Departamento de Arquitectura

Examen de Sistemas Operativos

22 de diciembre de 2022

Lea detenidamente las siguientes instrucciones. No cumplir los requerimientos puede implicar la pérdida del examen.

Formato

- Indique su nombre completo y número de cédula en cada hoja (no se corregirán las hojas sin nombre). Numere todas las hojas e indique la cantidad total de hojas en la primera.
- Empiece cada problema en una hoja nueva.
- Si se entregan varias versiones de un problema solo se corregirá la primera de ellas.

Dudas

- Sólo se contestarán dudas de letra.
- No se aceptarán dudas en los últimos 30 minutos del examen.

Material

- El examen es SIN material (no puede utilizarse ningún apunte, libro ni calculadora). Sólo puede tenerse las hojas del examen, lápiz, goma y lapicera en su banco. Todas sus demás pertenencias debe colocarlas en el piso debajo de su asiento.

Aprobación

- Para aprobar el examen se debe tener un mínimo de 60 puntos.

Finalización

- El examen dura **3 horas**.
- Al momento de finalizar el examen no se podrá escribir absolutamente nada en las hojas, debiéndose parar e ir a la fila de entrega. Identificar cada una de las hojas con nombre, cédula y numeración forma parte de la duración del examen.

Problema 1 (30 pts)

(a) (5 pts) Describa como funciona la protección de CPU. ¿Por qué no se puede asegurar la protección de CPU solamente con un algoritmo de planificación adecuado?

(b) (5 pts) Describa dos métodos para la asignación de espacio de disco. Mencione una ventaja y una desventaja de cada uno de ellos con respecto al otro.

(c) (5 pts) ¿Por qué en el PCB (Process Control Block) se reserva lugar para los registros de la CPU? Incluya en la respuesta una explicación de qué problema ocurriría si este lugar no existiera.

(d) (5 pts) En un sistema con paginación. ¿Qué ventaja y qué desventaja tiene agregar más jerarquías de tablas de página?

(e) (5 pts) Describa los pasos que sigue una aplicación de usuario para invocar una llamada al sistema.

(f) (5 pts) En general los sistemas operativos brindan un servicio de buffering para el subsistema de entrada/salida, ¿qué motiva brindar este servicio?

Problema 2 (35 pts)

Se tiene un sistema operativo simétrico multiprogramado con un planificador con dos colas de prioridad (alta/baja) con retroalimentación (multi-level feedback queue con dos niveles de prioridad). Un proceso de alta prioridad no puede ver restrasada su ejecución por uno de baja proridad. En cada una de las dos colas se utiliza un algoritmo round-robin con quantum de 2 unidades de tiempo.

Los procesos al ser creados se les asignan la prioridad alta, la cual es modificada según los siguientes criterios:

- Un pasa a la cola de baja prioridad si utilizó el recurso procesador durante las 2 últimas unidades de tiempo.
- Un proceso pasa a la cola de alta prioridad si en las últimas 4 unidades de tiempo no ha usado nada el recurso procesador.

A su vez, cuando un proceso pasa a un estado bloqueado su quantum es reiniciado y siempre que se producen empates al ingresar a una cola, se desempata favoreciendo al proceso con menor PID. Sean dos proceso con PID 1 y 2 creados en el instante t=0 que ejecutan durante 4 unidad de tiempo, sin bloquearse. Además, se tiene un proceso con PID 3 creado en el instante t=1 que ejecuta lo siguiente:

1. Ejecuta durante 1 unidad de tiempo.
2. Se bloquea durante 1 unidades de tiempo.
3. Ejecuta durante 1 unidad de tiempo.
4. Se bloquea durante 1 unidad de tiempo.
5. Ejecuta durante 1 unidad de tiempo.

Se pide:

(a) (6 pts) Presente un esquema y describa los diferentes estados y transiciones por los que puede pasar un proceso en el planificador planteado, modelando los procesos de alta y baja prioridad.

Solución:
Ver teórico por descripciones de estados y transiciones

(b) (13 pts) Asumiendo un sistema monoprocesador, realice un esquema de planificación (tiempo vs. procesos), en que se indique el estado de cada proceso en cada intervalo de tiempo. Además, se debe indicar el nivel de prioridad (alta/baja) y su posición en la cola correspondiente para cada proceso.

Solución:

PID												
1	E[A]	E[A]	L[1B]	L[1B]	L[1B]	E[B]	L[2B]	L[1B]	L[1B]	E[B]	T	T
2	L[1A]	L[1A]	E[A]	E[A]	L[2B]	L[1B]	L[1B]	E[B]	L[2B]	L[1B]	E[B]	T
3	-	L[2A]	L[1A]	L[1A]	E[A]	B[A]	E[A]	B[A]	E[A]	T	T	T
	1	2	3	4	5	6	7	8	9	10	11	12

Donde la letra A/B representa prioridad Alta/Baja y el número en el estado listo es la posición en la cola correspondiente

(c) Defina y calcule las siguientes métricas para la parte anterior.

i. (4 pts) Tiempo de retorno del proceso con PID 1

Solución: Ver teórico por definición. Tiempo de retorno de PID 1 es: 10-0=10

II. (4 pts) Tiempo de espera del proceso con PID 3

Solución: Ver teórico por definición. Tiempo de espera de PID 3 es: 5-2=3

(d) (8 pts) Considere ahora una arquitectura con dos procesadores. Realice otro esquema de planificación (tiempo vs. procesos), en que se indique el estado de cada proceso en cada intervalo de tiempo, considerando esta nueva arquitectura. Como antes, se debe indicar el nivel de prioridad (alta/baja) y su posición en la cola correspondiente para cada proceso.

