Elementos para resolver problemas de sincronización:

a) Hilos o procesos
b) Semáforos y áreas de memoria compartida
c) Buzones
d) Monitores y variables de condición
e) Regiones críticas condicionales
f) Memoria transaccional (atomic)

Varios problemas tienen las sugerencias de su versión original, algunas en inglés, pueden tomarlas o no para ofrecer su solución.

1) En una carretera de dos carriles que discurre de Norte a Sur (y de Sur a Norte) hay un **puente** estrecho de un solo carril. Por ese puente solo pueden circular vehículos en un solo sentido al mismo tiempo, con la regla de que, estando los vehículos circulando en un sentido, cuando haya N o más vehículos (N es un parámetro del problema) esperando a cruzar en sentido contrario, se impide que entren nuevos vehículos al puente en el sentido actual de la marcha, para permitir cambiar ésta cuanto antes. En el caso de que haya N o más vehículos esperando a cruzar en cada uno de los dos sentidos, se debe dar prioridad a los vehículos que suben, es decir, que van de Sur a Norte

Se pide escribir un objeto protegido dotado de cuatro operaciones (Entrar_Norte, Salir_Norte, Entrar_Sur, Salir_Sur) que implemente el protocolo anterior.

2) Un río es compartido por misioneros y caníbales, solo se utiliza un bote de tres espacios para cruzar el río. Con el fin de garantizar la integridad de los misioneros, no se pueden colocar dos caníbales y un misionero juntos en el mismo viaje, las demás combinaciones son válidas. Cada personaje está representado por un hilo, Caníbal y Misionero. Se sugiere la creación de dos métodos **LlegaM**() y **LlegaC**() que llaman los hilos respectivamente, ambos se encargan de realizar la sincronización. Puede existir otro método denominado **Cruzar**() que se llama cuando el bote está listo para cruzar.

3) Se dispone de un sistema de 3 procesos **fumadores** y uno estanquero (Patil-1971). El estanquero dispone de los tres ingredientes que son necesarios para liar un cigarrillo: papel, tabaco y mechero en cantidades infinitas. El funcionamiento del sistema es como sigue:

a) Cada fumador está continuamente liando un cigarrillo y después se lo fuma. Para ello necesita obtener una unidad de cada uno de los tres ingredientes
b) El estanquero pone en una mesa dos unidades cada vez, correspondientes a dos ingredientes aleatorios
c) El fumador que tiene el ingrediente que falta, lía un cigarrillo y se lo fuma. El estanquero repone los ingredientes consumidos inmediatamente por otros dos

4) El problema del **barbero dormilón**

Una barbería tiene una sala de espera con n sillas y una habitación con un sillón donde se atiende a los clientes. Si no hay clientes el barbero se duerme. Si un cliente entra en la barbería pero todas las sillas están ocupadas, entonces se va, dejando la barbería. Si el barbero esta ocupado entonces el cliente se sienta en una de las sillas disponibles. Si el barbero esta dormido el cliente lo despertará. Escribir una aplicación que coordine el barbero y sus clientes de forma adecuada utilizando semáforos.

5) **Santa Claus** tiene dos talleres pequeños, en los que sólo pueden trabajar dos duendes a la vez. Nos referiremos a estos talleres como H y L, donde se fabrican juguetes de alta tecnología (H) y de poca tecnología (L). Existen cuatro duendes que pueden hacer juguetes para Santa en estos talleres. Llamaremos a los duendes G, D, T y J. Los duendes gastan su tiempo trabajando o durmiendo. En cualquier momento, un duende que esté durmiendo puede ser despertado por Santa para asignarle un trabajo particular en un taller determinado. Vamos a escribir esta solicitud como `Santa->inicia_trabajo( duende, taller )`. Cuando un duende quiere dejar el taller debe solicitarlo a Santa primero. Vamos a escribir esta solicitud como `Santa->termina_trabajo( duende, taller )`. Para cada tipo de solicitud Santa puede concederla o, en su defecto, hacer que el duende espere hasta que la solicitud pueda ser concedida. Esta es una lista de las reglas que Santa tiene designadas para asegurar la máxima productividad.

a) No pueden estar más de dos duendes en cada taller
b) El duende G no puede trabajar con otro en el mismo taller
c) El duende D nunca puede estar solo en un taller
d) Los duendes T y J, por sus constantes pleitos, no pueden estar juntos en el mismo taller

Muestre cómo implantar el taller de Santa por medio de un monitor.

6) Una alcancía es compartida por varios hilos, ésta posee una propiedad denominada “saldo” y dos métodos **Retirar**(monto) y **Guardar**(monto). El método Guardar(monto) simplemente suma el monto indicado como parámetro al saldo, eventualmente despierta a los procesos Retirar(monto) que se han quedado suspendidos porque no había suficientes fondos al momento de hacer el retiro. El método Retirar(monto) intenta sacar de la alcancía el monto indicado como parámetro, pero si no hay suficientes fondos el hilo espera hasta que su solicitud pueda ser satisfecha. Su tarea es implantar ambos métodos utilizando **regiones críticas condicionales** de manera que se sincronicen los hilos que utilizan la alcancía.

