Examen Febrero de 2015

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

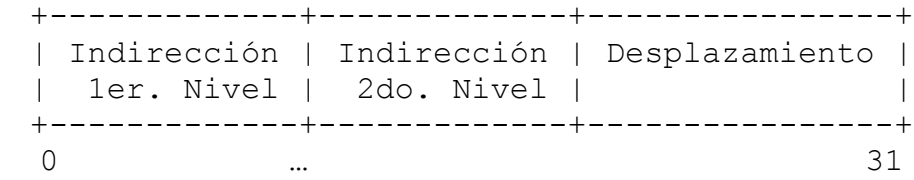
- **El examen dura 4 horas.**
- **Al momento de finalizar el examen no se podrá escribir absolutamente nada en las hojas, debiéndose parar e ir a la fila de entrega. Identificar cada una de las hojas con nombre, cédula y numeración forma parte de la duración del examen.**

Problema 1 (32 puntos)

1.
    i. Compare brevemente los sistemas monoprogramados vs. multiprogramados, indicando ventajas y desventajas.
    ii. Describa los tipos de procesamiento que existen en un sistema multiprocesador.
2.
    i. Indique para qué sirve el modo dual.
    ii. Indique cómo puede un SO evitar el excesivo uso de CPU por parte de un proceso.
3. Describa 3 servicios que debe proveer un Sistema Operativo.
4.
    i. Indique las ventajas del uso de DMA (Direct Memory Access).
    ii. Describa los pasos necesarios para realizar una transferencia mediante DMA.
5. Describa tres métodos utilizados para administrar los bloques de disco libres.
6. Describa qué entiende por fragmentación interna y fragmentación externa.
7. Describa los campos principales de una estructura de datos para representar un archivo en una FAT.
8. Describa el modelo de Conjunto de Trabajo (Working Set).

Problema 2 (35 puntos)

Sea un sistema que implementa memoria virtual con paginación bajo demanda. En donde las direcciones virtuales del sistema son de 32 bits y se realiza una traducción de direcciones a través de dos niveles de tabla de página. Las páginas tienen un tamaño de 4096 Bytes y las direcciones virtuales tienen el siguiente formato:



Responder las siguientes preguntas justificando:

1. Cuál es el tamaño de los marcos (*frames*)? **(3 pts.)**.
2. Determine el tamaño en bits de cada componente de la dirección virtual teniendo en cuenta que las entradas de la tabla de página son de 16 bits y se desea que la tabla de página de primer nivel ocupe sólo una página y completamente. **(5 pts.)**.
3. Asumiendo que el sistema cuenta con una cache TLB (*Translation Look-aside Buffer*) que está 'limpia' (no contiene 'cacheado' ningún valor) y que no cuenta con memorias cache entre el procesador y la memoria principal, determine la cantidad de accesos a memoria necesarios para leer un arreglo de 10000 caracteres: **(12 pts.).**

```
Var arreglo : Array[0..9999] of Char; // Tamaño del Char = 1 Byte.
  ...
for (i = 0; i < 10000; i++)
   ... arreglo[i] ...
```

Notas:
- Asuma que la dirección de comienzo de la variable arreglo es al principio de una página.
- No tome en cuenta los accesos para la variable **i**.

4. **El sistema tiene una estrategia de asignación de memoria local y utiliza un algoritmo de reemplazo LRU (*Least Recently Used*). Suponiendo que a un proceso se le asignan 4 marcos para utilizar, muestre mediante un esquema en el tiempo los fallos de páginas y el estado de la memoria si se realizan los siguientes accesos: (15 pts.)**

Tiempo	Dirección virtual		
	Primer nivel	Segundo nivel	Desplazamiento
t_1	10	3	2
t_2	10	4	2
t_3	11	3	100
t_4	10	3	520
t_5	12	2	128
t_6	12	2	130
t_7	14	1	768
t_8	11	3	50
t_9	10	3	2
t_{10}	10	4	4
t_{11}	12	2	132

Solución:

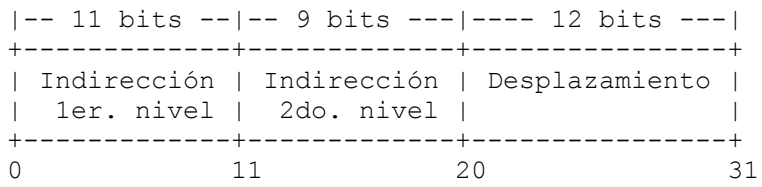
1) El tamaño de los marcos es de 4096 bytes dado que es igual al tamaño de las páginas.
2) Como el tamaño de la página es de 4096 bytes entonces se precisan 12 bits para direccionar cada byte dentro de la página.
Como las entradas en la tabla de páginas ocupan 16 bits y la tabla de primer nivel ocupa una única página completamente entonces, se tiene que

cantidad de entradas en la tabla de paginas = tamaño_página / tamaño_entradas

4096 bytes / 16 bits = 4096 bytes / 2 bytes = 2048 cantidad de entradas en la tabla de paginas

Entonces es necesario 11 bits para direccionar las 2048 entradas en la tabla de páginas. Para definir la cantidad de bits para direccionar la entrada en la tabla de páginas de segundo nivel alcanza con calcular la diferencia entre la cantidad de bits de las direcciones virtuales menos los bits de la para direccionar el primer nivel y desplazamiento.

32 bits - 12 bits - 11 bits = 9 bits



3) Dado que las páginas son de 4096 bytes y queremos acceder a 10000 bytes contiguos, los mismos serán almacenados en 3 páginas contiguas. De esa forma, se tiene un acceso para acceder a la página que contiene la tabla de primer nivel y leer la entrada que corresponde, esta entrada queda almacenado en la TLB. Luego, se realiza un acceso para acceder a la página que contiene la tabla de segundo nivel y leer el valor de la entrada correspondiente, este dato también queda almacenado en la TLB.
A seguir se realizan 4096 accesos para acceder a cada uno de los elemento cargado en la primer página.
Para leer la segunda página es necesario primero acceder a la página que contiene la tabla de segundo nivel y leer el valor de la entrada siguiente, este dato luego de leído queda almacenado en la TLB.
Se realizan otros 4096 accesos para acceder a cada uno de los elemento cargado en la segunda página.
A continuación, se tiene un acceso para acceder a la página que contiene la tabla de segundo nivel y leer la siguiente entrada de la tabla, este dato luego de leído se almacena en la TLB.
Finalmente, se realizan los 1808 accesos para acceder a cada uno de los elementos cargado en la tercera página.
Se obtiene entonces la siguiente cantidad de accesos:

1 + 1 + 4096 + 1 + 4096 + 1 + 1808 = 10004

Estos cálculos han tomado como supuesto que las tres páginas de datos se direccionan a partir de la misma entrada de primer nivel. En caso de que una de las tres páginas (la primera o la última) se direccione a partir de la entrada vecina del primer nivel se precisaría un acceso extra.

4) La página será identifica en el esquema por (entrada en la tabla de primer nivel, entrada en la tabla de segundo nivel)

Estado de los marcos:

tiempo	t00	t01	t02	t03	t04	t05	t06	t07	t08	t09	t10	t11
marco1		(10,3)	(10,3)	(10,3)	(10,3)	(10,3)	(10,3)	(10,3)	(10,3)	(10,3)	(10,3)	(10,3)
marco2			(10,4)	(10,4)	(10,4)	(10,4)	(10,4)	(14,1)	(14,1)	(14,1)	(14,1)	(12,2)
marco3				(11,3)	(11,3)	(11,3)	(11,3)	(11,3)	(11,3)	(11,3)	(11,3)	(11,3)
marco4						(12,2)	(12,2)	(12,2)	(12,2)	(12,2)	(10,4)	(10,4)

LRU		FALLO	FALLO	FALLO		FALLO		FALLO			FALLO	FALLO
tiempo	t01	t01	t02	t03	t04	t05	t06	t07	t08	t09	t10	t11
Mas RU		(10,3)	(10,4)	(11,3)	(10,3)	(12,2)	(12,2)	(14,1)	(11,3)	(10,3)	(10,4)	(12,2)
			(10,3)	(10,4)	(11,3)	(10,3)	(10,3)	(12,2)	(14,1)	(11,3)	(10,3)	(10,4)
				(10,3)	(10,4)	(11,3)	(11,3)	(10,3)	(12,2)	(14,1)	(11,3)	(10,3)
Menos RU						(10,4)	(10,4)	(11,3)	(10,3)	(12,2)	(14,1)	(11,3)

Luego de ejecutados los accesos a memoria tenemos que se generan 7 fallos de página y el estado de la memoria es:

- en el marco1 queda cargada la página (10,3).
- en el marco2 queda cargada la página (12,2).
- en el marco3 queda cargada la página (11,3).
- en el marco4 queda cargada la página (10,4).

