Posibles soluciones a los ejercicios del parcial práctico de memoria compartida del 09-12-24

Importante: las soluciones que se muestran a continuación no son las únicas que se pueden considerar correctas para los ejercicios planteados.

1. Existe una sala de cine 3D, a las que asisten N personas a ver una película. Antes de entrar a la salsa, los asistentes deben retirar los anteojos 3D en la máquina repartidora que se encuentra en la entrada. Se debe simular el uso de la máquina repartidora de anteojos 3D, con capacidad para A anteojos (A < N). Además, existen un repositor encargado de reponer los anteojos en la máquina cuando se agotan. Los usuarios usan la máquina según el orden de llegada. Cuando les toca usarla, sacan un par de anteojos y luego se dirigen a la sala. En caso de que la máquina se quede sin anteojos, entonces le debe avisar al repositor para que cargue nuevamente la máquina en forma completa. Luego de la recarga, saca un par de anteojos y se retira. Implemente un programa que permita resolver el problema anterior usando **SEMÁFOROS**. **Nota:** maximizar la concurrencia; la reposición de anteojos no debe impedir que otros asistentes puedan agregarse a la fila.

```
sem mutex = 1;
sem espera[N] = ([N] 0);
sem reponer = 0;
sem finReponer = 0;

Process Asistente [id: 0..N-1] {
      int i, auxU;

      P(mutex);
      if (not libre) {
            push(colaMaq, id);
            V(mutex);
            P(espera[id]);
      }
      else {
            libre = false;
            V(mutex);
      }
      if (cantAnteojos == 0) {
            V(reponer);
            P(finReponer);
      }
      //saca anteojos
      cantAnteojos--;
      P(mutex);
      if (empty(colaMaq))
            libre = true
      else {
            pop(colaMaq, auxU);
            V(espera[auxU]);
      }
      V(mutex);
}
```

```
bool libre = true;
queue colaMaq;
int cantAnteojos = A;


Process Repositor{
      while (true) {
            P(reponer);
            //reponer la máquina
            cantAnteojos = A;
            V(finReponer);
      }
}
```

2. En el Registro de la Propiedad se pueden realizar 4 trámites administrativos diferentes. Para cada trámite, hay un puesto de atención específico. Existen 100 personas que se dirigen a la oficina para resolver un trámite particular. La persona deja su trámite en el puesto correspondiente y espera a que le entreguen el resultado. El puesto atiende a las personas que le corresponden de acuerdo con el orden de llegada. Implemente un programa que permita resolver el problema anterior usando **MONITORES**. **Notas**: maximizar la concurrencia; todos los procesos deben terminar; la función *obtenerPuesto()* retorna el número de puesto al que la persona debe dirigirse para su trámite; la función *obtenerTrámite()* retorna el trámite a realizar; la función *procesarTrámite(t)* procesa el trámite recibido como entrada y retorna su resultado.

```
Process Persona[id: 0..99]
{ text tramite, res;
  int numP;
  numP = ObtenrPuesto();
  tramite = ObtenerTramite();
  Puesto[numP].Pedido (id, tramite, res);
}

Process Empleado[id: 0..3]
{ text datos, res;
  int idP;
  for (i=0; i<25; i++)
    { Puesto[id].Siguiente (idP, datos);
      res = procesarTrámite(datos);
      Puesto[id].Resultado (idP, res);
    }
}

MONITOR Puesto[id:0..3]
{  cola solicitudes;
   text resultados[C];
   cond empl, espera;

   Procedure Pedido (id: int, tramite: text, res: OUT text)
   { push (solicitudes, (id, tramite));
     signal (empl);
     wait (espera);
     res = resultados[id];
   }

   Procedure Siguiente (id: OUT int, datos: OUT text)
   { if ( empty(solicitudes) ) wait (empl);
     pop (solicitudes, (id, datos));
   }

   Procedure Resultado (id: int, res: text)
   { resultados[id] = res;
     signal (espera);
   }
}
```