manipulación del buffer); SIGNAL (espacio libre); consumir elemento; FIN;

Inicialice los semáforos según corresponda. ¿Es válida la solución anterior? ¿Por qué?

9. **Trabajo Práctico Especial -** Enunciado

Mirtha LeBig tiene un popular programa de televisión en el que invita a importantes referentes del show business y de la derecha argentina a almorzar y conversar sobre sus vidas y sobre temas de actualidad. La dinámica del programa es la siguiente:

• Al programa asisten N invitados. Los comensales se sientan a la mesa, siendo Mirtha la última en sentarse.

• Una vez que se sentó Mirtha, los M mozos sirven la comida. Por problemas de presupuesto, la cantidad de mozos es siempre menor a la cantidad de invitados.

• Los mozos no deben servir más de **N** platos. Cualquier comensal (incluida Mirtha) **no puede comenzar a comer**, incluso si no han terminado de servir a todos los invitados.

• Cuando Mirtha termina de comer, ella lanza dos pregunta polémica. Esta pregunta puede ser respondida **por cualquier invitado que haya o no terminado de comer**. Sólo un invitado debe responder cada una de las pregunta. Además para realizar la segunda pregunta polémica, se debe haber respondido la primera.

• Para mantener alto el rating, Mirtha se enoja muchísimo por la segunda respuesta polémica, se levanta y se va del estudio. Si Mirtha ya se ha levantado, los invitados pueden hacer lo mismo, pero sólo si ya han terminado de comer.

Las primitivas disponibles son:

• sentarse()
• servir_comida()
• comer()
• enojarse()
• levantarse()
• lanzar_pregunta_polemica()
• lanzar_respuesta_polemica()

Consignas:

1. Realice una solución usando semáforos para el problema descrito anteriormente.
2. Explicar brevemente cómo funciona la solución indicando qué semáforos necesitó y para qué; y cuántos procesos diferentes tuvo.
3. ¿En algún punto de la solución se puede producir inanición? ¿Dónde? ¿Por qué?