Sistemas Operativos

I. T. Informática de Sistemas

Córdoba, 18 de septiembre de 2003

1. Supóngase la siguiente solución al problema de la sección crítica con N procesos:

```
/* Variables globales compartidas. */
BOOLEAN PQuiereEntrar[N] = {FALSE, FALSE, ..., FALSE};
int turno = 0;

void
Proceso (int i)
{
  extern BOOLEAN PQuiereEntrar[N];
  extern int turno;
  int j;

  while (TRUE)
    {
      PQuiereEntrar[i] = TRUE;
      for (j = 0; j < N; j++)
        if (j != i && PQuiereEntrar[j])
          {
            PQuiereEntrar[i] = FALSE;
            while (turno != i && turno != 0);
            turno == i;

            PQuiereEntrar[i] = TRUE;
          }

      /* Sección crítica. */

      turno = (turno + 1) % N;
      PQuiereEntrar[i] = FALSE;

      /* Sección residual.  */
    }
}
```

Indique si esta solución respeta los principios para ser considerada una solución software correcta al problema de la exclusión mutua con N procesos, y explique su funcionamiento. En caso de no ser así, indique un ejemplo donde no se cumpliría alguna de las condiciones exigibles.

2. Suponga un proceso en UNIX que realiza las operaciones que se indican a continuación:

   a) El proceso es creado.

   b) El proceso es planificado para ejecutar por primera vez.

   c) El proceso ejecuta instrucciones que no necesitan llamadas al sistema y es planificado en varias ocasiones agotando siempre el cuanto de tiempo que le asigna el planificador.

   d) Mientras el proceso ejecuta en modo usuario se produce la finalización de una lectura de disco provocada por un fallo de página en otro proceso.

   e) El proceso realiza una escritura en el disco.

   f) El proceso realiza una llamada a la función `pthread_create()` para crear un hilo.

   g) El proceso realiza una lectura de una posición de memoria que no se encuentra en memoria real.

   h) El hilo creado por el proceso finaliza su ejecución.

   i) El proceso finaliza.

   - Indique por qué estados pasará el proceso a medida que va realizando las acciones anteriores.
   - Indique cómo se verán afectadas las estructuras de información del sistema, especialmente el PCB, a medida que se realizan las acciones anteriores.

3. Supóngase en un sistema la siguiente secuencia de procesos con los instantes de llegada y tiempos de ejecución de la ráfaga que se indican:

Proceso	**Llegada**	**Tiempo de ráfaga**	**Modo**	**Prioridad**
1	0	80 ms	Usuario	-
2	10	23 ms	Sistema	6
3	14	60 ms	Usuario	-
4	20	55 ms	Usuario	-
5	24	18 ms	Sistema	12
6	30	77 ms	Usuario	-
7	34	13 ms	Sistema	2

   El modelo de planificación consiste en un algoritmo Round-Robin con un cuanto $q = 20$. Para seleccionar el siguiente proceso a ejecutar se aplica una decisión basada en la prioridad. Los procesos del sistema tienen una prioridad fija. Los procesos de usuario calculan su prioridad en función de la siguiente ecuación:

   $P_i = P_b$ + (tiempo de ejecución acumulado) - (tiempo de espera acumulado)*CPU,

   donde CPU $\in [0, 1]$ es una factor que modula el peso del tiempo de espera en la formula anterior, y $P_b = 20$. La prioridad mayor es la numéricamente menor.

a) Describa el comportamiento del algoritmo para los valores del factor CPU $= \{0, 1, 10\}$.

b) Realice la planificación y el diagrama de Gant de los procesos indicados en la tabla para CPU$= 1$.

4. Sea el siguiente esqueleto de la solución al problema de la cena de los filósofos mediante el uso de monitores:

```
#define NPHILOSOPHERS   5

monitor Philosophers
condition Ok_to_eat[NPHILOSOPHERS];
int Fork[NPHILOSOPHERS], i;

void
Take_Fork (int I)
{
}

void
Release_Fork (int I)
{
}

/* Init area. */
for (i = 0; i < NPHILOSOPHERS; i++)
  Fork[i] = 2;

end monitor Philosophers

/* Life of a philospher. */
void
Philospher_Task (int I)
{
  while (TRUE)
    {
      /* Think. */
      Take_Fork (I);
      /* Eat. */
      Realease_Fork (I);
    }

}
```

Indique el seudocódigo de las functiones `Take_Fork()` y `Release_Fork()` para que el código implemente correctamente la solución al problema de la cena de los filósofos.

5. Dada la siguiente secuencia de referencias a memoria, de lectura y de escritura, y una memoria real de 4 marcos de página y una memoria virtual de 8 páginas:

```
2 1 3 2 4 5 2 3 4 7 6 1 1 6 5 4 0 1 2 7
E E L L L E L E E L L E L L E E L E L E
```

   a) Indique la cadena de reemplazo de este algoritmo utilizando un mecanismo de FIFO con segunda oportunidad.

   b) Considérese ahora que se aplica el algoritmo de reemplazo anterior utilizando un bit que marca las páginas modificadas, y reemplaza de forma preferente las páginas no modificadas. El tiempo para reemplazar la página se reduce en un 20 %, pero el número de fallos de página aumenta. ¿Qué porcentaje de aumento de fallos de página podría permitirse para que el algoritmo continúe siendo eficiente?

   c) Proponga como utilizaría un bit de modificado en el algoritmo FIFO con segunda oportunidad para favorecer el reemplazo de páginas no modificadas sin cambiar, en lo posible, la filosofía del algoritmo.

Nota: Justifique siempre todas las respuestas. Tiempo de realización: **4 horas**.