# Práctica 2 – Semáforos

**CONSIDERACIONES PARA RESOLVER LOS EJERCICIOS:**

- Los semáforos deben estar declarados en todos los ejercicios.
- Los semáforos deben estar inicializados en todos los ejercicios.
- No se puede utilizar ninguna sentencia para ***setear*** o ***ver*** el valor de un semáforo.
- Debe evitarse hacer ***busy waiting*** en todos los ejercicios.
- En todos los ejercicios el tiempo debe representarse con la función ***delay***.

1. Existen N personas que deben ser chequeadas por un detector de metales antes de poder ingresar al avión.
   a. Implemente una solución que modele el acceso de las personas a un detector (es decir si el detector está libre la persona lo puede utilizar caso contrario debe esperar).
   b. Modifique su solución para el caso que haya tres detectores.

2. Un sistema operativo mantiene 5 instancias de un recurso almacenadas en una cola, cuando un proceso necesita usar una instancia del recurso la saca de la cola, la usa y cuando termina de usarla la vuelve a depositar.

3. Suponga que existe una BD que puede ser accedida por 6 usuarios como máximo al mismo tiempo. Además, los usuarios se clasifican como usuarios de prioridad alta y usuarios de prioridad baja. Por último, la BD tiene la siguiente restricción:
   - no puede haber más de 4 usuarios con prioridad alta al mismo tiempo usando la BD.
   - no puede haber más de 5 usuarios con prioridad baja al mismo tiempo usando la BD.

   Indique si la solución presentada es la más adecuada. Justifique la respuesta.

```
Var
    sem: semaphoro := 6;
    alta: semaphoro := 4;
    baja: semaphoro := 5;
```

```
Process Usuario-Alta [I:1..L]::
  {   P (sem);
      P (alta);
      //usa la BD
      V(sem);
      V(alta);
  }
```

```
Process Usuario-Baja [I:1..K]::
  {   P (sem);
      P (baja);
      //usa la BD
      V(sem);
      V(baja);
  }
```

4. Existen N personas que deben imprimir un trabajo cada una. Resolver cada ítem usando semáforos:
   a) Implemente una solución suponiendo que existe una única impresora compartida por todas las personas, y las mismas la deben usar de a una persona a la vez, sin importar el orden. Existe una función *Imprimir(documento)* llamada por la persona que simula el uso de la impresora. Sólo se deben usar los procesos que representan a las Personas.
   b) Modifique la solución de (a) para el caso en que se deba respetar el orden de llegada.
   c) Modifique la solución de (a) para el caso en que se deba respetar estrictamente el orden dado por el identificador del proceso (la persona X no puede usar la impresora hasta que no haya terminado de usarla la persona X-1).
   d) Modifique la solución de (b) para el caso en que además hay un proceso Coordinador que le indica a cada persona que es su turno de usar la impresora.
   e) Modificar la solución (d) para el caso en que sean 5 impresoras. El coordinador le indica a la persona cuando puede usar una impresora, y cual debe usar.

5. Suponga que se tiene un curso con 50 alumnos. Cada alumno debe realizar una tarea y existen 10 enunciados posibles. Una vez que todos los alumnos eligieron su tarea, comienzan a realizarla. Cada vez que un alumno termina su tarea, le avisa al profesor y se queda esperando el puntaje del grupo, el cual está dado por todos aquellos que comparten el mismo enunciado. Cuando un grupo terminó, el profesor les otorga un puntaje que representa el orden en que se terminó esa tarea de las 10 posibles.
   ***Nota:*** Para elegir la tarea suponga que existe una función ***elegir*** que le asigna una tarea a un alumno (esta función asignará 10 tareas diferentes entre 50 alumnos, es decir, que 5 alumnos tendrán la tarea 1, otros 5 la tarea 2 y así sucesivamente para las 10 tareas).

6. A una empresa llegan E empleados y por día hay T tareas para hacer (T>E), una vez que todos los empleados llegaron empezaran a trabajar. Mientras haya tareas para hacer los empleados tomarán una y la realizarán. Cada empleado puede tardar distinto tiempo en realizar cada tarea. Al finalizar el día se le da un premio al empleado que más tareas realizó.

