

Examen 5 de febrero de 2016

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, sin excepciones). Numere todas las hojas e indique la cantidad total de hojas que entrega en la primera.
- Escriba las hojas de un solo lado y empiece cada problema en una hoja nueva.
- Si se entregan varias versiones de un problema solo se corregirá el primero de ellos.

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 4 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 (32 puntos)

Para las siguientes partes, conteste justificando brevemente, cada una de las preguntas.

1. Describa tres servicios que debe brindar un sistema operativo.
2. Realice el diagrama de estados y transiciones de los procesos. Explique cada uno de sus elementos.
3. En un sistema con planificador de colas multinivel, en donde a cada nivel se le asigna un algoritmo de planificación round-robin con diferente quantum y teniendo mayor prioridad las colas con quantum más chico, ¿qué tipos de procesos (alto consumo CPU o alto consumo de operaciones de E/S) clasificarían para colas con quantum más bajo y cuáles otros para colas con quantum más alto.
4. Defina los siguientes conceptos:
 - a) Tiempo de respuesta
 - b) Tiempo de retorno
 - c) Tiempo de espera
5.
 - a) Compare los métodos de entrada/salida programada (Programmed IO) y mediante DMA.
 - b) Describa los pasos de una entrada/salida a través de un DMA.
6. Explique el sistema RAID 0+1 y compárelo con los sistemas RAID 0 y RAID 1.
7. Describa y compare los métodos de asignación de datos de archivos contiguo e indexado.
8. Describa brevemente tres métodos de planificación de disco.

Problema 2 (35 puntos)

Se desea implementar una gestión de memoria virtual mediante paginación en un equipo informático. En este equipo el bus de direcciones es de 32 bits, la unidad mínima direccionable es un byte, y el tamaño de la página es de 4KB. La entrada en la tabla de traducción de página es de 4 bytes, pero solo los 24 bits menos significativos son utilizables. Se desea poder controlar si una página se encuentra presente, si se ha accedido, si se ha modificado, así como también protegerla de lectura, escritura y ejecución. Para implementar esto se debe utilizar un bit individual para cada control.

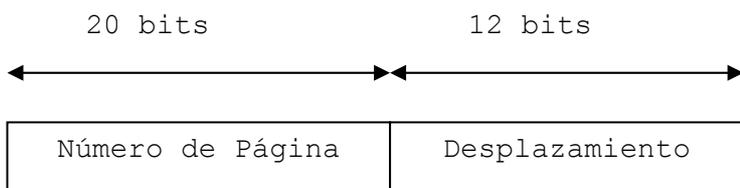
El sistema dispone de un dispositivo de almacenamiento secundario disponible exclusivamente como memoria swap para dar soporte a memoria virtual. Este dispositivo cuenta con bloques de 4 KB y una capacidad total de hasta 1.5 GB. Cuando una página no se encuentra en memoria real, en su entrada en la tabla de traducción se guarda el número de bloque del dispositivo de almacenamiento secundario donde reside la página. La numeración de bloques de disco comienza en '1', así que se reserva el número '0' para indicar que la página no tiene ningún bloque asignado.

Se pide:

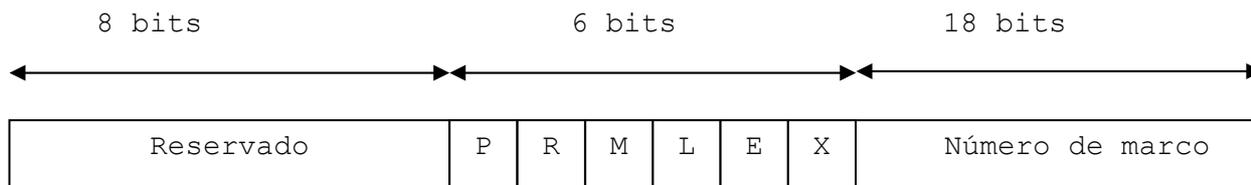
- a) Especifique los formatos de dirección lógica (virtual) y de la entrada en la tabla de traducción de página. Justifique su respuesta.
- b) Escriba el tamaño de la tabla de páginas (en bytes). Justifique su respuesta
- c) Escriba la cantidad máxima de memoria real soportada por este sistema (en bytes). Justifique su respuesta.
- d) Escriba la cantidad máxima de memoria virtual de la que dispone el sistema (en bytes). Justifique su respuesta.

Debido al gran tamaño en memoria de la tabla de páginas se optó por que ésta no tenga que estar completamente en memoria principal, sino que pueda estar también paginada.