Problema 3 (33 puntos)

Sea una empresa dedicada al reciclado de pilas. La misma dispone de una máquina receptora de pilas que tiene 5 receptores y que acepta 8 tipos de pilas.
Las pilas se guardarán en 8 recipientes distintos dependiendo del tipo de pila con capacidad para 50 pilas del mismo tipo cada uno.
Cuando un recipiente se llena, el encargado deberá ser avisado para que lo cambie. Si se ingresa una pila de este tipo el receptor utilizado quedará trancado hasta que el encargado cambie el recipiente.
La máquina deberá indicar el recipiente a cambiar a través del seteo de un evento. Por su parte, el encargado deberá chequear el evento para determinar cuál recipiente cambiar. No se deberá setear un evento si hay otro evento anterior que aún no fue chequeado por el encargado.
Cuando una pila es ingresada al receptor se ejecuta automáticamente el procedimiento ***NUEVA_PILA(num_receptor: in integer)*** que tendrá como único objetivo indicarle al receptor que tiene una nueva pila para recibir.

Se pide: Modelar este sistema utilizando **semáforos**, implementando el procedimiento tipo RECEPTOR y los procedimientos ENCARGADO y NUEVA_PILA.

Se dispone de las funciones:

- **num_receptor():integer**, que ejecutado desde el procedimiento RECEPTOR indica su numero de receptor.
- **tipo_pila(receptor: in integer):integer**, que da el tipo de pila ubicado en el receptor.
- **seteo_evento(recipiente: in integer)**: que setea el evento del recipiente a cambiar.
- **chequear_evento():integer**, que retorna el recipiente que hay que cambiar.
- **cambiar_recipiente(recipiente: in integer)**: que cambia el recipiente.
- **poner_pila (recipiente: in integer, lugar: in integer)**: que pone la pila en el lugar correspondiente del recipiente (*lugar* debe ser un número de 1 a 50).

Notas:

- Las bocas deberán poder funcionar concurrentemente.
- Se podrá colocar una cantidad no acotada de pilas en cada boca las que se irán encolando hasta ser puestas en su recipiente correspondiente.
- No se aceptarán soluciones con “busy waiting”.
- Se deberá cuidar especialmente que 2 o más pilas no sean puestas en el mismo lugar del recipiente.

Solución:

```
int count_recipiente [8];
semaphore mutex_recipiente [8];

semaphore receptor [5];
semaphore evento_pendiente, encargado;

procedure RECEPTOR() {
      int tipo;
      id = num_receptor();
      while(true) {
            P(receptor[id]);
            tipo = tipo_pila(id);
            P(mutex_recipiente[tipo]);
            poner_pila(tipo, ++count_recipiente[tipo]);
            if (count_recipiente[tipo] == 50) {
                  P(evento_pendiente)
                  seteo_evento(tipo);
                  V(encargado);
            }
            else {
                  V(mutex_recipiente[tipo]);
            }
      }
}

procedure NUEVA_PILA(num_receptor: in integer) {
      V(receptor[num_receptor]);
}

procedure ENCARGADO() {
      int recipiente;

      while(true) {
            P(encargado);
            recipiente = chequear_evento();
            cambiar_recipiente(recipiente);
            count_recipiente[recipiente] = 0;
            V(mutex_recipiente[recipiente]);
            V(evento_pendiente);
      }
}

int main () {
      for (int i=0: i<8; i++) {
            count_recipiente[i] = 0;
            init(mutex_recipiente[i], 1);
      }

      for (int i=0: i<5; i++) {
            init(receptor[i], 0);
      }

      init(evento_pendiente, 1);
      init(encargado, 0);
}
```