

Práctica 5 – Rendezvous (ADA)

CONSIDERACIONES PARA RESOLVER LOS EJERCICIOS:

1. NO SE PUEDE USAR VARIABLES COMPARTIDAS
2. Declaración de tareas
 - Especificación de tareas sin ENTRY's (nadie le puede hacer llamados).
TASK Nombre;
TASK TYPE Nombre;
 - Especificación de tareas con ENTRY's (le puede hacer llamados). Los entry's funcionan de manera semejante los procedimientos: solo pueden recibir o enviar información por medio de los parámetros del entry. NO RETORNAN VALORES COMO LAS FUNCIONES
TASK [TYPE] Nombre IS
 ENTRY e1;
 ENTRY e2 (p1: IN integer; p2: OUT char; p3: IN OUT float);
END Nombre;
 - Cuerpo de las tareas.
TASK BODY Nombre IS
 Código que realiza la Tarea;
END Nombre;
3. Sincronización y comunicación entre tareas
 - *Entry call* para enviar información (o avisar algún evento).
NombreTarea.NombreEntry (parametros);
 - *Accept* para atender un pedido de entry call sin cuerpo (sólo para recibir el aviso de un evento para sincronización).
ACCEPT NombreEntry (p1: IN integer; p3: IN OUT float);
 - *Accept* para atender un pedido de entry call con cuerpo.
ACCEPT NombreEntry (p1: IN integer; p3: IN OUT float) do
 Cuerpo del accept donde se puede acceder a los parámetros p1 y p3.
 Fuera del entry estos parámetros no se pueden usar.
END NombreEntry;
 - El accept se puede hacer en el cuerpo de la tarea que ha declarado el entry en su especificación. Los entry call se pueden hacer en cualquier tarea o en el programa principal.
 - Tanto el entry call como el accept son bloqueantes, ambas tareas continúan trabajando cuando el cuerpo del accept ha terminado su ejecución.

4. Select para ENTRY CALL.

- Select ...OR DELAY: espera a lo sumo x tiempo a que la tarea correspondiente haga el accept del entry call realizado. Si pasó el tiempo entonces realiza el código opcional.

```
SELECT
    NombreTarea.NombreEntry(Parametros);
    Sentencias;
OR DELAY x
    Código opcional;
END SELECT;
```

- Select ...ELSE: si la tarea correspondiente no puede realizar el accept inmediatamente (en el momento que el procesador está ejecutando esa línea de código) entonces se ejecuta el código opcional.

```
SELECT
    NombreTarea.NombreEntry(Parametros);
    Sentencias;
ELSE
    Código opcional;
END SELECT;
```

- En los select para entry call sólo puede ponerse un entry call y una única opción (OR DELAY o ELSE);

5. Select para ACCEPT.

- En los select para los accept puede haber más de una alternativa de accept, pero no se puede haber alternativas de entry call (no se puede mezclar accept con entries). Cada alternativa de ACCEPT puede ser o no condicional (WHEN).

```
SELECT
    ACCEPT e1 (parámetros);
    Sentencias1;
OR
    ACCEPT e2 (parámetros) IS cuerpo; END e2;
OR
    WHEN (condición) => ACCEPT e3 (parámetros) IS cuerpo; END e3;
    Sentencias3
END SELECT;
```

Funcionamiento: Se evalúa la condición booleana del WHEN de cada alternativa (si no lo tiene se considera TRUE). Si todas son FALSAS se sale del select. Sino, de las alternativas cuya condición es verdadera se elige en forma no determinística una que pueda ejecutarse inmediatamente (es decir que tiene un entry call pendiente). Si ninguna de ellas se puede ejecutar inmediatamente el select se bloquea hasta que haya un entry call para alguna alternativa cuya condición sea TRUE.

- Se puede poner una opción OR DELAY o ELSE.
- Dentro de la condición booleana de una alternativa (en el WHEN) se puede preguntar por la cantidad de entry call pendientes de cualquier entry de la tarea.

NombreEntry'count

- Después de escribir una condición por medio de un WHEN siempre se debe escribir un accept.

