

Possibles soluciones a los ejercicios del parcial práctico del 15-12-25

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

1. Implemente una solución para el siguiente problema usando **SEMÁFOROS**. Se debe simular el procesamiento de un sistema clasificador de 365 datasets meteorológicos. Todos los datasets se encuentran ya cargados en una estructura de datos. Para procesarla, el sistema dispone de 5 workers que trabajan colaborativamente procesando los datasets de a uno por vez. Cada procesamiento toma exactamente el mismo tiempo y para realizarlo los workers disponen de la función *Procesar(d)*, la cual retorna como resultado un booleano que indica si los datos corresponden a un clima tropical (true) o no (false). De acuerdo con el resultado, cada worker inserta el dataset en una cola de "*tropicales*" o de "*no-tropicales*". **Nota:** maximizar concurrencia.

```

sem sem_d = 1;
sem sem_trop = 1;
sem sem_notrop = 1;
Datasets ds [365] = ...; // Ya cargada
Queue tropicales, notropicales;

Process worker[id=0..4] {
    // determinar sección de datos para el worker
    int strip = 365 / 5;
    int ini = id*strip;
    int fin = (id+1)*strip-1;
    // procesar datos de worker
    for i := ini to fin do
        bool res = Procesar(ds[i]);
        // de acuerdo al resultado lo encolo. cada cola es independiente.
        if (res == true) then
            P(sem_trop);
            push(tropicales,d);
            V(sem_trop);
        else
            P(sem_notrop);
            push(notropicales,d);
            V(sem_notrop);
        end
    end
}

```

Comentario: otra solución adecuada consiste en que cada worker emplee colas locales para clasificar los datasets que procesa y, antes de finalizar, sí encolar en las colas globales usando semáforos para garantizar la exclusión mutua.

2. En el comedor de un establecimiento laboral asisten C comensales. Cada comensal lleva su propio almuerzo y lo calienta usando el único microondas disponible para todos los comensales. El microondas es usado en forma exclusiva por cada comensal de acuerdo con el orden de llegada. Implemente una solución utilizando **SEMÁFOROS** que sólo emplee procesos Comensales. **Nota:** la función *UsarMicroondas()* le permite calentar el almuerzo al comensal.

```
Sem s=1;
Sem esperar[C]={ [C] 0}
Boolean libre = true;
Queue cola;

Process Comensal[id=0..C-1] {
    // solicita acceso al microondas
    P(s)
    if (libre==false) {
        push(cola,id);
        V(s)
        P(esperar[id]);
    }
    else
        libre=false;
    V(s);
end
// usa microondas con exclusión mutua
UsarMicroondas();
// libera microondas para que otro pueda usarlo
P(s)
if (not empty(cola)) then
    V(esperar[pop(cola)]);
else
    libre=true;
V(s);
}
```

3. En un cine se utiliza una terminal para el canje de entradas vendidas en forma anticipada por la web. Existen N asistentes que deben usar la terminal y un Empleado del cine que administra el acceso de acuerdo con el orden dado por la edad (cuando la terminal está libre, deja pasar al de mayor edad de entre los que están esperando por usarla). La terminal sólo puede ser usada por una persona a la vez. Implemente una solución al problema con **MONITORES** utilizando procesos para representar a los Asistentes y al Empleado. **Notas:** existe la función *CanjeeEntrada()* que representa el uso de la terminal; cada asistente conoce su edad mediante la función *obtenerEdad()*.

```
Monitor AdminTerminal {
    cond colas[N], autoridad, fin;
    Queue<int,int> esperando;

    procedure llegada (int id, int edad) {
        push(esperando, (edad, id)); // inserta ordenado por edad
        signal(autoridad);
        wait(colas[id]);
    }

    procedure salida () {
        signal(fin);
    }

    procedure siguiente () {
        if (empty(esperando))
            wait (autoridad);
        int id = pop(esperando);
        signal(colas[id]);
        wait (fin);
    }
}

Process Cliente [i: 1..N]{
    int edad = obtenerEdad();
    // solicitar acceso
    AdminTerminal.llegada(i,edad);
    // usar terminal
    CanjeeEntrada();
    // liberar
    AdminTerminal.salida();
}

Process Empleado {
    while (true) {
        // dar acceso para usar terminal a la siguiente persona
        AdminTerminal.siguiente();
    }
}
```