- e) Indique si modificaría o no el algoritmo del módulo de traducción (MMU) para dar soporte a esta opción y en su caso en qué consistiría esta modificación. Justifique su respuesta.
- f) Describa en líneas generales cómo funcionaría la traducción de una dirección con esta opción.

Solución**a) Formato de dirección lógica (virtual)**

Para expresar un desplazamiento en una página de 4KB se necesitan 12 bits. Como la arquitectura es de 32 bits, quedan 20 para el número de página.



Donde los bits indican:

P: Presencia

R: Referencia

M: Modificación

L: Protección de lectura

E: Protección de escritura

X: Protección de ejecución

Cuando el bit de presencia indica que no está presente, si el número de marco es 0 es porque la página no está en el espacio de direccionamiento virtual del proceso.

b) La tabla de páginas tendrá 2^{20} entradas y como cada entrada ocupa 4 bytes, el tamaño de la Tabla de Páginas será de 4 MB. ($1 \text{ MB} = 2^{20}$ bytes)

c) Ya que el número máximo de marcos es 2^{18} marcos y cada marco es de 4 KB (4×2^{10} bytes), la cantidad máxima de memoria real será:

$$2^{18} \times 4 \text{ KB} = 2^{18} \times 4 \times 2^{10} = 2^{18} \times 2^2 \times 2^{10} = 2^{30} = 1 \text{ GB}$$

d)

La cantidad máxima de bloques de almacenamiento secundario direccionables por la tabla de página es $2^{18}-1$ (espacio direccionable por 18 bits, menos el valor '0' usado como bandera). Por lo tanto, el máximo espacio de almacenamiento secundario utilizable es 1 GB - 4 KB.

Para determinar la memoria virtual máxima de la que se puede disponer, tenemos que discutir en función del algoritmo de carga de páginas.

Si éste permite que hayan páginas en memoria que no tienen respaldo en almacenamiento secundario, entonces la cantidad máxima de memoria virtual disponible es de 1 GB (mem. física) + 1 GB - 4 KB (paginado) = 2 GB - 4 KB.

Si el algoritmo de carga de páginas no permite que hayan páginas en memoria sin respaldo en almacenamiento secundario, entonces la cantidad máxima de memoria virtual disponible es de 1 GB - 4 KB, que es la cantidad de memoria secundaria que disponemos.

e) Para paginar la tabla de páginas habría que modificar el algoritmo de traducción para que indexe dos veces usando una tabla de páginas de 2 niveles. Por ejemplo, utilizar los 10 bits más significativos de la dirección virtual para acceder a la entrada de la tabla de primer nivel (que

apunta a la tabla de segundo nivel) y luego con los siguientes 10 bits acceder a la entrada de la tabla interna. De esta manera se tendría una tabla externa con 2^{10} entradas de 4 bytes (total 4Kb) más 2^{10} tablas internas, cada una también de 4Kb.

f) Cuando se hace referencia a una dirección, se obtiene el número de página de ésta (20 bits superiores). Con dicho número y un identificador de proceso se pasa por la TLB (Translation Look-aside Buffer) para ver si se encuentra cacheado el mapeo. Si se encuentra el mapeo se procede a interpretar la entrada (cotejar permisos), dar una excepción si no se cumplen o proceder a realizar la transacción de memoria con el marco asociado. Si la entrada no se encuentra en la TLB (asumiremos que no hay mapeos internos hacia tablas del segundo nivel) entonces se procede a acceder a la tabla de primer nivel, ubicar la entrada correspondiente a la tabla de segundo nivel para hacer una nueva lectura para acceder a la entrada de segundo nivel. Luego de cotejados los permisos si la página no está en memoria se produce un fallo de página que hace que se traiga a memoria dicha página desde disco, continuando el proceso a partir de aquí como el caso anterior.

Problema 3 (33 puntos)

En la tienda de mascotas “La cuevita del hamster” están teniendo problemas serios para mantener contentos a los hamsters. Para evitar esto, se decidió construir un nuevo bebedero y una rueda para hacer ejercicio. Sin embargo, debido al bajo presupuesto, la rueda es muy pesada para un solo hamster y además el bebedero no soporta más de cinco hamsters a la vez.

Dado que los hamsters son naturalmente peleadores y territoriales al momento de la comida, si hay hamsters de distinto sexo a la vez en el bebedero, estos siempre se pelean hasta morir. Además, para evitar que los hamsters de un sexo se apoderen del bebedero, cuando llega alguien del sexo opuesto a los que están bebiendo, deberá evitar que sigan entrando hamsters del sexo de los que están bebiendo. También se desea evitar que quede olor a los animales del sexo opuesto en el bebedero por lo que antes de cambiar de sexo se debe dejar ventilar el bebedero.