7. Resolver el funcionamiento en una fábrica de ventanas con 7 empleados (4 carpinteros, 1 vidriero y 2 armadores) que trabajan de la siguiente manera:
   - Los carpinteros continuamente hacen marcos (cada marco es armando por un único carpintero) y los deja en un depósito con capacidad de almacenar 30 marcos.
   - El vidriero continuamente hace vidrios y los deja en otro depósito con capacidad para 50 vidrios.
   - Los armadores continuamente toman un marco y un vidrio (en ese orden) de los depósitos correspondientes y arman la ventana (cada ventana es armada por un único armador).

8. A una cerealera van T camiones a descargarse trigo y M camiones a descargar maíz. Sólo hay lugar para que 7 camiones a la vez descarguen, pero no pueden ser más de 5 del mismo tipo de cereal. ***Nota:*** no usar un proceso extra que actué como coordinador, resolverlo entre los camiones.

9. En un curso hay dos profesores que toman examen en forma oral, el profesor A llama a los alumnos de acuerdo con el orden de llegada, mientras que el profesor B llama al de menor número de alumno (que esté esperando ser llamado para rendir). Existen N alumnos que llegan y se quedan esperando hasta ser llamados para rendir, luego de que uno de los dos profesores lo atiende, se va. Indicar si la siguiente solución realizada con semáforo resuelve lo pedido. Justificar la respuesta.

```
string estado[N] = ([N], "Esperando" )
queue colaA, colaB
sem llegoA, llegoB = 0
sem esperando[N] = ([N], 0)
sem mutexA, mutexB = 1
```

```
Alumno[i: 1..N]
{  P(mutexA)
   push(colaA, i)
   V(mutexA)
   V(llegoA)
   P(mutexB)
   push(colaB, i)
   V(mutexB)
   V(llegoB)
   P(esperando[i])
   if (estado[i] == "A")
      //Interactúa con el Prof A//
   else
      //Interactua con el Prof B//
   P(esperando[i])
}
```

```
Profesor A::
{  int  idAlumno
   while (true)
    {  P(llegoA)
       P(mutexA)
       idAlumno = pop(colaA)
       V(mutexA)
       If (estado[idAlumno] = ="Esperando")
         {   estado[idAlumno] = "A"
             V(esperando[idAlumno])
             //Se toma el examen//
             V(esperando[idAlumno])
         }
    }
}
```

```
Profesor B::
{  int idAlumno
   while (true)
    {  P(llegoB)
       P(mutexB)
       idAlumno = popAleatorio(colaB)
       V(mutex(B))
       If (estado[idAlumno] == "Esperando")
         {   estado[idAlumno] = "B"
             V(esperando[idAlumno])
             //Se toma el examen//
             V(esperando[idAlumno])
         }
    }
}
```

10. En un vacunatorio hay ***un empleado*** de salud para vacunar a ***50 personas***. El empleado de salud atiende a las personas de acuerdo con el orden de llegada y de a 5 personas a la vez. Es decir, que cuando está libre debe esperar a que haya al menos 5 personas esperando, luego vacuna a las 5 primeras personas, y al terminar las deja ir para esperar por otras 5. Cuando ha atendido a las 50 personas el empleado de salud se retira. ***Nota:*** todos los procesos deben terminar su ejecución; asegurarse de no realizar *Busy Waiting*; suponga que el empleado tienen una función *VacunarPersona()* que simula que el empleado está vacunando a UNA persona.

11. Simular la atención en una Terminal de Micros que posee 3 puestos para hisopar a ***150 pasajeros***. En cada puesto hay ***una Enfermera*** que atiende a los pasajeros de acuerdo con el orden de llegada al mismo. Cuando llega un pasajero se dirige al puesto que tenga menos gente esperando. Espera a que la enfermera correspondiente lo llame para hisoparlo, y luego se retira. ***Nota:*** sólo deben usar procesos Pasajero y Enfermera. Además, suponer que existe una función *Hisopar()* que simula la atención del pasajero por parte de la enfermera correspondiente.