Solución:

PID								
1	E[A]	E[A]	E[B]	E[B]	T	T	T	T
2	E[A]	E[A]	L[1B]	E[B]	E[B]	T	T	T
3	-	L[1A]	E[A]	B[A]	E[A]	B[A]	E[A]	T
	1	2	3	4	5	6	7	8

Nota: Asuma que el tiempo requerido por el sistema operativo y el hardware son despreciables.

Problema 3 (35 pts)

En la fábrica de juguetes de Papá Noél trabajan **duendes** incansablemente envolviendo regalos. Durante su ciclo de trabajo, cada duende primero accede a un depósito de donde retira un regalo. El **depósito** es un tanto chico y solo permite que 5 duendes accedan a la misma vez. Los duendes adicionales deben esperar fuera, en orden de llegada, hasta que el depósito se libere.

Una vez que tiene su regalo, el duende se acerca a la **mesa de trabajo**, que siempre tiene suficiente espacio para trabajar, y ejecuta la función *envolver_regalo()*. Una vez finalizado el envoltorio el duende procede a abandonar la mesa y descansa antes de volver a comenzar su ciclo de trabajo.

Papá Noél, en cambio, se pasa el día descansando, pero ocasionalmente realiza una inspección de la fábrica que incluye el depósito y la mesa de trabajo. Los duendes no pueden acceder al depósito hasta que finalice la inspección de toda la fábrica (depósito y mesa de trabajo) y tampoco pueden comenzar a envolver nuevos regalos en la mesa de trabajo (deben esperar en una fila para acceder)

Papá Noél inspecciona primero el depósito, para el cual tiene prioridad de acceso y debe acceder cuando el depósito esté vacío. Una vez inspeccionado el depósito, inspecciona los envoltorios en la mesa de trabajo para lo cual no requiere de un acceso exclusivo. Los duendes que ya se encontraban trabajando, pueden continuar ejecutando la función *envolver_regalo()*, pero si esta finaliza antes de que Papá Noél haya finalizado la inspección, el duende debe esperar a que finalice toda la inspección para poder retirarse a descansar.

Se dispone de los siguientes procedimientos auxiliares:

- `buscar_regalo()`: Ejecutado por los duendes dentro del depósito para obtener un regalo para envolver.
- `envolver_regalo()`: Ejecutado por los duentes en la mesa para envolver un regalo.
- `inspeccionar_deposito()`: Procedimiento ejecutado por Papá Noél para inspeccionar el depósito.
- `inspeccionar_mesa_de_trabajo()`: Procedimiento ejecutado por Papá Noél para inspeccionar la mesa de trabajo.
- `descansar()`: Ejecutado por los duendes y Papá Noél.

(a) (35 pts) Implemente utilizando **ADA** las tasks **Duende**, **Papá Noél**, **Depósito** y **Mesa**. No está permitido usar tareas auxiliares.

Solución:

```
program fabrica_juguetes is
    task Deposito is
        entry LLEGA_DUENDE;
        entry LLEGA_PAPA_NOEL;
        entry SALE_DUENDE;
        entry SALE_PAPA_NOEL;
    end Deposito;
    task body Deposito is
        var cantDuendes: Integer;
        var estaPapaNoel: Boolean;
    begin
        cantDuendes := 0;
        estaPapaNoel := False;
        loop
            select
                when cantDuendes == 0  =>
                    accept LLEGA_PAPA_NOEL;
```

```
                estaPapaNoel := True;
        or
            accept SALE_PAPA_NOEL;
            estaPapaNoel := False;
        or
            when LLEGA_PAPA_NOEL'Count == 0 and not estaPapaNoel
            and cantDuendes < 5 =>
                accept LLEGA_DUENDE;
                cantDuendes := cantDuendes + 1;
        or
            accept SALE_DUENDE;
            cantDuendes := cantDuendes - 1;
        end select;
    end loop;
end Deposito;

task Mesa is
    entry LLEGA_DUENDE;
    entry LLEGA_PAPA_NOEL;
    entry SALE_DUENDE;
    entry SALE_PAPA_NOEL;
end Mesa;
task body Mesa is
    var estaPapaNoel: Boolean;
begin
    estaPapaNoel := False;
    loop
        select
            accept LLEGA_PAPA_NOEL;
            estaPapaNoel := True;
        or
            accept SALE_PAPA_NOEL;
            estaPapaNoel := False;
        or
            when LLEGA_PAPA_NOEL'Count == 0 and
                not estaPapaNoel =>
                accept LLEGA_DUENDE;
        or not estaPapaNoel =>
            accept SALE_DUENDE;
        end select;
    end loop;
end Mesa;

task type Duende;
task body Duende is
    begin
        loop
            Deposito.LLEGA_DUENDE();
            buscar_regalo();
            Deposito.SALE_DUENDE();
```

```
                Mesa.LLEGA_DUENDE();
                envolver_regalo();
                Mesa.SALE_DUENDE();
            end loop
    end Duende;

    task type Papa_Noel;
    task body Papa_Noel is
        begin
            loop
                Mesa.LLEGA_PAPA_NOEL();
                Deposito.LLEGA_PAPA_NOEL();

                inspeccionar_deposito();
                inspeccionar_mesa_de_trabajo();

                Deposito.SALE_PAPA_NOEL();
                Mesa.SALE_PAPA_NOEL();
            end loop
    end Papa_Noel;

main begin
end program.
```