Por otro lado, como la rueda es muy pesada para un hamster solo, deberá permitirse el acceso a la rueda cuando haya al menos tres hamsters para jugar. Los hamsters entrarán en grupos de a tres a la rueda y estos podrán dejar el juego también solamente en grupos de a tres. Como el acceso a la rueda es único, una vez que un hamster quiere entrar no deberá permitir que salgan hamsters hasta que haya entrado el grupo y viceversa.

A fin de evitar estos problemas, la tienda de mascotas quiere (con lo que le queda de presupuesto) implementar un sistema de control de hamsters que evite que estos se peleen y permita que se diviertan.

En resumen:

Este sistema deberá **evitar**:

- que haya hamsters de distinto sexo a la vez en el bebedero
- que haya más de cinco hamsters en el bebedero
- que entre al bebedero un nuevo hamster del mismo sexo de los que están bebiendo cuando hay uno del sexo opuesto esperando

y deberá **permitir**:

- que únicamente deje entrar o salir a la rueda de a tres hamsters por vez
- que se ventile el bebedero antes de cambiar el sexo de los animales que lo usan
- mientras se está ventilando el bebedero que también se pueda entrar o salir de la rueda

Notas:

- Existen múltiples instancias de los hamsters aburridos en el sistema
- Los hamsters SOLO ejecutarán el código presentado (son reacios al nuevo código)

Se dispone de la siguientes rutinas auxiliares:

- `que_soy() : {MACHO, HEMBRA}`
- `ventilar()`

Para esto los hamsters “ejecutarán” siempre este código para entrar a beber o para jugar en la rueda:

```
Procedure hamster_Aburrido()
Var
    que_hago: Integer;

Begin
    While (true) do
        Begin
            que_hago = pensar_un_rato();
            If que_hago = Beber then
                Begin
                    Quiero_entrar_al_bebedero();
                    Beber_agua();
                    Quiero_salir_del_bebedero()
                End
            Else
                If que_hago = Jugar then
                    Begin
                        Quiero_entrar_a_la_rueda();
                        Jugar_un_rato();
                        Quiero_salir_de_la_rueda()
                    End
                End
            End
        End
    End
End
```

Se pide implementar en Ada los procedimientos:

- Quiero_entrar_al_bebedero();
- Quiero_salir_del_bebedero();
- Quiero_entrar_a_la_rueda();
- Quiero_salir_de_la_rueda();

y las tareas que crea conveniente para modelar este problema

Solución:

```
procedure Quiero_entrar_al_bebedero is
  if que_soy() == macho
    Bebedero.entrar_macho();
  else
    Bebedero.entrar_hembra();
  end if;
end Quiero_entrar_al_bebedero;

procedure Quiero_salir_del_bebedero is
  Bebedero.salir();
end Quiero_salir_del_bebedero;

procedure Quiero_entrar_a_la_rueda is
  Rueda.entrar();
  Rueda.pasar();
end Quiero_entrar_a_la_rueda;

procedure Quiero_entrar_de_la_rueda is
  Rueda.salir();
  Rueda.pasar();
end Quiero_entrar_de_la_rueda;

task bebedero is
  entry entrar_hembra();
  entry entrar_macho();
end task

task body bebedero is
  hamsters : integer;
  sexo: {MACHO, HEMBRA, NULL};
begin
  hamsters = 0;
  sexo = NULL;
  loop
    select
      when hamsters == 0 or
        (hamsters < 5 and sexo == MACHO and entrar_hembra'count == 0) =>
        accept entrar_macho() do
          if (sexo = HEMBRA)
            ventilar();
          endif;
        end accept
        sexo = MACHO;
        hamsters = hamsters + 1;
      or
        when hamsters == 0 or
          (hamsters < 5 and sexo == HEMBRA and entrar_macho'count == 0) =>
          accept entrar_hembra() do
            if (sexo = MACHO)
              ventilar();
            endif;
          end accept
          sexo = HEMBRA;
          hamsters = hamsters + 1;
      or
        accept salir() do
        end accept;
        hamsters = hamsters - 1;
    endselect;
  end loop;
end task body bebedero;
```

```
    endloop;
end task

task Rueda is
    entry entrar();
    entry salir();
    entry pasar();
end task

task body Rueda is
begin
    loop
        select
            accept entrar();
            accept entrar();
            accept entrar();
        or
            accept salir();
            accept salir();
            accept salir();
        endselect;
        for i = 1 to 3
            accept pasar();
        endfor;
    endloop;
end task
```