1. Se requiere modelar un puente de un solo sentido, el puente solo soporta el peso de 5 unidades de peso. Cada auto pesa 1 unidad, cada camioneta pesa 2 unidades y cada camión 3 unidades. Suponga que hay una cantidad innumerable de vehículos (A autos, B camionetas y C camiones).
 - a. Realice la solución suponiendo que todos los vehículos tienen la misma prioridad.
 - b. Modifique la solución para que tengan mayor prioridad los camiones que el resto de los vehículos.
2. Se quiere modelar la cola de un banco que atiende un solo empleado, los clientes llegan y si esperan más de 10 minutos se retiran.
3. Se dispone de un sistema compuesto por **1 central** y **2 procesos**. Los procesos envían señales a la central. La central comienza su ejecución tomando una señal del proceso 1, luego toma aleatoriamente señales de cualquiera de los dos indefinidamente. Al recibir una señal de proceso 2, recibe señales del mismo proceso durante 3 minutos.
El proceso 1 envía una señal que es considerada vieja (se desecha) si en 2 minutos no fue recibida.
El proceso 2 envía una señal, si no es recibida en ese instante espera 1 minuto y vuelve a mandarla (no se desecha).
4. En una clínica existe un **médico** de guardia que recibe continuamente peticiones de atención de las **E enfermeras** que trabajan en su piso y de las **P personas** que llegan a la clínica ser atendidos.
Cuando una *persona* necesita que la atiendan espera a lo sumo 5 minutos a que el médico lo haga, si pasado ese tiempo no lo hace, espera 10 minutos y vuelve a requerir la atención del médico. Si no es atendida tres veces, se enoja y se retira de la clínica.
Cuando una *enfermera* requiere la atención del médico, si este no lo atiende inmediatamente le hace una nota y se la deja en el consultorio para que esta resuelva su pedido en el momento que pueda (el pedido puede ser que el médico le firme algún papel). Cuando la petición ha sido recibida por el médico o la nota ha sido dejada en el escritorio, continúa trabajando y haciendo más peticiones.
El *médico* atiende los pedidos dándoles prioridad a los enfermos que llegan para ser atendidos. Cuando atiende un pedido, recibe la solicitud y la procesa durante un cierto tiempo. Cuando está libre aprovecha a procesar las notas dejadas por las enfermeras.
5. En un sistema para acreditar carreras universitarias, hay UN Servidor que atiende pedidos de U Usuarios de a uno a la vez y de acuerdo con el orden en que se hacen los pedidos. Cada usuario trabaja en el documento a presentar, y luego lo envía al servidor; espera la respuesta del mismo que le indica si está todo bien o hay algún error. Mientras haya algún error vuelve a trabajar con el documento y a enviarlo al servidor. Cuando el servidor le responde que está todo bien el usuario se retira. Cuando un usuario envía un pedido espera a lo sumo 2 minutos a que sea recibido por el servidor, pasado ese tiempo espera un minuto y vuelve a intentarlo (usando el mismo documento).

6. En una playa hay 5 equipos de 4 personas cada uno (en total son 20 personas donde cada una conoce previamente a que equipo pertenece). Cuando las personas van llegando esperan con los de su equipo hasta que el mismo esté completo (hayan llegado los 4 integrantes), a partir de ese momento el equipo comienza a jugar. El juego consiste en que cada integrante del grupo junta 15 monedas de a una en una playa (las monedas pueden ser de 1, 2 o 5 pesos) y se suman los montos de las 60 monedas conseguidas en el grupo. Al finalizar cada persona debe conocer grupo que más dinero junta. **Nota:** maximizar la concurrencia. Suponga que para simular la búsqueda de una moneda por parte de una persona existe una función *Moneda()* que retorna el valor de la moneda encontrada.
7. Hay un sistema de reconocimiento de huellas dactilares de la policía que tiene 8 *Servidores* para realizar el reconocimiento, cada uno de ellos trabajando con una Base de Datos propia; a su vez hay un *Especialista* que utiliza indefinidamente. El sistema funciona de la siguiente manera: el *Especialista* toma una imagen de una huella (TEST) y se la envía a los servidores para que cada uno de ellos le devuelva el código y el valor de similitud de la huella que más se asemeja a TEST en su BD; al final del procesamiento, el especialista debe conocer el código de la huella con mayor valor de similitud entre las devueltas por los 8 servidores. Cuando ha terminado de procesar una huella comienza nuevamente todo el ciclo. **Nota:** suponga que existe una función *Buscar(test, código, valor)* que utiliza cada *Servidor* donde recibe como parámetro de entrada la huella *test*, y devuelve como parámetros de salida el *código* y el *valor de similitud* de la huella más parecida a *test* en la BD correspondiente. Maximizar la concurrencia y no generar demora innecesaria.
8. Una empresa de limpieza se encarga de recolectar residuos en una ciudad por medio de 3 camiones. Hay *P personas* que hacen continuos reclamos hasta que uno de los camiones pase por su casa. Cada persona hace un reclamo, espera a lo sumo 15 minutos a que llegue un camión y si no vuelve a hacer el reclamo y a esperar a lo sumo 15 minutos a que llegue un camión y así sucesivamente hasta que el camión llegue y recolece los residuos; en ese momento deja de hacer reclamos y se va. Cuando un camión está libre la empresa lo envía a la casa de la persona que más reclamos ha hecho sin ser atendido. **Nota:** maximizar la concurrencia.
9. En una clase hay un docente que les toma un examen escrito a 25 alumnos. El docente espera a que todos los alumnos hayan llegado para entregarles el enunciado. Cada alumno resuelve el examen, se lo entrega al docente y luego espera a que este le indique la nota. **Nota:** suponga que existe una función *Resolver_Examen* llamada por los alumnos para simular la resolución del examen, y otra *Corregir_Examen* llamada por el docente para simular la corrección de un examen.