7) En el parque nacional Kruger en Sudáfrica hay un cañón muy profundo con una simple cuerda para cruzarlo. Los **babuinos** necesitan cruzar ese cañón constantemente en ambas direcciones y pueden hacerlo poniendo una mano delante de otra de forma sucesiva. Como los babuinos son muy agresivos, si dos de ellos se encuentran en cualquier punto de la cuerda yendo en direcciones opuestas, estos se pelearán y terminarán cayendo por el cañón y muriendo. Por otro lado, la cuerda no es muy resistente y aguanta a un máximo de **cinco**

babuinos simultáneamente. Si en cualquier instante hay más de cinco babuinos en la cuerda, ésta se romperá y los babuinos caerán también al vacío.
Suponiendo que podemos enseñar a los babuinos a usar semáforos, queremos diseñar un sistema de sincronización que:

1. No permita más de cinco babuinos en la cuerda
2. Asegura que, una vez que un babuino ha cogido la cuerda, ningún otro babuino comenzará a recorrer la cuerda en la dirección opuesta
3. No es necesario prevenir la inanición, es decir, una fila interminable de babuinos en una dirección puede provocar inanición sobre los babuinos que pretenden recorrer la cuerda en la otra dirección. Se supone que nunca existirá una fila interminable de babuinos. Existe una solución que previenen la inanición pero es muy ineficaz en el uso del recurso

Para ello es necesario crear un programa que lance aleatoriamente en el tiempo procesos B*abuinoVa*() y B*abuinoViene*(). También es necesario que los procesos B*abuinoVa*() y B*abuinoViene*(int) dispongan de la sincronización anteriormente indicada. Asumir la existencia de las funciones: C*ruzarCuerda*() y N*uevoBabuino*(). La última de estas funciones espera un tiempo determinado y devuelve 1 o 0 dependiendo de la dirección de cruce del siguiente babuino que va a aparecer. Para la sincronización de los procesos, utiliza variables compartidas y semáforos. De igual forma, se utilizarán las funciones up(semaforo), up(semaforo,n), down(semaforo) y down(semaforo,n) para manipular los semáforos.

8) Sea un **bar** de dudosa reputación, que sirve bebidas alcohólicas sin permiso del ayuntamiento. La barra del bar está atendida por **tres** camareros, que realizan, a petición de los clientes, mezclas de **seis** bebidas (Whisky, Ginebra, Cognac, Anís del Mono, Cas y Zumo de Piña). Los clientes, piden su consumición, que será una combinación cualquiera de dos de las seis bebidas, a un camarero libre, si lo hay. Si todos los camareros están ocupados, esperan a que se libere uno de ellos. El camarero, una vez recibida la petición, coge las botellas correspondientes y hace la mezcla. Mientras tanto, si otro camarero recibe una nueva petición que requiere el uso de una de las botellas que se está utilizando, espera pacientemente a que el otro camarero termine de hacer su mezcla, para proceder con la suya.

Se supone que el máximo número de clientes distintos que se pueden atender en un día está limitado a 100, aunque cada cliente puede pedir tantas combinaciones –gratis– como quiera.

La policía, sospechando que en el bar se distribuye alcohol sin licencia, puede entrar en cualquier momento en el bar. Cuando así ocurre, los tres camareros sirven a todo el mundo Cas con Zumo de Piña, independientemente de lo que hayan pedido.

Escriba los procedimientos *camarero*(), *cliente*(), *policia*() que ejecuten las funciones descritas. Para la sincronización de los procesos, utiliza variables compartidas y semáforos.

9) **Santa Claus** es un hombre muy mayor y se pasa la mayor parte del tiempo durmiendo en su casa del Polo Norte. Sólo se despierta si vuelven los **nueve** renos de sus vacaciones en el Pacífico Sur o si hay varios duendes que tienen alguna dificultad con la fabricación de juguetes. Para permitir que Santa Claus duerma lo máximo posible, los duendes sólo lo despertarán si hay **tres duendes** que tengan problemas. Cuando hay más de tres duendes con problemas, sólo los tres primeros lo despiertan y los demás esperan a que Santa Claus resuelva los problemas de los tres anteriores. Sin embargo, si cuando Santa Claus se despierta además de los tres duendes, ha llegado también el último reno de sus vacaciones en el trópico, Santa decide que los duendes pueden esperar hasta después de Navidad porque es más importante preparar su trineo y repartir los juguetes (se asume que los renos no quieren dejar el trópico hasta el último momento). Cuando llega el último reno es éste el que despierta a Santa Claus mientras que los anteriores esperan en un cobertizo caliente hasta que llegue el último de ellos. El número de renos es siempre nueve pero el número de duendes no está determinado.

Además se tienen que cumplir las siguientes condiciones:

a) Cuando están los nueve renos Santa Claus invoca a la función preparaTrineo() y los renos a atiendeTrineo()
b) Cuando hay tres duendes que necesitan ayuda Santa Claus invoca a la función ayudaDuendes() y los duendes a recibeAyuda()
c) Los tres duendes deben de llamar a la función recibeAyuda() antes de que llegue otro duende (antes de que se incremente el contador de duendes)

Escribir los procesos reno(), duende() y santa(). Para la sincronización de los procesos, utiliza variables compartidas y semáforos.

10) **Bees and semaphores**

Abejas amistosas (N) están recolectando miel para un oso atrapado. Por el momento la vida del oso es solamente tomar miel y dormir. Las abejas le llevan miel a su jarro, una porción a la vez hasta que el jarro se llene. Cuando el jarro está lleno (J porciones de miel), la abeja que deposita la última porción despierta al oso para que coma. El oso comienza a tomar la miel y mientras tanto las abejas detienen el proceso de llenado del jarro hasta que el oso haya finalizado toda la miel y el jarro se encuentra vacio. Entonces el oso comienza a dormir de nuevo y las abejas comienzan a depositar la miel en el jarro de nuevo.

Las abejas pueden depositar la miel en el jarro de manera concurrente, pero nunca se debe exceder su capacidad de J porciones.

Brinde una solución para lograr que las abejas rellenen el jarro con miel sin sobrepasar su capacidad y que el oso pueda tomar la miel cuando el jarro contenga las porciones establecidas. Utilice semáforos para esta solución. Explique cuáles situaciones del problema exigen exclusión mutua y sincronización y cómo fueron resueltas en su solución.