10. Hay un sitio web para identificación genética que resuelve pedidos de N clientes. Cada cliente trabaja continuamente de la siguiente manera: genera la secuencia de ADN, la envía al sitio web para evaluar y espera el resultado; después de esto puede comenzar a generar la siguiente secuencia de ADN. Para resolver estos pedidos el sitio web cuenta con 5 servidores idénticos que atienden los pedidos de acuerdo al orden de llegada (cada pedido es atendido por un único servidor). **Nota:** maximizar la concurrencia. Suponga que los servidores tienen una función *ResolverAnálisis* que recibe la secuencia de ADN y devuelve un entero con el resultado.
11. En un negocio hay un empleado para atender los pedidos de clientes de N clientes; algunos clientes son ancianos o embarazadas. El empleado atiende los pedidos de acuerdo a la siguiente prioridad: primero las embarazadas, luego los ancianos, y por último el resto. Cada cliente hace sólo un pedido de la siguiente manera: en el caso de las embarazadas, si no son atendidas inmediatamente se retiran; en el caso de los ancianos, esperan a lo sumo 5 minutos a ser atendidos, y si no se retira; cualquier otro cliente espera si o si hasta ser atendido. El empleado atiende a los clientes de acuerdo al orden de llegada, pero manteniendo las siguientes prioridades: primero las embarazadas, luego los ancianos y luego el resto. **Nota:** suponga que existe una función *AtenderPedido()* que simula que el empleado está atendiendo a un cliente.
12. Simular el funcionamiento de un Complejo de Canchas de Futbol 5 que posee 8 canchas y donde hay UN encargado. Al complejo acuden 80 jugadores divididos en 8 grupos (cada grupo tienen las 10 personas que jugarán en una cancha), cada jugador ya conoce a qué grupo pertenece. Cuando un jugador llega le indica al encargado el grupo al que pertenece para que este le indique el número de cancha a la cual debe ir, y se dirige a ella. Cuando los 10 integrantes de un grupo han llegado a la cancha que les corresponde, comienzan a jugar el partido que dura 50 minutos; y luego se retiran. El encargado asigna el número de cancha de acuerdo al orden en que llega el primer jugador de cada grupo. **Nota:** cuando el encargado ya atendió a los 80 jugadores debe terminar su ejecución.
13. En una empresa de turismo hay 5 micros que van a diferentes ciudades (cada micro tiene lugar para 30 pasajeros). Cada micro espera a que suban sus 30 pasajeros y luego realiza el viaje (el viaje dura un tiempo aleatorio entre 3 y 5 horas). Hay 150 personas que deben viajar en alguno de los micros (cada persona ya sabe a cuál micro debe subir); cuando una persona llega sube al micro correspondiente, y luego debe esperar a que el viaje finalice para terminar su ejecución. **Nota:** todas las tareas deben terminar.